

ULabGrid, an infrastructure to develop distant laboratories for undergrad students over a Grid

O. Ardaiz, D. Royo, P. Artigas, L. Daz de Cerio, F. Freitag,
S. Sanjevan, R. Messeguer, and L. Navarro

Computer Architecture Department
Polytechnic University of Catalonia, Spain
{oardaiz, dolors, partigas, ldiaz, felix, sanji, messeguer,
leandro}@ac.upc.es

Abstract. Nowadays, there is a big discussion about two different topics: how distance learning and the old fashioned learning can be improved using the new technologies. In both cases, there are many collaborative tools based on the web infrastructure such as e-mail, web discussing groups, virtual campuses or audio and video conferences, that basically give a way of exchanging information among the different groups involved in learning tasks, but very few of them have been thought to help or to develop laboratory classes (labs). In this paper we describe a GRID infrastructure (ULabGrid) that supports distant laboratories for undergrad students.

1 Introduction

Many of the systems for distance learning proposed in the literature that target individual tools tend to be not reusable (ability to reuse the computing system with other applications without making any modifications to the system itself) in spite of the fact they involve a significant amount of duplicated effort. For example, a large number of systems (e.g., the Exploratium [1], JSPICE [2]) are based on scripts that need to be modified in order to add new applications to the system.

Other designs are more flexible, for example, the MOL [3] prototype, employs static web interfaces that can be adapted for individual tools. However those static interfaces are not adequate for any tool.

Solutions that address individual issues are generally reusable, but the tasks of adapting them for a production environment and integrating them into a complete computing infrastructure are non-trivial (VNC [5]).

As far as we know, PUNCH (Purdue University Network Computing Hubs) is the first and, to date, the only web-based computing system that is designed to support arbitrary tools and is utilized on a regular basis in a live environment [6, 8, 7], but PUNCH system is not built over a GRID.

In this project we propose to design and implement an infrastructure, ULabGrid, to develop educational distant laboratories over a Grid [4]. The users we target are undergraduated students. The system will allow users to run the

tools needed to develop the labs through the net and will provide operating system services for networked resources (provides user-transparent file, process, and resource management functions, handles security and access control across multiple administrative domains, and manages state information (session management). Users will be capable to run tools remotely from everywhere at any time using their own computers.

2 Advantages of using ULabGrid infrastructure

In the following, we highlight the more common problems emerging when using educational software in a lab class at the present time and how can be solved using the infrastructure we want implement.

Software installing and maintenance. Installation of software tools on different systems is not an easy task. The software must be downloaded and installed. The tool installation often requires porting and debugging efforts. There are different options for different platforms or even the software not runs on the available architecture or operating system version.

Under ULabGrid infrastructure, software will be installed and maintained by an expert (system administrator) so the students will not have to spend time on other issues than running the tools.

Licensing. Often the software used in educational labs is not free distributed. The educational center have a limited number of licenses but students interested in working at home are not willing to buy his own license due to the expensive price. The solution adopted by most of the students is getting an unlicensed copy.

ULabGrid will allow to access the software installed in the educational center through a web browser. The students will execute programs in a remote way and the system will control the number of users do not exceed the number of available licenses. The program will never run on the student computers, so the students will not incur in an illegal action.

Lack of hardware resources. When decisions must be taken about which software tools are being installed in the labs, a very important factor to be in consideration is that the students have the chance of installing that software on their own machines (usually PC's). Because of this, many times tools only running in powerful or very specific machines are discarded.

Due to the fact that ULabGrid infrastructure will allow to access tools installed in remote machines, there will be a wider range of options in the tool selection process.

High cost in changing the lab contents (hardware and software). Updating the educational software in the lab (upgrading versions, new applications) involves a high cost for the student. New versions are more expensive than older ones and most of the times require more computational resources.

Through ULabGrid system this problem will not have sense because students will not have to acquire nor install the tools on their own machines.

Non homogeneous results. Another problem is the fact that the results of the exercises developed by the students depend on the environment where the software is executed. If the software is run on their own machines the results could not be equivalent.

