G-Monitor Enhanced: A Web Portal for Managing and Monitoring Application Execution on Global Grids

Martin Placek and Rajkumar Buyya

Grid Computing and **D**istributed **S**ystems (GRIDS) Laboratory
Department of Computer Science and Software Engineering
The University of Melbourne, Australia
Email: martin.placek@siemens.com and raj@cs.mu.oz.au

Abstract. G-Monitor was developed in response to the need for Web-based portals that hide low level details of accessing Grid services for deployment and execution management of applications. Following this, a requirement for a more complete solution was highlighted, which enabled the user to upload their experiment files, start experiments, collect results and authenticate the grid proxy. We responded to these requirements by enhancing G-Monitor appropriately, with the ultimate aim of removing the need for users to rely on third party applications providing such capabilities.

1 Introduction

Grids provide the infrastructure to harness a heterogeneous environment comprising of geographically distributed computer domains, to form a massive computing environment through which large scale problems can be solved. For this to be achieved, Grids need to accommodate various tools and technologies that can support: security, uniform access, resource management, scheduling, application composition, computational economy and accounting [3][7]. Additionally, Grids need to provide a ubiquitous user-interface that hides complexities associated with the deployment and management of execution applicable to experiments using distributed resources.

A sample wide-area Grid computing environment is shown in Figure 1. In this environment, the Grid consumers interact with GRBs (Grid Resource Brokers) such as Nimrod-G [7] responsible for scheduling applications on the distributed resources, based on their availability, cost, capability, and user-specified QoS (Quality of Service) requirements. The initial version of G-monitor [20] was created to provide the users with a pervasive interface that allows them to monitor and control the execution of their applications. The enhanced version of G-monitor encompasses all the functionality provided by the initial version, but also provides a multi-user environment where the users are able to deploy and collect experiment files, authenticate their grid proxy and analyze graphs illustrating experiment progress. Ultimately G-monitor has been enhanced to provide the users with a more complete set of functionality, removing the need for third party applications. G-monitor was developed and is maintained as part of the Gridbus project [8].

As Grids in general are distributed in nature, it is important that G-Monitor provides the users with an ubiquitous interface that allow them to access from anywhere at anytime from any access device. The users only need access to a web browser and with it they are able to fully utilize the functionality provided by G-Monitor. As G-Monitor is a web-based portal the users do not have to worry about setting up SSH or exporting X displays, which can all be made more complicated depending on their firewall settings.

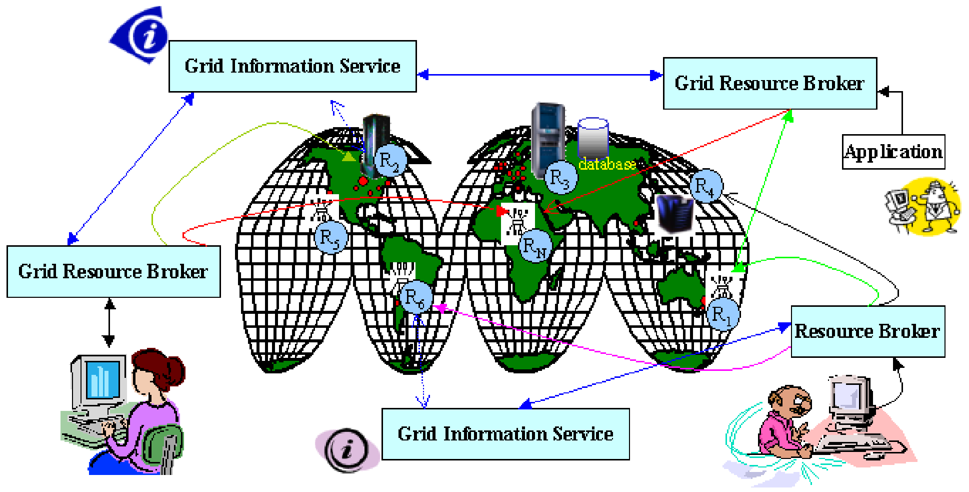


Figure 1: A Sample Grid computing Environment and key components.

