Journal of Physics: Conference Series **120** (2008) 062050 doi:10.1088/1742-6596/120/6/062050

A Login Shell interface for INFN-GRID

S.Pardi[2,3], E. Calloni[1,2], R. De Rosa[1,2], F. Garufi[1,2], L. Milano[1,2], G. Russo[1,2]

[1]Università degli Studi di Napoli "Federico II", Dipartimento di Scienze Fisiche, Complesso di Monte S.Angelo – Via Cintia 80126 Napoli, Italy

[2]INFN – Sezione di Napoli, Complesso di Monte S.Angelo – Via Cintia 80126 Napoli, Italy

[3]Università degli Studi di Napoli "Federico II", C.S.I. Centro di Ateneo per i Servizi Informativi - Complesso di Monte S.Angelo – Via Cintia 80126 Napoli, Italy

E-mail: silvio.pardi@na.infn.it

Abstract.
The user interface is a crucial service to guarantee the Grid accessibility. The goal to achieve, is the implementation of an environment able to hide the grid complexity and offer a familiar interface to the final user.
Currently many graphical interfaces have been proposed to simplify the grid access, but the GUI approach appears not very congenital to UNIX developers and users accustomed to work with command line interface. In 2004 the GridShell project proposed an extension of popular UNIX shells such as TCSH and BASH with features supporting Grid computing. Starting from the ideas included in GridShell, we propose IGSH (INFN-GRID SHELL) a new login shell for the INFN-GRID middleware, that interact with the Resource Broker services and integrates in a "naturally way" the grid functionality with a familiar interface. The architecture of IGSH is very simple, it consist of a software layer on the top of the INFN-GRID middleware layer. When some operation is performed by the user, IGSH takes in charge to parse the syntax and translate it in the correspondents INFN-GRID commands according to some semantic rules specified in the next sections. The final user interacts with the underlying distributed infrastructure by using IGSH instead of his default login shell, with the sensation to work on a local machine.

1. Introduction

The University of Naples "Federico II" is one of greatest Universities in Italy. In collaboration with INFN and other national and local research institutes, it is engaged in all the main fields of applied research.
In order to support the scientific community involved in the computational sciences, starting from 2006, the University "Federico II" have started the S.Co.P.E. project (Italian acronymic for high Performance Cooperative and distributed System for scientific Elaboration), which aims to upgrade the existing CAMPUS-GRID[1] with a new supercomputing center. It will integrate CAMPUS-GRID, the new hardware and the other computational resources already available and distributed in metropolitan scale, under the Grid paradigm by using the INFN-GRID [2] distribution (see Fig.1).

In order to accelerate Grid adoption, the S.Co.P.E. project promotes the research and development of new middleware solutions to obtain a stabler and user friendly infrastructure.

Figure 1. The S.Co.P.E. infrastructure in the city of Naples

An interesting topic is the accessibility to the Grid that currently appear non immediate for a user accustomed to interact with a local operative systems. We observe that every Grid implementation introduce a new syntax for the basic operations, as jobs submission or data management.
So the goal to achieve, is the creation of an environment able to hide the grid complexity, automate some processes and offer a more familiar interface to the final user.

Currently some graphical interfaces have been proposed as GANGA[3], the web interface GENIUS[4] or Grid2Win[5] which aims to integrate in a natural way the grid infrastructure in the M.S.Window environment. The GUI approach can simplify the access to the Grid,but appears not very congenital to UNIX developers and users accustomed to work with command line interface.

In 2004 the GridShell[6] project proposed an extension of popular UNIX shells such as TCSH and BASH with features supporting Grid computing. In the last implementation, it supports Globus, Condor, Portable Batch System (PBS) and Load Sharing Facility (LSF).

Starting from the ideas included in GridShell, in this paper we propose IGSH (INFN-GRID SHELL) a new login shell for the INFN-GRID middleware, that interact with the Workload Manager System (WMS) [7] and integrates in a "naturally way" the grid functionality with a familiar interface.

The paper is structured as follows. In Section 2, we present the architecture of INFN-GRID shell, in sections 3 and 4 we discuss about jobs submission and data management facilities. In section 5 we

present a first implementation with some use case. Finally, in Section 6 we draw up the conclusions and future works.

2. The Architecture

The architecture of IGSH is very simple, it consist of a software layer on the top of the INFN-GRID middleware layer. When some operation is performed by the user, IGSH takes in charge to parse the syntax and translate it in the correspondents INFN-GRID commands according to some semantic rules specified in the next sections.
The final user interacts with the underlying distributed infrastructure by using IGSH instead of his default login shell, with the sensation to work on a local machine. Note that this architecture enables the user to utilize the commands translated by IGSH or directly the standard INFN-GRID syntax as shown in Fig.2.

