CACR Technical Report

CACR-185 **March 2000**

Grid Computing on the Web Using the Globus Toolkit

Giovanni Aloisio, Massimo Cafaro, Paolo Falabella, Carl Kesselman,
Roy Williams

Mailing Address: CACR Technical Publications, California Institute of Technology,
Mail Code 158-79, Pasadena, CA 91125. Phone: (626) 395-6953 Fax: (626) 584-5917

Grid Computing on the Web Using the Globus Toolkit

Giovanni Aloisio[1], Massimo Cafaro[1], Paolo Falabella[1],
Carl Kesselman[2], and Roy Williams[3]

[1] Dept. of Innovation Engineering,
University of Lecce, Italy
{giovanni.aloisio, massimo.cafaro, paolo.falabella}@unile.it
[2] Institute of Science Information,
University of Southern California
carl@isi.edu
[3] Center for Advanced Computing Research,
California Institute of Technology
roy@caltech.edu

Abstract. In this paper we present and discuss an architecture that allows transparent access to remote supercomputing facilities from a web gateway. The implementation exploits the Globus toolkit and provide users with fast, secure and reliable access to parallel applications. We show the usefulness of our approach in the context of Digital Puglia, an active digital library of remote sensing digital data.

1 Introduction

This paper describes a design pattern that enables a user to run applications on a Computational Grid [1] using a standard Java-enabled browser. We define a Grid application as consisting of one or more programs, located on geographically distributed machines; these programs access data stored on distributed databases. Among the most important requirements is the capability to find the right programs and to run them with the right data by means of high level requests. The user should be also allowed to check the status of the remote execution, get the output of the application and, in case of an interactive application, steer it from the browser.

In order to integrate Grid applications and the Web, we advise the use of a three tier architecture: (1) the client web browser, (2) middleware running on a secure webserver, (3) the supercomputers of the Computational Grid.

The issues related to Computational Grids and metacomputing have been abundantly described in literature and, in our context, we address essentially the same problems: how do we locate resources, how do we secure these resources and the transactions involving them, how do we start executing an application on a remote machine and so forth. Instruments to help solving these problems have been developed in recent years [2]; among them we chose to use the Globus toolkit which by this time has been released in its version 1.1. The rest of the paper is organized as follows. Section 2 presents the issues related to Grid applications; we discuss them in sections

The work presented in this paper was supported in part by ISUFI (Istituto Superiore Universitario di Formazione Interdisciplinare) and by CACR (Center for Advanced Computing Research

3 – 7 describing the solutions provided by our framework. Section 8 shows how we exploit the framework in the context of Digital Puglia, an active digital library of remote sensing data and section 9 concludes this work.

2 Web – Grid Integration

In this section we discuss the issues arising from the integration of Computational Grid applications with the Web; in particular, here we address the following:

1. Security: The Globus tools use X509v3 certificates to mutually authenticate a user and the Grid machines. We propose a two-step mechanism to enable user – Grid authentication from a web browser: a login and password from the browser to the web-server and a certificate from there to the Grid.
2. Resource management: An application running on the Grid could exploit distributed, heterogeneous resources. If we want our application to find the resources dynamically we must store somewhere the information needed to locate the machines, files, applications and so on. This is exactly the aim of the Grid Information Service, an LDAP [3] Directory Service provided by the Globus organization. Applications may need a broker to translate high level requests made by the users in low level system requests.
3. Remote start and status: Commands are provided in the Globus toolkit to run interactive applications or to submit batch jobs on a remote machine taking care of the authentication issues and to check the status of the execution. These commands can be included in CGI scripts or java servlets to be run on a web server hosted on the second tier of our proposed architecture.
4. Steering: A Java applet can be used to provide the user with a user friendly interface to the Grid application.
5. Result retrieval: The final output produced by the application is held on a remote machine or on a collection of remote machines. The choice of an appropriate strategy is needed to let the user have the results on her web browser in the minimum time.