G-Monitor interacts with the Nimrod-G GRB (Grid Resource Broker), which is implemented using low-level services provided by middleware such as Globus [4] and Legion [5]. It provides a consistent interface that is easy to use, enabling the end-user to monitor, control, and steer execution of application jobs running within the Grid environment. G-Monitor is flexible enough to be run from anywhere without the need for custom client software. With the use of a web browser the users will be able to fully utilize the functionality of enhanced G-Monitor outlined below:

1. **Retrieve and set QoS (quality of service) parameters, such as**
   - Deadline, Budget and Optimisation preferences
2. **Monitor/Control Jobs Information, such as**
   - Start, Stop, Grid node status and Execution time
3. **Monitor Resource status, such as**
   - Server name, Host name, Service cost and Status
4. **Monitor Experiment status, such as**
   - Deadline, Budget, Job status and Resource status
5. **Multi-User Interface**
   - User login and User preferences relating to available brokers and interface
6. **Real time graphs**
   - Graphs of the experiment progress are provided throughout experiment execution
7. **Start Experiment**
   - Ability to upload experiment files using the G-Monitor interface and start experiments
8. **Grid Authentication**
   - Ability to update Grid Authentication, without the need to access the Nimrod Broker
9. **Collect Experiment Files**
   - Upon completion, experiment files with results can be collected

The remainder of this paper provides an insight into how G-Monitor fits into the Grid Infrastructure and how it has evolved from its initial state to incorporate these new functionalities. We initially discuss G-Monitor in relation to other similar applications currently available. We then discuss G-Monitor's new Architecture and Design, Implementation process, and various example applications. Finally we conclude by reflecting upon the benefits these enhancements provide to the users.

2 Related Work

A number of projects have explored development of toolkits for development of Grid portals and construction of application specific portals. Some representative efforts include GridPort [9] and HotPage [10] from San Diego Supercomputing Centre, GPDK (Grid Portal Development Kit) [14] from Lawrence Berkeley National Laboratory, Legion portal [18] from the University of Virginia, GRB portal [13] from University of Lecce, SGE (Sun Grid Engine) Technical Portal [11] from Sun Microsystems, and PBSWeb[12] from the University of Alberta. GridPort and GPDK allow construction of application-specific portals by providing generic interfaces/libraries for accessing Grid resources using Globus. Legion portal provides an interface to Legion Grid infrastructure. As these toolkits provide an interface to low-level Grid services, the users of portals constructed using them are responsible for manually identifying resources that are suitable for running their applications and deploying them. The same is the case with the GRB portal as it also provides Web-based interface to Grid resources running Globus and the users are responsible for selection of suitable resources for the execution of their applications. Unlike these systems, the unique aspect of G-monitor is that it has been designed to provide access to high-level Grid services (e.g., the Nimrod-G broker and Gridbus scheduler). As high-level Grid services (e.g., Nimrod-G) are in turn implemented using low-level Grid servies (e.g., Globus), they hide issues related to the identification of resources that are suitable for running user applications and their aggregation. SGE Portal and PBSWeb basically provide Web-portal for accessing cluster resources management systems, SGE and PBS (Portal Batch System) respectively.

3 Architecture and Design

This section is divided into a discussion of the architecture and how G-Monitor fits into the Grid environment; followed by a design summary detailing the main modules which make up the implementation of G-Monitor.

3.1 Architecture

The architecture of G-Monitor and its interaction with other components within a Grid architecture [19] framework is shown in Figure 2. This figure appears similar to a typical grid architecture, having three major categories: Grid Resources, Grid Broker, and Applications and Portals. G-monitor resides in the application level and has the responsibility of providing an easy to use interface to the Nimrod-G broker system. With the enhanced version of G-Monitor the users may upload their experiment files, start their experiment and upon completion download the resulting output files.

