

# Online Steering of HEP Grid Applications \*

Daniel Lorenz<sup>(1,2)</sup>, Peter Buchholz<sup>(1)</sup>, Christian Uebing<sup>(3)</sup>,  
Wolfgang Walkowiak<sup>(1)</sup>, and Roland Wismüller<sup>(2)</sup>

<sup>(1)</sup> Experimental Particle Physics, University of Siegen, Germany

<sup>(2)</sup> Operating Systems and Distributed Systems, University of Siegen, Germany

<sup>(3)</sup> Center for Information and Media Technology, University of Siegen, Germany

## Abstract

Online steering and visualization of scientific applications is a well-established method for accelerating research and saving resources. However, for Grid environments no appropriate, secure online steering tools exist. As a part of the HEPCG (High Energy Physics Community Grid) project we are developing the online steering system RMOST for Grid applications, which is specifically targeted towards HEP applications used for the ATLAS experiment at the LHC. RMOST is fully integrated into the ATLAS software, i.e., the Athena framework and the ROOT toolkit used for visualization; users just have to extend the job description files. The steering system is based on a layered architecture with well defined, application independent interfaces. In this paper, we will focus on the communication layer, which implements solutions for the two most important requirements in a Grid environment: (1) a user-friendly way of finding the running job, and (2) a mechanism to enable a secure communication, even across firewalls or private IP networks.

## 1 Introduction

Although Grid computing offers access to a huge collection of data and computing resources, efficient science is still hindered by the delays between submitting a Grid job and receiving its first feedback: Today, after submitting a long-running Grid job, a researcher has to wait until the job has finished before he can retrieve any output. Only then, he or she can evaluate the results and see if the results are useful. Some job parameters could have been set incorrectly or still need to be optimized to get significant results. In these cases the job must be re-submitted, again with a possibly long waiting time.

Online steering, i.e., the monitoring of intermediate results combined with an interactive control over the running job, has proved to be an effective method to circumvent these problems, thus accelerating computational scientific research. This is even more true in a Grid environment, given its long submission delays and the insufficient accessibility of running jobs. If there would be any possibility to steer running Grid jobs, the execution time could be reduced significantly.

---

\* This work is partly funded by the *Bundesministerium für Bildung und Forschung* (BMBF) as part of the German e-Science Initiative (Contract 01AK802E, HEP-CG).

Thus, as part of the HEP Community Project of the German eScience initiative, the online steering tool RMOST (Result Monitoring and Online Steering Tool) is being developed, which is specially targeted towards the needs of High Energy Physics (HEP) applications. In particular, the software used for ATLAS is supported, one of several experiments at by the new particle accelerator LHC (Large Hadron Collider) at the European particle physics laboratory CERN, which will start operating in 2007. Like all the other experiments, this experiment produces huge amounts of data that have to be evaluated. Since it is infeasible to do all the processing on a central computing resource, scientists will have to use the LHC computing Grid (LCG). RMOST will provide online access to their jobs.

When designing such a tool, two major challenges have to be addressed: First, the integration of the steering functionality into the existing, software frameworks should not require any source code changes. Thus, the basic functionalities of RMOST, like steering the job execution and monitoring the most common intermediate results, can be applied to a Grid job by just adapting the job description. Only advanced functionality, like steering arbitrary user defined data, requires the user to modify his source code. Second, a secure communication link to the Grid job has to be established. E.g., there may be a firewall blocking any direct connection to the job's worker node, or the worker node may be part of a private network, not connected to the Internet. The communication layer of RMOST has been designed to deal with all these cases.

The remainder of the paper is organized as follows: First, we survey the related work in Section 2. In the main part, Section 3, we present the architecture of our steering framework, focusing on two issues: the integration into the existing software framework of the ATLAS experiment, and the establishment of a secure communication link between the steering tool and the Grid job. Finally, in Section 4 we present the status of the project and future work.

## 2 Related work

There are three different kinds of tools, which are related to the work presented here. First, there are steering tools for the Grid: at the moment, besides HEP-CG, just the RealityGrid project is working in this area. However, a couple of other Grid projects (CrossGrid, I-GASP) are dealing with interactive access to Grid jobs, but with different goals. Finally, there are online steering tools for non-Grid environments like CUMULVS and SCIRun.