In order to implement the approach proposed in this paper, the Grid has to be set up in the following way:

- The first tier needs just a Java-enabled browser: this is the basic functionality that all workstations, laptops, and operating systems can provide,
- Globus has to be installed on the machines on the second tier and on the third tier
- A web server must be installed on the machine on the second tier: we strongly suggest it to be a *secure* web server

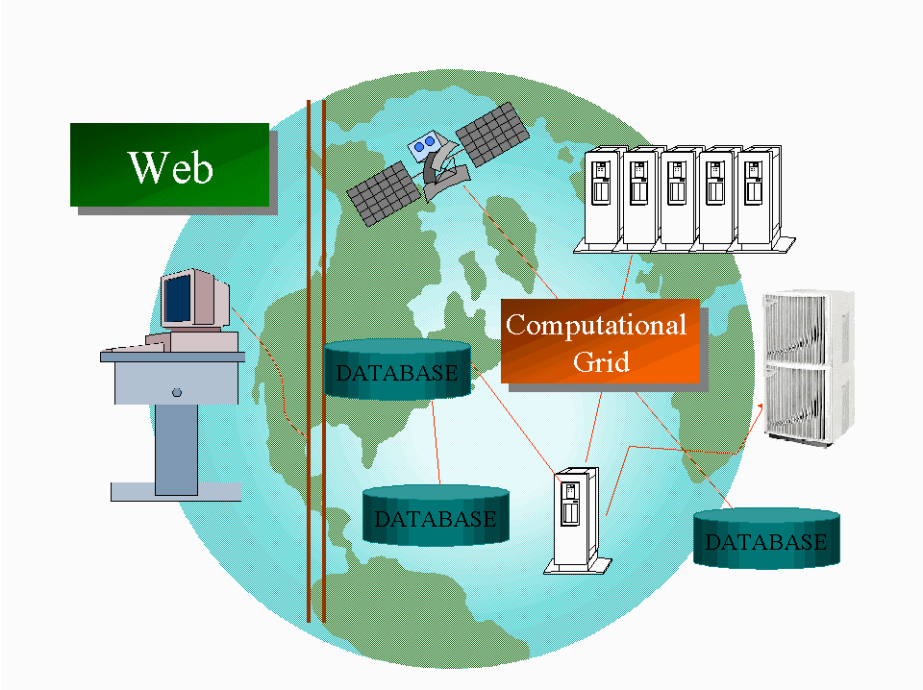

Fig. 1. Accessing the power of a Computational Grid via Web

3 Security

A Computational Grid consists of a set of valuable resources like parallel supercomputers, workstations, databases and smart instruments, so in order to prevent unauthorized accesses to the Grid we need a strong authentication mechanism.

We propose a two step mechanism: the user must authenticate herself from the client browser to the second tier and also from the second tier to the Grid. A list of trusted users is held on the second tier, and authentication via web is done through a login and password scheme. To guarantee the safety of the password we strongly suggest the use of a secure web server to avoid transmitting the password in clear over an insecure channel.

The second step of the authentication may depend on different security policies. We can decide to map all the users on the same Globus certificate (the “guest” account) or have each trusted user store her individual certificate and private key on the second tier. If we use just one certificate and private key for all the users, it is simpler because we shall not need to create an account for all the users on every machine of the Grid and to store every user’s certificate and private key on the second tier. This approach is much easier to manage and it scales better with an increasing number of machines on the third tier; however it is risky to have many people using a common “guest” password. On the other hand, we might need to use the more complex one to one mapping if the application needs to know which user is making a certain request: for instance in case of personalized options, of different privileges, or to capture accurate accounting information from the supercomputing resource.

Before the machine on the second tier can run any Globus command on the Grid, a Globus proxy must be created with the "grid-proxy-init" command. If the mapping is one to one, the password the user has to enter to access the system must be the PEM pass phrase that protects her private key.