G-Monitor is able to provide a high level interface by utilising tools such as Nimrod-G and Globus to handle all the low-level requirements of Grid Resources. Nimrod-G plays a broker role by farming out tasks to each grid node using the "Globus Client Interface". Globus is used by Nimrod-G for submitting jobs, authentication, security and gathering information about resources. Grid Resources provide a hybrid environment which Globus and Nimrod-G play a vital role in managing and ultimately providing a unified interface which G-Monitor utilises.

G-Monitor provides a web portal, which is used by the end-user and therefore is categorised as a component within the "Applications and Portals" category. G-Monitor resides on a web server, which sits between the GRB and the Web browser. The Grid consumer uses the Web browser to access G-Monitor's functionality. The users' requests are served by G-Monitor communicating with the GRB. The GRB manages all the Grid nodes using low-level Grid middleware services, keeping a detailed database of information about the status of the Grid nodes whilst offering scheduling and farming out facilities.

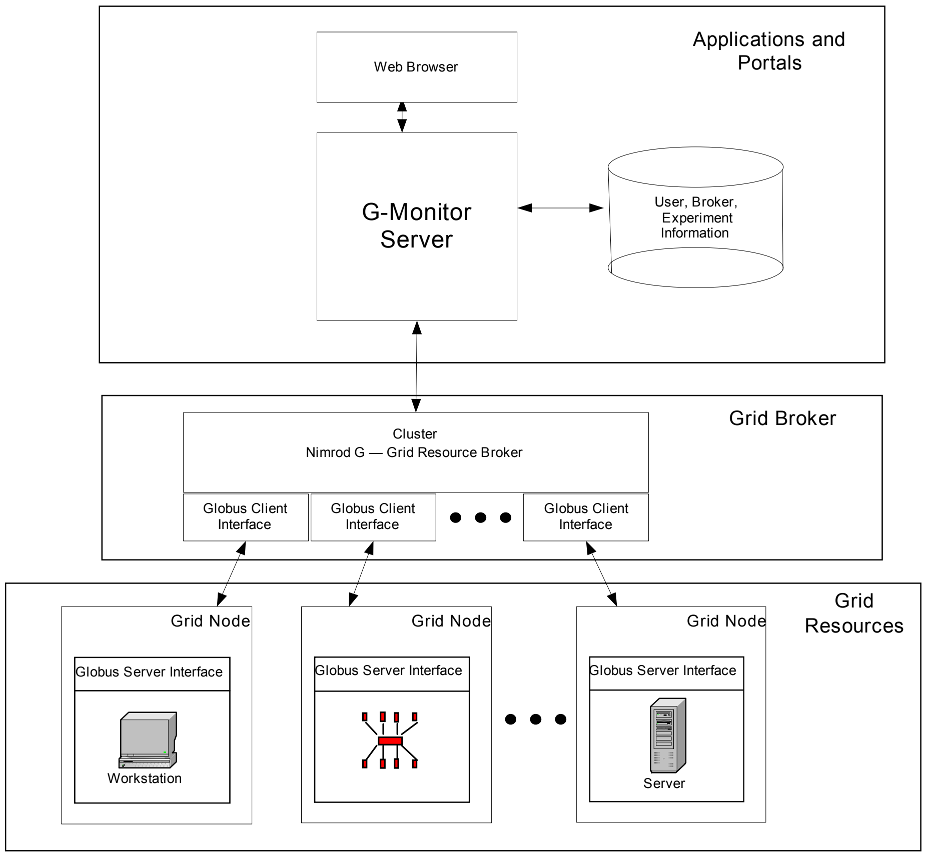


Figure 2: G-Monitor: Grid Architecture.