In the **RealityGrid** project [1, 2], a steering library for Grid applications was developed. To use this steering library, the source code of the application has to be instrumented with calls to the steering API. Basically the steering library notifies the application upon every steering event and the application has to react on that notification. The communication either uses TCP sockets, pipes, or SOAP over HTTP, without additional security mechanisms.

As part of the **CrossGrid** project [3] the Migrating Desktop and Roaming Access Server (MD/RAS) was developed. It enables the user to work interactively with a Grid job, but it is not a steering tool, since it just provides a

communication link across the Grid. To establish the connection, a shadow process for the job is started during job submission. The job connects to this job shadow and the user can connect to the shadow as well. This always requires outbound connectivity of the worker nodes, and an open port range on the host where the job shadow is running. The CrossGrid project also developed another tool called glogin [4, 5], which allows an interactive login to Grid sites (i.e., the compute element) and also supports the forwarding of TCP connections.

The Interactive Grid architecture for Application Service Provider (**I-GASP**) [6] developed by HP, provides display forwarding for interactive applications. The architecture is designed for company Grids. To establish a communication channel, both peers connect to a well known communication server, which forwards all traffic.

**CUMULVS** [7, 8] is a software system for collaborative online steering of parallel simulations in distributed environments. The source code of the application needs to be instrumented with CUMULVS library calls. At a location in the application, where the data is consistent, a special call to a CUMULVS function is required in order to respond to steering commands.

**SCIRun** [9, 10] is a problem solving environment where the user composes his job from different components in a graphical way. In the SCIRun GUI, the user can steer and monitor the execution of the different components. The use of SCIRun means, however, that the whole application must be developed for the SCIRun environment. The adaption of existing software to SCIRun requires some effort.

Compared to the above tools the distinguishing feature of RMOST is its simultaneous support for application steering, the Grid, and enhanced security.

### 3 Architecture of RMOST

The steering system is based on a layered architecture with well defined, application independent interfaces between each layer, consisting of a communication layer, a data consistency layer, a data preparation layer and the application layer (c.f. Fig. 1). The communication layer establishes a secure, interactive connection between the visualization tool and the remote Grid job. This layer will be presented in more detail in Sect. 3.2. The data consistency layer handles the higher-level data exchange between the job and the visualization tool. The data preparation layer is provided to support automated pre-evaluation of the job's data, either in the steering tool or in the job itself. It is also planned to include automated checking of intermediate results and error reporting. Finally, the application layer provides all application specific functionality, esp. data acquisition and visualization.

#### 3.1 Application Layer: Integration into ATLAS Software

A major goal of the application layer is to facilitate integration of the online steering tool into the existing ATLAS software framework, consisting of the

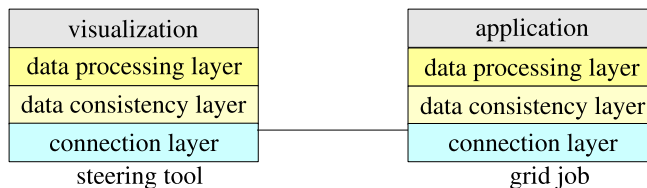


Fig. 1: Layered overall architecture of RMOST

experiment software framework Athena [11] and the ROOT toolkit [12] used for visualization.

In the ATLAS experiment, compute jobs typically evaluate large numbers of detector events, which are independent of each other. For each event a processing chain is executed, consisting of several different algorithms called in sequence, each one possibly working on the output of previous ones. The Athena framework [11] supports this kind of evaluation by providing a collection of suitable algorithms (e.g., for detector modeling) together with proper data objects and a set of services for common tasks, like storing results, loading data or printing messages. The user composes his jobs from the provided components (algorithms and services) in a job description file (called job options) without programming any source code in addition. Algorithms and services also can have adaptable properties, which are configured in the job options file, too. The Athena job then executes the specified list of algorithms for each event in a loop, c.f. Fig. 2.