4 Resource Management

Applications running in a Grid environment should be resource aware and adaptive, i.e., capable of handling heterogeneity, and able to adjust their behavior dynamically in response to changes in the Grid. When the user formulates a high-level query the application should be capable of runtime discovery of computing resources (that can be geographically distributed), resource reservation through allocation and/or co-allocation and remote execution on the available pool of machines; one or more executables may be involved in the computation, and a dataset collected from a distributed database. To achieve this goal there should be:

1. A way to store and retrieve information about machines, executables and data; the Globus organization maintains the Grid Information Service, formerly known as Meta Directory Service, which is an LDAP based directory service used to provide uniform access to structure and state information about entities composing the Grid.
2. One or more *Resource Brokers* in charge of processing high-level requests formulated by the users and translating them in low level requests understandable to the Grid, if necessary exploiting the information available in the Grid Information Service.
3. Grid software infrastructure tools like the Globus toolkit middleware that provides a bag of core services, including a low level scheduler API, the Grid Information Service, multimethod communication and QoS management, single sign-on and key management, remote file access and Grid status monitoring.

5 Remote start and status of the execution

Once the machine on the second tier has located the executables, we can use the Globus commands or API to run it remotely. A java servlet or CGI script can be easily written to incorporate the *globusrun* (or *globus-job-submit*) command.

The essential parts of this CGI script are shown below (PERL language is used in this example):

```
# environment variables needed by Globus
$ENV{HOME} = "/users/myapplication";
$ENV{X509_CERT_DIR} = \
"/usr/local/globus/share/certificates/";

# creation of a grid proxy: if we are using a one to
# one mapping, certname and keyname will point to the
# certificate and private key we are using for
# everybody. The pwstdin option allows us to read the
# PEM pass phrase from the stdin and thus to use
# redirection
```

```
system(grid-proxy-init -pwstdin -cert $certname -key \
$keyname < $PEM_pass_phrase);

# execution of the remote program
system(globus-job-submit $remote_machine $executable);
```

The status of execution can be checked with the utility *globus-job-status*, using the job-id returned by *globus-job-submit* and which uniquely identifies the submitted job.

The application may require command-line arguments and/or input files. An applet or web form can assist the users during the phase of input generation providing a friendly GUI that can be used to derive visually input parameters; a CGI or a servlet on the second tier will then process the request generating command line arguments, files or both.

The *globus-job-submit* and *globusrun* commands both allow passing command-line arguments to the application, so it is quite straightforward to add them to the sample code snippet presented above.

The *globus-rcp* command mimics on the Grid the functionalities provided by the standard *cp* and *rcp* Unix commands and can thus be used to transfer input files on the machines belonging to the third tier before starting the remote execution.

6 Steering

Not all of the Grid applications will be batch jobs that can be submitted to a pool of Grid machines, so that steering of an interactive application is an important concern here.

Let us assume that we have to develop an interactive application. We advise writing a multithreaded application in which one of the threads will be responsible for handling the flow of control instructions via sockets, and the use of java applets to provide the users with a friendly GUI that can be used to steer the application at runtime.

The interaction involves bi-directional network communication over sockets, one channel is used to transmit steering instructions, the other one to report diagnostic output about commands executed.