G-Monitor stores information locally on the G-Monitor server. This information includes user account information, broker information and experiment information. Storing this information locally provides the user with the following set of functionality:

- A multi-user environment that remembers user preferences
- Ability to run many experiments at the same time, as multiple brokers can be logged and monitored
- Provide graphs of how the experiment is progressing and how it has progressed throughout its execution history

3.2 Design

G-Monitor's design has evolved dramatically since its first incarnation. It now not only communicates with the broker system using Nimrod-G protocols [2], but also uses Telnet and FTP protocols to issue commands and transfer files respectively – making it possible for the users to transfer their experiment files to and from the broker and start the experiment all through the G-monitor interface. G-monitor stores information locally using a flat file structure. Information stored includes username and password, each of the user's preferences including broker attributes and experiment information. G-Monitor has the capability of monitoring multiple experiments in the background, polling the GRB for information at a set interval and logging the data to a flat file. The experiment information logged is then used to generate graphs, which aid the users in analyzing how their experiment is progressing.

The design (Figure 3) behind G-Monitor can be split up into the following main categories:

Category	Modules	Description
Nimrod-G Server transport modules	▪ Telnet ▪ Ftp ▪ Socket and Nimrod-G communications	Responsible for issuing commands, transferring files and communicating with the Nimrod-G server
G-Monitor Data Storage and Re-trieval	▪ Data Storage and Re-trieval	This module serves requests for user, broker and experiment information. A flat file structure is used to store all the information
G-Monitor Function-ality Modules	▪ Administration ▪ Login ▪ User preferences ▪ TimeZone	Provide the user with the means to update the user and broker information. The administration, login and user preferences modules generate HTML which is then served back up to the user's web browser. The time-zone module works out which time-zone the user resides in and converts times to the user's time-zone
Core Broker Functionality Modules	▪ Grid Authentication ▪ Start Experiment ▪ Job Information ▪ Resource Information ▪ Graph Generator	These modules are responsible for all the broker related functionality and therefore communicate with the Nimrod G server via one of the transport mod-ules. All these modules rely on the "data storage and retrieval" module for retrieving broker and user information for they must be aware under which user and broker they are to execute their functions. These modules also generate HTML which is then served back up through the web server to the user's web browser

A further description of the implemented modules is as follows:

Module Name	Description
Grid Authentication	Responsible for Grid Proxy Authentication, uses telnet module to login into broker and issue grid-proxy-init commands based on user specification
Start Experiment	Provides the user with the functionality to start experiments on the broker. The user can either upload experiment files or select existing files to start experiment.
Collect Files	Upon completion of experiment the user is able to this module to download their resultant experiment files
Administration	Allows the creation and modification of user accounts on G-Monitor
User Preferences	Store broker, experiment and interface preferences
Login	Responsible for G-monitor's authentication
TimeZone	Logs which time-zone the user is in and is used for making time compensation calculations
Job Information	Presents the user with a breakdown of all current jobs that are running for the experiment they are monitoring
Resource Information	Lists the available resources to the user
Graph Generator	Runs constantly on the actively monitored experiment and any experiments the user wishes to monitor in the background. The scripts responsible poll the Broker for the current experiment status at user specified interval and log it in a file. This file is then used to generate experiment graphs
Data Storage and Re-trieval.	Responsible for providing an interface for storing information in a flat file sys-tem
Nimrod G communica-tions module	Responsible for establishing a TCP/IP socket connection with the Nimrod-G broker server and parsing the Nimrod-G protocol