In addition, it is possible to create own components and use them in the Athena framework. For this purpose, the user creates a shared library, which then can be used in the job options file. This mechanism allows to smoothly integrate online steering into the Athena framework. The basic functionality is provided by an additional algorithm called `RM_Spy`, which allows

- to monitor most of the intermediate results, stored in ROOT files,
- to steer the job execution, e.g., to terminate, suspend and resume a job, to switch to single step (event-by-event) execution, or to restart the job without resubmission, and
- to change the job options file and to apply the changes after a restart without resubmission of the job.

To enable this functionality, the user just needs to change the job options file in order to insert the `RM_Spy` algorithm; no changes of the source code are necessary.

More advanced functionality is available via a new service, called `RM_SteeringSvc`, which allows monitoring and steering of arbitrary user-defined data. However, to use this service, source code instrumentation is required in order to register the data with the `RM_SteeringSvc` service.

The results of Athena jobs are usually visualized interactively, using the ROOT toolkit [12]. ROOT contains a command line interpreter for C++ statements and a class library with visualization components. It provides an interface to dynamically load new classes from the command line, using shared libraries.

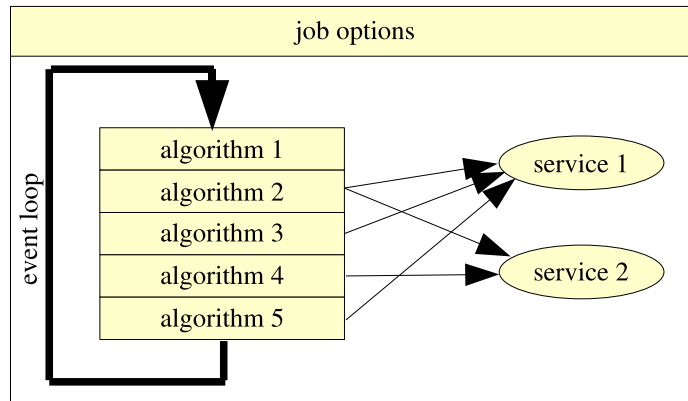


Fig. 2: Simplified structure of a job in the Athena framework

In order to integrate RMOST with ROOT, a class library is provided which makes the job's intermediate results available within ROOT and also provides a simple GUI for the steering commands.

### 3.2 Connection Layer: Establishing a Grid Connection

The core of every Grid steering tool is a communication structure to exchange data between the local tool and the remote Grid job. The gLite middleware<sup>1</sup> used by the LHC computing Grid does not provide a suitable mechanism to establish an interactive connection to a Grid job. Thus, a connection layer was developed, which is presented in the following paragraphs.

**Requirements to the connection layer.** The connection layer has to address two substantial challenges: The first one is being able to create a connection at all, the second one is to make this connection secure.

Creating a connection is first hampered by the fact that, in general, the user does not know on which particular host (worker node, WN) his job is running. The only reference to the job the user has, is the job identifier obtained during job submission. Therefore, the connection layer connects to a job via this job identifier and hides the complexity of the real network addresses and how to find them from the user. The steering tool should connect to the job, rather than vice versa, since the user should be able to run his job without the steering tool and connect only later, if he suspects a problem. This is a major difference to other approaches implementing interactivity for Grid jobs.

Even if the address of the WN and a proper port are known, it may not be possible to directly connect to it. E.g., there might be a firewall, which blocks incoming connections. Or, the WNs may be part of a cluster using a private IP