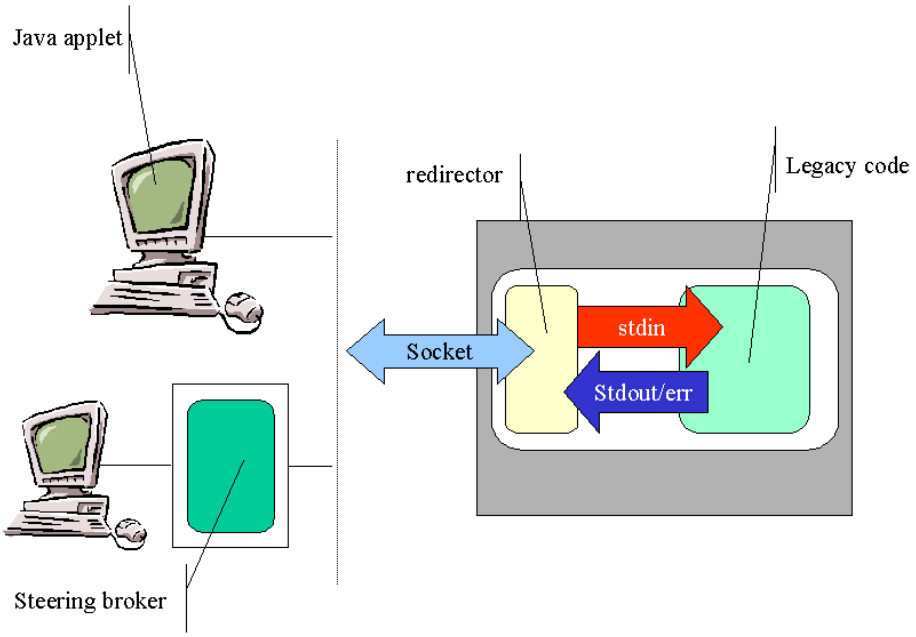

Fig. 2. The use of a *redirector* allows the steering of interactive application without having to change the code. The user interface is in an applet. We might have to use a *steering broker* to connect the applet to the application if there is no possibility to install a web server on the back-end machine.

The next steps in the design depend on the possibility to install a web server on the machines belonging to the third tier. This in turn depends on the local policies in use at the different sites composing the Grid. As an example, it may not be feasible to install a web server on a supercomputer because it can compromise the security of the system. Anyway, if system managers allow it, the easiest thing to do is to have the applet come directly form the third tier to the browser.

Otherwise, the applet must be installed on the middle tier web server. Since unsigned applets can't communicate over a network connection with arbitrary machines, we can address the problem either exploiting signed applets or by means of a *steering broker* whose aim is to dispatch to Grid machines the commands forwarded to it from the applet and to the applet the diagnostic stream output generated by the remote machines. The process of applet signing is not straightforward and differs considerably if the applet is to be signed for use with different client browsers, so we feel confident that the steering broker approach is to be preferred.

Communication between the steering broker and the machines on the Grid may use XTI, TLI or the Nexus communication library instead of sockets while communication with the applet is restricted to sockets due to java limits.

Now, let us suppose that we have at our disposal an interactive legacy application we want to make available on the Web using the steering mechanism described above. In addition to the steering broker we need to add a *redirector*, that is, a small piece of code that runs on the same machine as our legacy application with the purpose of redirecting the standard input, output and error of the application to sockets connected to the steering broker connected to the steering broker (fig 2).

7 Result retrieval

The final output produced by the application is held on a remote machine or on a collection of remote machines. Two strategies are possible here. We can choose either to retrieve the result directly from the machines of the third tier or to gather the results on the middle tier web server. It is difficult to predict which option will give the best results in terms of faster file transfers lacking dynamic information about the speed of network connections and the available bandwidth that may be reduced due to heavy traffic. We advise the use of a dynamic reconfiguration procedure to choose the route that minimizes the time needed by file transfers.

If accelerating data output movement is not a critical issue, the designers may decide to trade the benefits arising from a dynamic reconfiguration with the ease of a static design, in which one of the available options is simply chosen randomly.

If the output has to be retrieved directly from the third tier, Globus provides a useful tool, i.e., the *globus-gass-server*, that can be exploited to transfer files using the HTTP protocol directly from the client browser. Otherwise, if the output has to be migrated on the second tier, we suggest the use of a manager – worker scheme, designating the machine on the middle tier as the manager and the others belonging to the Grid as the workers.

The manager machine is in charge of gathering the output generated by the application, so that the user will retrieve all the files from the manager machine; the task can be carried out using the globus-rcp command to put the output files on a directory accessible from the web.

8 Web access to the Grid in Digital Puglia

Digital Puglia [4-7] is an active digital library of remote sensing data which allows interactive browsing and parallel post processing on the Grid using the Web as a gateway. A set of trusted user was defined and we provided them with a couple of parallel applications, a supervised bayesan classifier and a Principal Component Analysis.