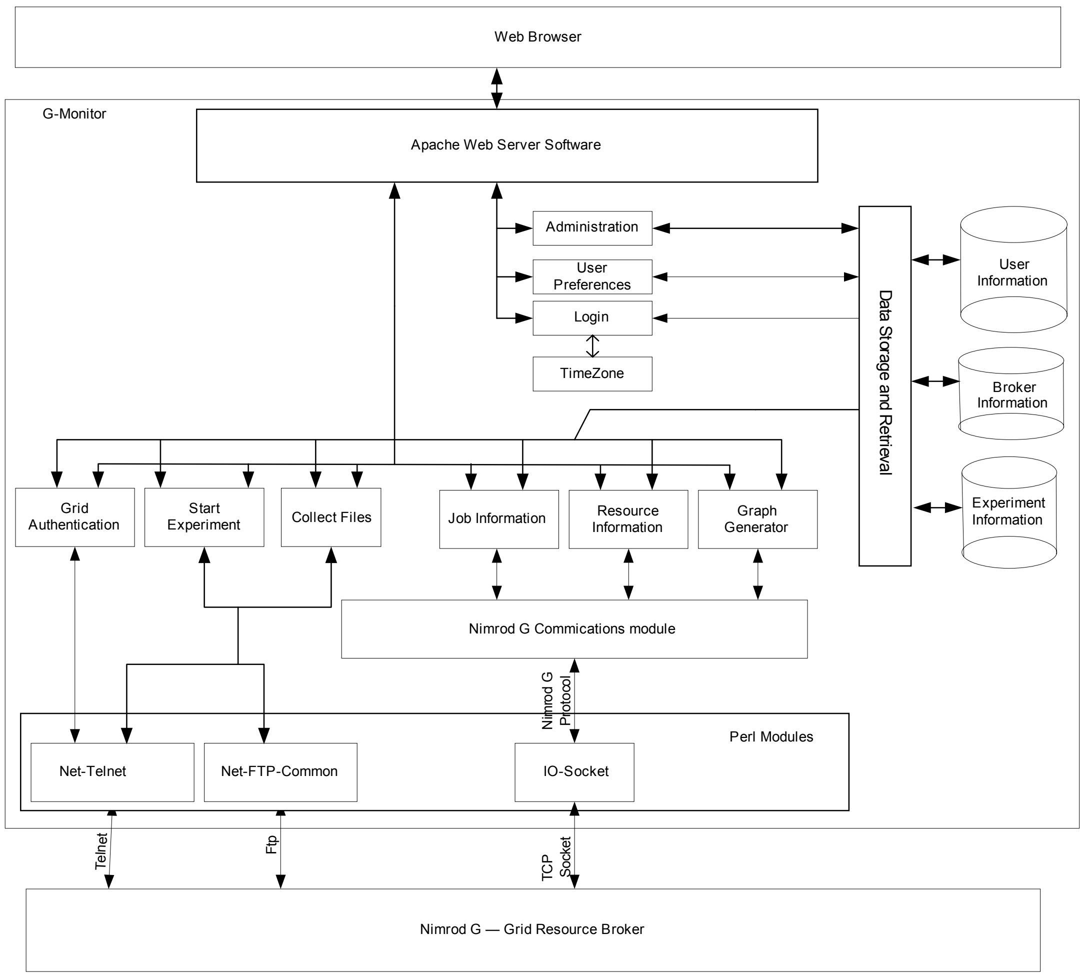


Figure 3: G-Monitor: Design.

4 Implementation

One of the main aims of developing G-Monitor was to overcome problems and limitations associated in using interfaces with heavy bandwidth requirements. G-monitor was built with a light weight interface to overcome these limitations. It was important that any modifications/enhancements made to G-Monitor kept the light weight interface, otherwise defeating the original purpose of building G-Monitor. The enhancements were made with an effort to maintain the original architecture [20].

G-Monitor is a web-based portal that uses the Apache [1] web server software to serve requests made by the user's web browser. Apache's CGI module was used to call scripts that in turn provided the functionality the user requested. The scripting language used to implement and enhance G-Monitor was Perl. The Nimrod-G Server transport modules were implemented with the help of the following Perl modules:

Perl Module Name	**Description**
IO-Socket	TCP/IP socket module
Net-FTP-Common	Provides high level ftp interface
Net-Telnet	Provides a series of telnet functions

The "Graph Generator" Perl scripts use a special version of GnuPlot that was compiled against GD graphics library [25] to generate the final graphs in PNG (Portable Network Graphics) image format. The user is able to specify which experiments they wish to be "monitored" and in doing so the Graph Generator scripts are forked-off in the background polling the broker for data and then using it to generate the graphs.