<sup>1</sup> see <http://www.glite.org>

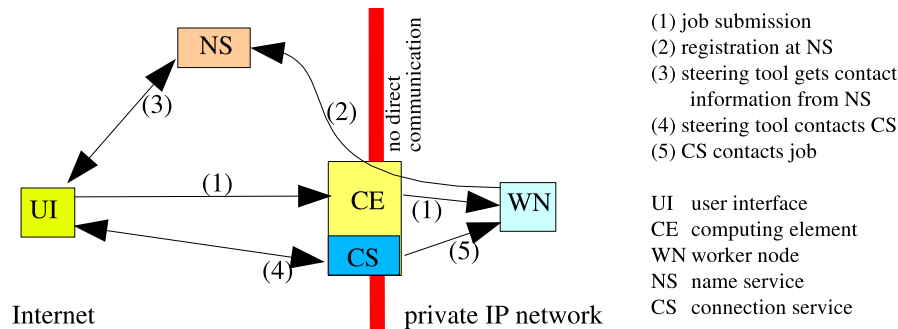


Fig. 3: The process of connecting to a grid job on a WN in a private IP network

network, which is accessible only via its front end (computing element, CE). The connection layer deals with these situations in a user-friendly and transparent way. Since the Grid is a heterogenous and dynamic environment, where each site might have a different configuration, the connection strategy is not known at job submission time, but it is automatically determined at run time.

One of the most critical issues to both users and administrators is the security of the connection. The user does not want that anybody else can steer his job. The administrator is concerned about the resources, he is responsible for. He does not want that unauthorized persons can get access to his resources, or that people having a valid Grid account can gain additional permissions that may harm the resources. Thus, the connection layer implements authentication and authorization. In addition, if the communication needs to bypass a firewall, it does so without breaking the site's security policies.

**The connection process.** When designing the connection layer, several scenarios that can occur have been defined and analyzed. They differ in the configuration of firewalls, the existence of private IP networks, and the location of the connection service, which is an additional component of RMOST aiding the establishment of the connection. In the following, just one of these scenarios will be discussed as an example. In this scenario, the WN is located in a private IP network, which is not directly accessible from the Internet (see Fig. 3). In this case it is required that the connection service (CS) is running on the CE. (It may be started by the user as a new Grid job, just after he submitted his regular job.) Once the job has been submitted (1), it first registers its contact information (address/port of CS and job) to a naming service (NS)(2). Currently the R-GMA monitoring infrastructure [13] is used for this purpose, which includes its own mechanisms for forwarding the information. Now, when the user starts the steering tool on the user interface machine (UI), it requests the job's contact information from the NS (3). Next, it connects to the CS (4) and the CS in turn connects to the job on the WN (5). Both connections are secured via the Generic Security Service (GSS) API from Globus [14]. In addition, an

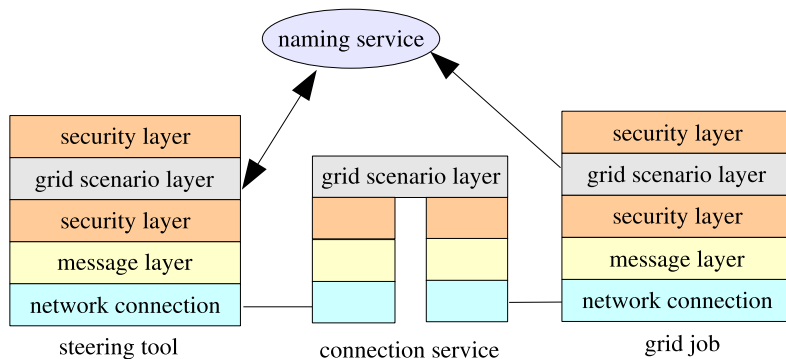


Fig. 4: Architecture of the connection layer

end-to-end authentication is performed, so the user doesn't have to trust the CS.

**The architecture.** The communication layer is again divided into several sub-layers (see Fig. 4). The bottom layer implements an asynchronous TCP connection. From the second layer on, message-based communication is used. The third layer is a security layer, which secures the connections between the steering tool and the CS, as well as between the CS and the job. Its major goal is to ensure that the CS can not be misused. Due to this layer only persons with a valid grid account at the site are allowed to connect to the CS, all others are rejected. Furthermore, it only allows to contact ports at the site which are secured by the GSS-API as well. The fourth layer is the grid connection layer, which deals with finding the Grid job and the different site configurations and scenarios. The top layer again is a security layer, which performs the end-to-end user authentication.

## 4 Conclusion and outlook

The online steering tool RMOST for the High Energy Physics experiment ATLAS has been developed, which is integrated into the existing ATLAS software and can be applied easily. A communication layer for Grid environments has been developed, which allows secure, interactive connections to running Grid jobs. The implementation of the first prototype of RMOST, which uses this communication layer, has recently been finished. It has been released, including full documentation, in November 2006.<sup>2</sup>

In the future RMOST will be extended with features for automated error detection and reporting, as well as monitoring and steering of parallel Grid jobs.