Since our users are remote sensing experts, not computer scientists, we decided to map all of them to a single Globus certificate. The users access Grid applications from the University of Lecce Digital Puglia web site, filling in an authentication form. Then, an applet allows them to interactively derive the input parameters, as needed by the applications. A couple of CGI scripts start and monitor execution progress on the remote machine, an HP Exemplar machine at Caltech.

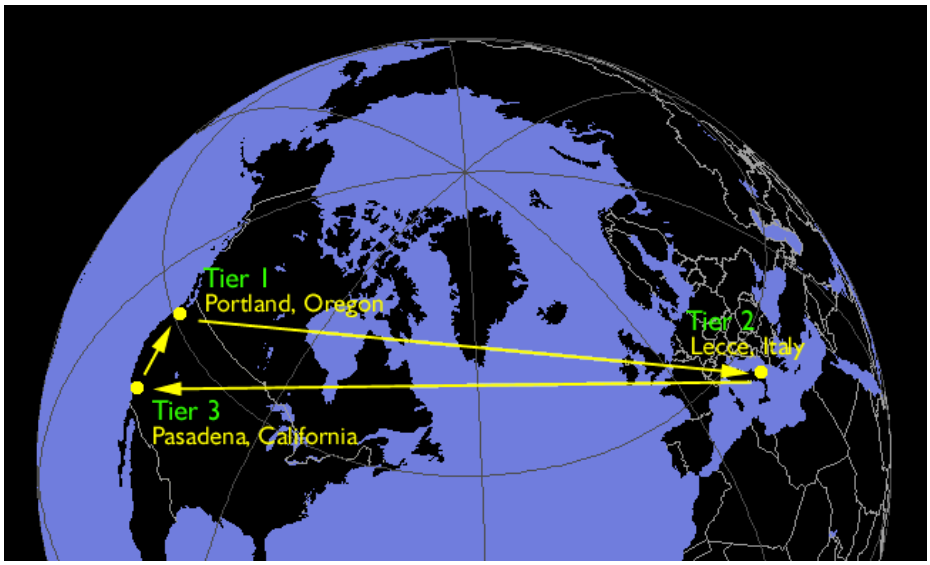

Fig. 3. In the Digital Puglia active library, presented at SuperComputing'99 the possibility to access a worldwide Computational Grid via a web browser was demonstrated

The final output, which is an image, is then retrieved directly from the browser where it can be visualized and saved.

An example session with the supervised classifier was shown in the NPACI exhibit at SuperComputing 1999 to demonstrate the feasibility of interactive web access to Grid applications.

9 Conclusions

We have presented and discussed a framework to build Grid applications that can be transparently started and steered from a Java-enabled web browser. The architecture is based on the Globus toolkit to provide users with fast, secure and reliable access to applications on the Computational Grid. We have showed an example of the use of this framework in the context of Digital Puglia, an active digital library of remote sensing digital data.

References

1. Ian Foster, Carl Kesselman: The Grid. Blueprint for a new computing infrastructure. Morgan Kaufmann, San Francisco (1999)
2. http://www.gridforum.org
3. LDAP v3 Protocol (RFC 2251)
4. R. D. Williams, G. Aloisio, M. Cafaro, P. Messina, J. Patton, "SARA: The Synthetic Aperture Radar Atlas", http://www.cacr.caltech.edu/sara/
5. G. Aloisio, M. Cafaro, P. Messina, R. Williams, "A Distributed Web-Based Metacomputing Environment", Proc. HPCN Europe 1997, Vienna, Austria, Lecture Notes In Computer Science, Springer-Verlag, n.1225 (1997) 480-486.

6. G. Aloisio, G. Milillo, R.D. Williams, ”An XML Architecture for High Performance Web-Based Analysis of Remote Sensing Archives”, Future Generation Comp. Sys., 16 (1999) 91-100.
7. G. Aloisio, M. Cafaro, R. Williams, “ The Digital Puglia Project: an active digital library of remote sensing data”, Lect. Notes in Comp.Sci. (Springer) 1593 (1999) 563-572