5 Use Case Study

Research carried out at The University of Melbourne, by the Experimental Particle Physics (EPP) group [23] and the GRIDS Lab, involves investigation into the development of Grid technologies for high energy physics. The EPP group is a member of the Belle collaboration which consists of 400 physicists from 50 institutions around the world [24]. The Belle experiment, situated at the KEK B-factory in Japan, is operational and has been collecting data and producing results for a number of years. G-Monitor has been used in the Belle application analysis on global Grids demonstration at the 4th PRAGMA workshop [21] [22] held in June 2003, Monash University, Melbourne, Australia.

The demonstration involved the construction of a data grid test bed. The grid test bed was constructed within 9 days prior to the PRAGMA workshop and was made possible by the loan of high performance servers from IBM Asia Pacific. The demonstration setup depicting application deployment and access locations of various Grid entities is shown in Figure 4. Servers from around Australia were used to analyze data collected from the Belle simulations. During the demonstration G-Monitor was used to manage and monitor experiment progress at both application and job levels. An application jobs execution statistics plot from the experiment are shown in Figure 8.

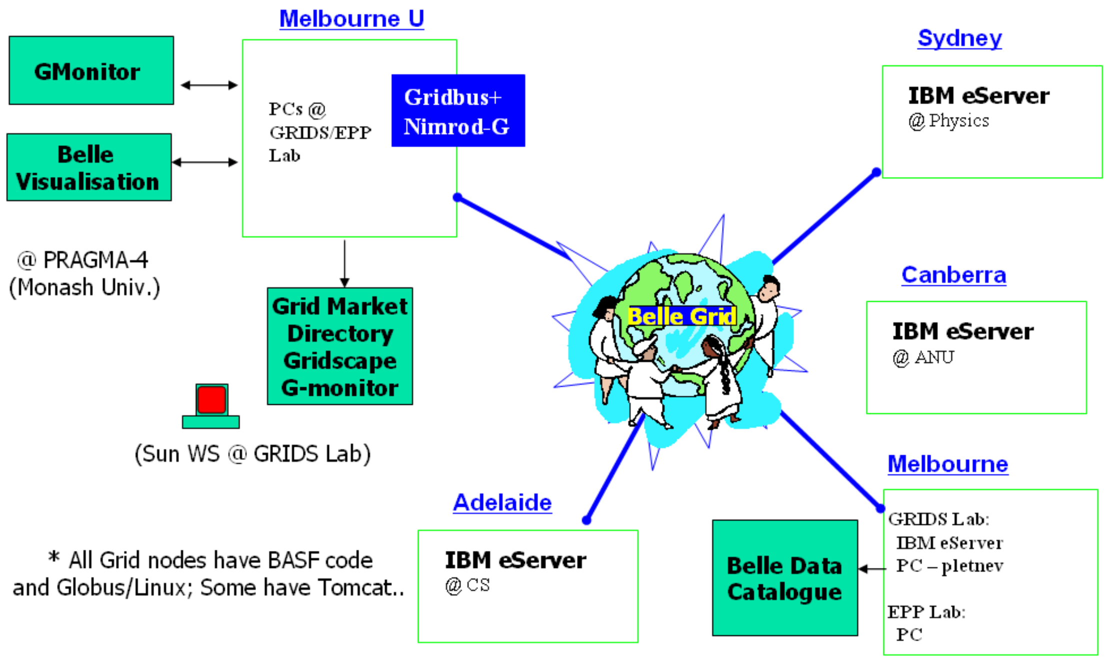


Figure 4: G-Monitor Usage during the Belle Analysis demo at PRAGMA conference.