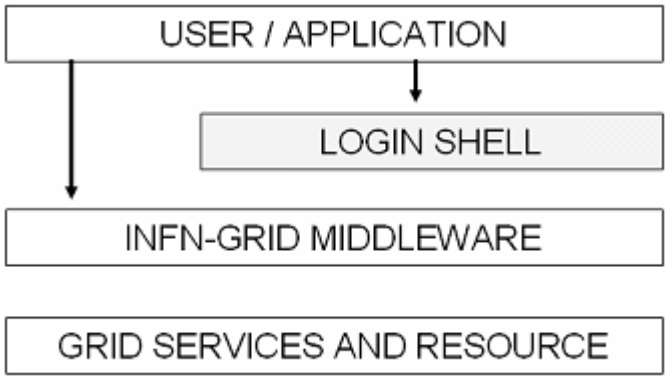

Fig.2 Architecture of IGSH. The arrows show that the user can interact with the Gird by IGSH or directly with the INFN-GRID command lines.

3. Job submission

The job submission is a crucial operation for a grid user interface. The INFN-GRID standard interface
requires that, instead of the classical command line, e.g../myexe, to execute a local job, the grid user must create a Job Description Language(JDL) file that contains some information about the executable, the standard input/output, the virtual organization and other fields required by the WMS in order to correctly schedule the job. When the JDL has been prepared, the job can be submitted to the grid by the command line:
edg-job-submit myexe.jdl

The syntax proposed by IGSH, automates the JDL creation by extracting the required information directly from the command lines and from some environmental variables defined by the user.

Within IGSH, the general syntax to submit a job to the Grid is the following:

```
[igsh]#./myexe [arg1,..,argN][> stderr_out outfile1 , …,outfileM][< infile1,…,infileT]
```

This command will be translated by IGSH in the following JDL file:

```
Executable ="myexe";
Arguments  ="arg1, …, argN"
StdOutput  ="stderr_out.out";
StdError   ="stderr_out.err";
InoutSandbox   = {"myexe”,"infile1"
,…,"infileT};
OutputSandbox  = {"stderr_out.out" ,
"stderr_out.err", "outfile1",…,
"outfileM"};
VirtualOrganisation = "virgo"
```

Where the first argument after the symbol ">", is the base name of the file where the standard error and standard output are directed by the job, while outfle1,…, outfileM are other files eventually produced by the job.
The files infile1,…, infileT that follow the symbol "<", are the files required by the job for a correct execution e.g. the executable, input files, etc.

The additional JDL field, as VirtualOrganization and other options not present in the previous example, are completed by extracting the information contained in some environmental variables e.g. $IGSH_VO, $IGSH_REQUIREMENTS etc. Each additional field is included in the JDL file only if the corresponding variable is defined.

After the JDL creation, the IGSH takes in charge the job submission to the WMS and defines a local PID for each job. The status of the jobs is shown by the IGSH command ps with different verbosity level, and every job can be killed by the IGSH command kill PID. The previous two commands are translated respectively in the INFN-GRID commands: edg-job-status and edg-job-cancel. This job management tools, hide the job ID system of the grid and offer to the final user thesensation to work locally by using a well understood interface.

Finally the command getoutput PID, allows the user to obtain the output generated by the job identified by the PID, when the job status is DONE.

4. Data Management

The INFN-GRID middleware supplies many different commands for data management, in order to support the different storage systems and protocols. To individuate a file or a directory we must always refere to the complete path, whereas in a local file systems we can use also the relative paths.

The IGSH tries to extend the standard shell commands to for navigate in the local file system, in order to define a single interface to individuate the remote storage system characteristics and make the proper translation.

The IGSH **`ls`** command, recognizes the gsiftp, srm and lfn prefixes and use them to understand witch syntax must be used. The command in the example below, lists the directory **`/home/virgo`** published by the INFN-CASCINA Storage Element, using gsiftp protocol:

```
[igsh]# ls gsiftp://grid-se.virgo.infn.it/dpm/virgo.infn.it/home/virgo
```

IGSH translates this command in:

```
glite-gridftp-ls gsiftp://grid-se.virgo.infn.it/dpm/virgo.infn.it/home/
```

The following command lists the /grid/virgo directory of the file catalog contained in the environmental variable $LFC_HOST.

```
[igsh]# ls lfn:/grid/virgo
```

The command is translated in:

```
lfc-ls /grid/virgo
```

Extending the prefix convention - gsiftp. srm and lfn - to the other data management commands e.g. **`mkdir, chmod, rename, rm`** the user can simply access the remote file system. In the same way, we define a new implementation of the commands for text management as **`cat, less, more`**, and the file transfer utility **`cp`**, thus completing the set of data management facilities and giving the sensation to navigate in a local file system.

5. A first implementation of IGSH

In order to test the capability offered by this approach, a first release of IGSH has been implemented. The software is written in C++ and Perl by using flex and yacc as lexical and grammatical parsers. The source can be downloaded from [8].
Waiting for the S.Co.P.E. infrastructure to be fully functional, the login shell has been tested on the national INFN-GRID infrastructure.
The user starts the IGSH session with the following syntax:

```
 > igsh <virtual_organization>
```

The mandatory argument will set the environmental variable $IGSH_VO used for job submission. Before showing the command prompt, the software asks for the user certificate password to authenticate the user on the Gird. If the authentication fails, the process exits, and the user cannot access to the IGSH. The authentication guarantees that the future operations performed on the IGSH login shell will be authorized by the underlaying middelware.