ULabGrid will guarantee a unique environment and therefore the same results for all the students. It will make easier the detection of errors by both the student during the practical exercise and the teacher during the evaluating process.

Intricate interfaces. Often, using or even understanding the interfaces between users and software applications is not an easy task. They require the knowledge of specific parameters, configuration files, flags and other stuff. The students must spend significant amount of time in learning "how to use" instead of learning "using" the required tools.

The ULabGrid system will let the administrators develop smooth web interfaces to access and to introduce the required parameters before using the tools.

Other inherent advantages to distance learning offered by ULabGrid will be: a) students will be able to familiarize with the execution tool environment before starting their practices, consulting the manuals or the examples provided by the system, and b) students will be able to do their exercises at home without physical presence at the educational center.

3 Project Status

ULabGrid infrastructure benefits from the functionalities provided by Globus toolkit [9–11] (www.globus.org): the Resource Allocation Manager (GRAM) and Monitoring and Discovery Service (MDS) to support wide area execution, estimated execution times and queue delays provided by GRAM and MDS will be used to schedule lab exercises, GridFTP to manage data among machines, and the Grid Security Infrastructure (GSI) to ensure a secure access to the resources.

ULabGrid infrastructure is being developed using Globus Portal Development Kit (GPDK, [http:// doesciencegrid.org/projects/GPDK](http://doesciencegrid.org/projects/GPDK)).

Initially, we are designing and implementing two main components:

1. Session Scheduler: a web portal is being designed and implemented which will allow to start simulation processes in remote machines, monitors the execution and give the results back to the students.
2. Resource Manager: one of the important issues to address by networked-computing systems (like ULabGrid) is the efficient management of networked resources (software, machines, network...). The resource management system must provide an efficient way to organize all networked resources and will be responsible for choosing the best available resource, depending on static parameters (user requirements) and dynamic parameters (network load, machine load, number of users connected. . .) [12, 13].

Both components are being designed to support graphical interactive tools (like NS-NAM network simulator). The nature of this type of tools (interactivity, quick response time, graphical interfaces. . .) directs the design of both

components. From the point of view of the Resource Manager, we need to know constantly the state of the resources (user load, machine load. . .) to decide which resource (machine) is the best to run a new simulation. From the point of view of the Session Scheduler we need a remote display system which allows you to view a computing 'desktop' environment not only on the machine where it is running, but from anywhere on the Internet where the user machine is.