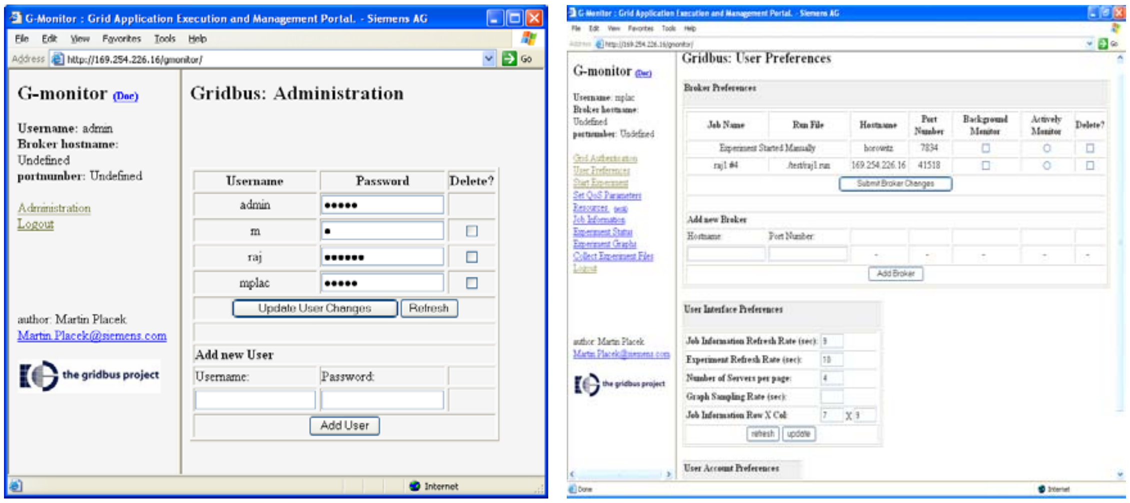
We shall now briefly cover the functionality provided by G-Monitor and discuss in detail the new set of functions provided by the enhanced version. G-Monitor's original specifications support the following set of functions and are discussed in detail in [20].

- User sets QoS parameters
- User experiment status
- Job Information and Experiment Status
- Grid Infrastructure status

In the following sub-sections we will walk through some screenshots to demonstrate G-Monitor's new set of features and their usefulness in starting and managing application execution in Grid environments.

5.1 Multi-user functionality

G-Monitor now provides a multi-user interface, whereby each user of G-monitor has an account. These user accounts can be created and administered by the "admin" user account, from the "Administration" Page as seen in Figure 5. Providing multi-user functionality allows the user to customize G-monitor using the preferences page [Figure 5]. The Preferences page allows the user to manually enter brokers they know experiments are running on, and select which of those brokers they would like to actively monitor. The user also has the option to monitor experiments in the background. This enables G-Monitor to poll the broker for the set interval collecting data on the experiments progress. Then when the user decides to "Actively" monitor the experiment they are able to see the entire experiment's progress by looking at the experiment graphs. The user also has the ability to customize refresh rates, sample rates and general cosmetic items via the Preferences page.


G-Monitor : Grid Application Execution and Management Portal. - Siemens AG
File Edit View Favorites Tools Help
Address http://169.254.226.16/gmonitor/
Go
Internet

G-monitor (Doc)

Username: admin
Broker hostname: Undefined
portnumber: Undefined

Administration
Logout

author: Martin Placek
Martin.Placek@siemens.com

the gridbus project

Gridbus: Administration

Username	Password	Delete?
admin	•••••	
m	•	☐
raj	••••••	☐
mplac	•••••	☐

Update User Changes | Refresh

Add new User

Username:	Password:

Add User

G-monitor (Doc)

Username: mplac
Broker hostname: Undefined
portnumber: Undefined

Grid Authentication
User Preferences
Start Experiment
Set QoS Parameters
Resources
Job Information
Experiment Status
Experiment Graphs
Collect Experiment Files
Logout

author: Martin Placek
Martin.Placek@siemens.com

the gridbus project

Gridbus: User Preferences

Broker Preferences

Job Name	Run File	Hostname	Port Number	Background Monitor	Actively Monitor	Delete?
Experiment Started Manually		horowitz	7834	☐	○	☐
raj1 #4	./test/raj1.run	169.254.226.16	41518	☐	○	☐