This first implementation supplies job management facilities like, automatic JDL generation, **`ps`** and **`kill`** commands. In Fig.3 we show how to send a simple "hello word" job with an argument, and the output of the **`ps`** command after the submission of 9 identical jobs.

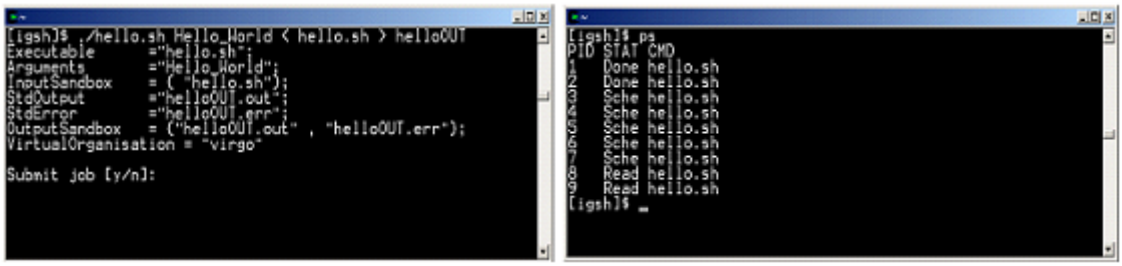

Fig.3 a. The igsh interface generate automatically the JDL and ask before submit the job to the WMS.
b. The ps command list of all job submitted to the Grid by IGSH.

As regarding the data management facility, this first implementation, allow the user to navigate in the remote file systems by implementing **`ls`** (list) and **`cd`** (change directory) commands and enable the read the file content with the **`cat`** command that accept local file, gsiftp, srm and lfn path. The Fig.5 show an use case in which the user access at file contained in the storage element of the INFN-CASCINA site, by change directory and so show the file content by using local path, with the command **`cat test1`**, as the file is in the local file system.

The last example show how the IGSH can simplify the access to the data distributed on the Grid, the user must know only the location of the files but not the syntax that must be used for each storage platform or protocol.

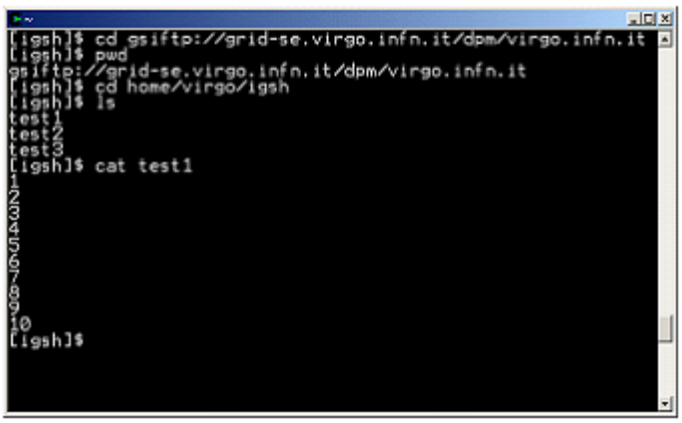

Fig.5 The cd command enable the user to change directory and user relative path as in a local file system.

6. Conclusions

The S.Co.P.E. project proposes a Grid infrastructure to support all the computational requirements of the scientific community of University "Federico II".

The success of this plan as well as of the other Grid projects, will depend on how much the researchers will adopt the GRID infrastructure. In order to diffuse the GRID use as much as possible, the middleware must offer, to the final user, simple and friendly interfaces.

The login shell interface proposed in this paper, represents an interesting alternative to the graphical interface, by offering a simplified way to access the GRID familiar to the users and developers that preferring the command line interface. The use case tests on the first implementation had show the access simplicity to the Grid and push to complete the work with other useful commands. In future we plan to investigate about the possibility to use the shell script language to create workflow to translate in the Grid syntax.

7. References

[1] For references see http://www.na.infn.it/fileadmin/infn/public/GRID/Mastroserio-CampusGrid-Bari2004.pdf
[2] For references see the official web page http://grid.infn.it/ "Ganga: a user-Grid interface for
[3] K. HArrison et al. in Proceedings of "e-Science All ATLAS and LHCb", Nottingham, 31st August - 3rd Hands Meeting 2004" September 2004
[4] For references see https://genius.ct.infn.it/
[5] For references see jessica.trigrid.it/grid2win
[6] E. Walker, T. Minyard, and J. Boisseau, "GridShell: A Login Shell for Orchestrating and Coordinating Applications in a Grid Enabled Environment", Proceedings of the International Conference on Computing, Communications and Control Technologies, Austin, Texas, August 2004, pp. 182-187.
[7] For references see http://egee-jra1-wm.mi.infn.it/egee-jra1-wm/wms.shtml
[8] A test release is available in http://people.na.infn.it/~spardi/index.php?page=igsh.htm