4 Implementation

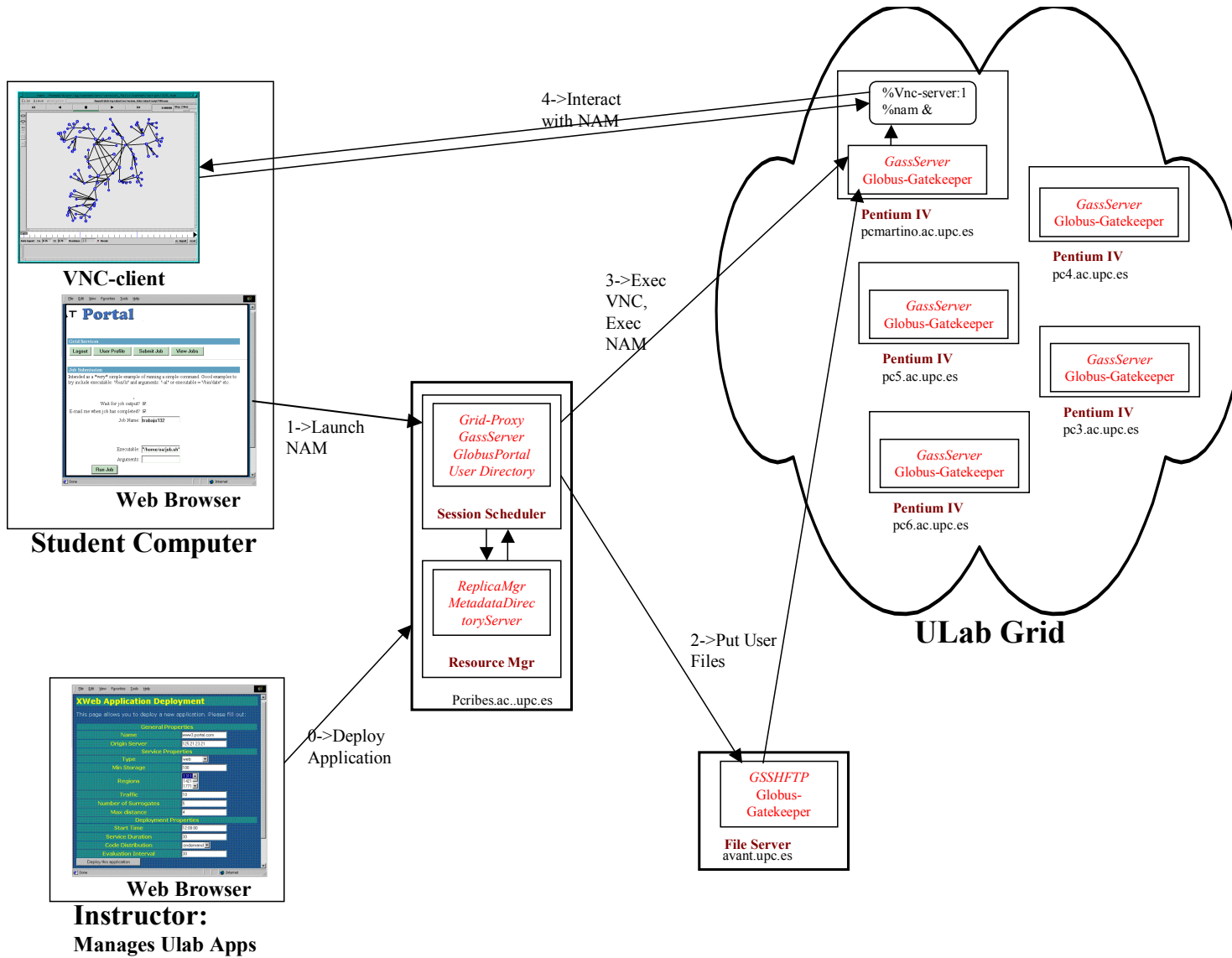
In this section the actual implementation is described. Figure 4 shows the architecture of the actual system. The system works as follows:

1. Users connect to the server machine through a web browser. Users can choose the tool they want to run from a tool list. Each user has an account in the file server machine where can store files. The user selects all files that will be used in the remote execution.
2. Once the tool and files are choosen, the server sends to the session scheduler all this information. The session scheduler should ask the resource manager which is the best machine where the tool can be executed (closer to the user, less user loaded ...). At this moment the system only has one remote machine and the resource manager always chooses this machine.
3. Before starting the remote execution, the session scheduler transfer all files the user has selected via globus GSIFTP interface from the file server machine to the selected remote machine. With the tool name, the remote machine name and all information about the user the session scheduler creates a script. The script is sended via globus GASS to the selected remote machine and is executed via globus GRAM interface. A vncserver process at the remote machine under an specific vnc account is started. For each user a new vncserver is started. The vncserver process starts an xterm which excutes the tool. After all environment is set in the remote machine the session scheduler contacts with the user server and sends to it all information needed to start the vncviewer process -machine name, encrypted password file, vncserver number.
4. A user script starts the vncviewer process at the student computer and let the user uses the tools. From the vncviewer users have the option to transmit files generated during the tool execution to user account at file server machine.
5. When users finish using the tool, user script notifies the session scheduler. Then the session scheduler kill the vncserver process at remote machine and clean the vnc account.

ULabGrid applications are managed by students instructor also through a web interface. Instructors have to:

- give access to new students to ULabGrid,
- deploy new applications and tools,
- add new machines to resource manager configuration tables,

Fig. 1. ULabGrid implementation diagram



- configure resource manager policies.