<sup>2</sup> see <http://www.hep.physik.uni-siegen.de/grid/rmost/index.html>

Another interesting possibility is the use of the communication infrastructure presented here for connections between Grid jobs at different sites.

## References

1. S. Jha, S. Pickles, and A. Porter. A Computational Steering API for Scientific Grid Applications: Design, Implementation and Lessons. In *Workshop on Grid Application Programming Interfaces*, Brussels, Belgium, Sept. 2004.  
<http://www.cs.vu.nl/ggf/apps-rg/meetings/ggf12/pickles-final.pdf>
2. S. M. Pickles, R. Haines, R. L. Pinning, and A. R. Porter. Practical Tools for Computational Steering. In *Proc. UK e-Science All Hands Meeting*, Sept. 2004.  
<http://www.allhands.org.uk/proceedings/papers/201.pdf>
3. M. Plociennik, B. Palak, and R. Pajak. *Developer Manual: Migrating Desktop and Roaming Access Server*. [http://www.eu-crossgrid.org/developer\\_manuals/CG3.1-v1.0-PSNC-MD\\_RASDeveloperManual.pdf](http://www.eu-crossgrid.org/developer_manuals/CG3.1-v1.0-PSNC-MD_RASDeveloperManual.pdf)
4. H. Rosmanith and D. Kranzlmüller. glogin - A Multifunctional, Interactive Tunnel into the Grid. In *5th IEEE/ACM Intl. Workshop on Grid Computing*, Pittsburgh, USA, Nov. 2004.
5. H. Rosmanith and J. Volkert. Traffic forwarding with GSH/GLOGIN. In *Proc. of EUROMICRO-PDP*, Lugano, Switzerland, Feb. 2005.
6. S. Basu, V. Talwar, B. Agarwalla, and R. Kumar. Interactive Grid Architecture for Application Service Providers. In *Intl. Conf. on Web Services (ICWS'03)*, Las Vegas, 2003.
7. J. A. Kohl and P. M. Papadopoulos. Efficient and Flexible Fault Tolerance and Migration of Scientific Simulations Using CUMULVS. In *2nd SIGMETRICS Symposium on Parallel and Distributed Tools*, Welches, OR, Aug. 1998.
8. G. A. Geist, J. A. Kohl, and P. M. Papadopoulos. CUMULVS: Providing Fault-Tolerance, Visualization and Steering of Parallel Applications. *Intl. Journal of High Performance Computing Applications*, 11(3):224-236, Aug. 1997.
9. S. G. Parker, M. Miller, C. D. Hansen, C. R. Johnson, and P.-P. Sloan. An Integrated Problem Solving Environment: The SCIRun Computational Steering System. In *31st Hawaii Intl. Conf. on System Sciences (HICSS-31)*, Vol. VII, pages 147-156, Kona, Hawaii, Jan. 1998.
10. D.M. Weinstein, S.G. Parker, J. Simpson, K. Zimmerman, and G. Jones. Visualization in the SCIRun Problem-Solving Environment. In C.D. Hansen and C.R. Johnson, editors, *The Visualization Handbook*, pages 615-632, Elsevier, 2005.
11. European Laboratory for Particle Physics. *Athena Developer Guide*  
<http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/00/architecture/GeneralDocumentation/AthenaDeveloperGuide-8.0.0-draft.pdf>
12. R. Brun, F. Rademakers, P. Canal, I. Antcheva, and D. Buskulic. *ROOT user Guide* [ftp://root.cern.ch/root/doc/Users\\_Guide\\_5.12.pdf](ftp://root.cern.ch/root/doc/Users_Guide_5.12.pdf)
13. A. Cooke et al. R-GMA: An Information Integration System for Grid Monitoring. In *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, pages 462-481, LNCS 2888, 2003. Springer Verlag.
14. I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. A security architecture for computational grids. In *Proc. 5th ACM Conf. on Computer and Communications Security*, pages 83-92, San Francisco, 1998. ACM.