The software needed:

- Globus toolkit 2.2 has to be installed at server, fileserver and remote machines. Host certificates have to be created and installed. User certificates are given to students by their instructor.
- The vncserver has to be installed in all remote machines.
- The vncviewer has to be installed in user machines.

5 Conclusions and future work

In this paper we have described an implementation of a ULabGrid system which provides to students an easy to use but powerful environment for experimentation and learning in distant laboratory-based classrooms. ULabGrid provides students with access to a Grid of computational resources where to execute their laboratory assignments from university labs or from their homes.

We have described its architecture and implementation using Globus Grid software. Its main components are Grid computational resources, a resource manager, a session sheduler, a file service, an instructor management interface and students computers.

During implementation we have founded several issues due to the particular application we are investigating, which need to be solved. How can interactive sessions be migrated between grid resources in case a machine comes down?. Which resource mapping algorithm is best suited to this application: selecting always machines nearer to users, or least loaded machines?. How to best take into account dynamics of the system, nodes coming down, network degradations, etc., implementing a prediction service or a fast notification service? Besides solving this issues, future work in our list is: how to provide collaboration between students for group based learning clases, where students work in pairs or groups.

6 Acknowledgements

This work has been partially supported by the Spanish Ministry of Science and Technology through project TIC 2001-0995.

References

1. Adasiewicz, C.: Exploratorium: User friendly science and engineering. NCSA Access 9, 2, 10–11
2. Souder, D., Herrington, M., Garg, R. P., and DeRyke, D.: JSPICE: A component-based distributed Java front-end for SPICE. In Proceedings of the 1998 Workshop on Java for High- Performance Network Computing (1998)

3. Reinefeld, A., Baraglia, R., Decker, T., Gehring, J., Laforenza, D., Ramme, F., Romke, T., and Simon, J.: The MOL project: An open, extensible metacomputer. In Proceedings of the 1997 IEEE Heterogeneous Computing Workshop (HCW97) (1997), pp. 17–31
4. Foster I., Kesselman C., Nick J., Tuecke S.: Grid Services for Distributed System Integration. *Computer*, 35(6), 2002
5. Richardson T., Stafford-Fraser T., Wood K.R., Hopper A.: Virtual network computing. *IEEE Internet Computing*, 2(1):33–38, January-February 1998
6. Kapadia N.H., Figueiredo R.J., Fortes J.A.B.: PUNCH: Web Portal for Running Tools. *IEEE Micro*. May-June 2000
7. Kapadia, N. H., Fortes, J. A. B., Lundstrom, M. S., Royo D.: PUNCH: A Computing Portal for the Virtual University. Forthcoming in a special issue on Virtual Universities and Engineering Education.
8. Kapadia, N. H., Robertson, J. P., and Fortes, J. A. B. 1998. Interface issues in running computer architecture tools via the world-wide web. In Proceedings of the Workshop on Computer Architecture Education at the 25th Annual International Symposium on Computer Architecture (ISCA'98) (Barcelona, Spain, June 1998).
9. Foster I., Kesselman C.: The Globus project: A status report. In Proceedings of the 1998 Heterogeneous Computing Workshop (HCW'98), pp 4–18, 1998.
10. Czajkowski, K., Foster, I., Karonis, N., Kesselman, C., Martin, S., Smith, W., Tuecke, S.: A Resource Management Architecture for Metacomputing Systems. Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, 1998.
11. Fitzgerald, S., Foster, I., Kesselman, C., von Laszewski, G., Smith, W., Tuecke, S.: A Directory Service for Configuring High-Performance Distributed Computations. Proc. 6th IEEE Symp. on High-Performance Distributed Computing, pp. 365–375, 1997.
12. Raman, R., Livny, M., Solomon, M.: Resource Management through Multilateral Matchmaking, Proceedings of the 9th IEEE Symposium on High Performance Distributed Computing (HPDC9), Pittsburgh, Pennsylvania, August 2000, pp 290–291
13. Raman, R., Livny, M., Solomon, M.: Matchmaking: Distributed Resource Management for High Throughput Computing, Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, July 28-31, 1998, Chicago, IL.