Submit Broker Changes

Add new Broker

Hostname:	Port Number:					
		-	-	-	-	-

Add Broker

User Interface Preferences

Job Information Refresh Rate (sec):	5
Experiment Refresh Rate (sec):	10
Number of Servers per page:	4
Graph Sampling Rate (sec):	
Job Information Row X Col:	7 X 3

refresh | update

User Account Preferences

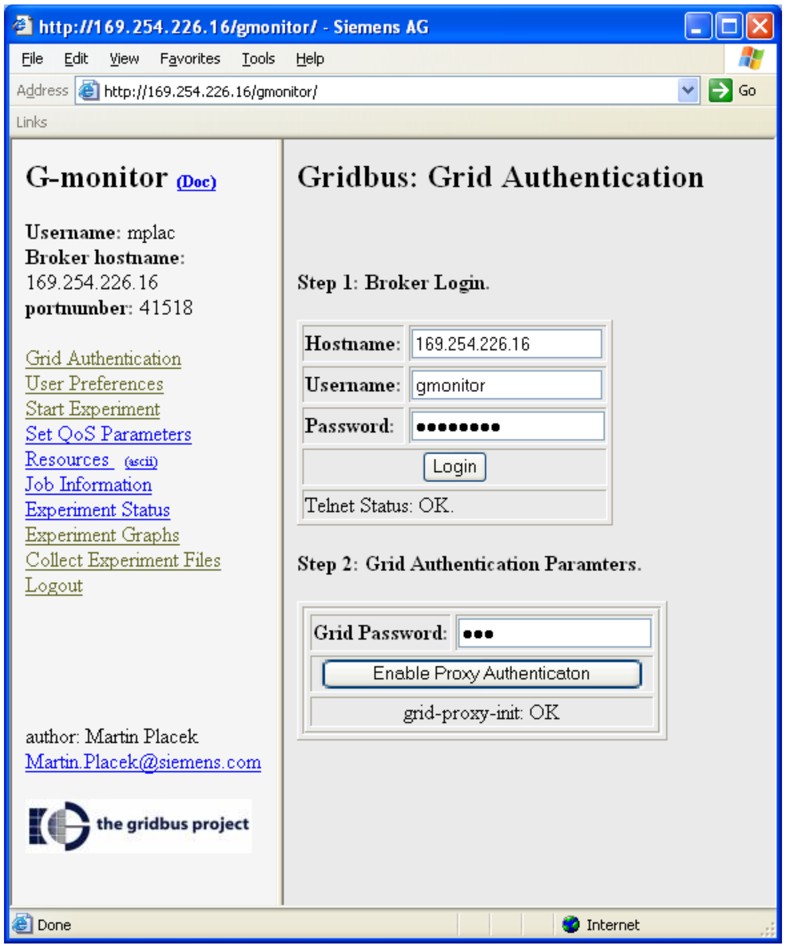
Figure 5: G-Monitor Administration and Preferences functionality.

5.2 Grid proxy authentication

The Globus toolkit allows people to share resources to create the Grid environment. The Globus software relies on the user to create a proxy certificate, which is then used as a means to authenticate the user when using Grid resources. These certificates subsequently expire and need to be renewed. This certificate can be renewed by using the "Grid Authentication" page [Figure 6]. This page has been divided into the following two steps:

1. *Broker login:* The user is required to provide the hostname and user account of the broker they wish to renew their certificates

2. *Grid Authentication Parameters:* Once the user specifies the grid-proxy-init password and clicks on the "Enable Proxy Authentication" button, G-Monitor will attempt to renew the Grid certificates and provide the user with feedback



Username: mplac
Broker hostname: 169.254.226.16
portnumber: 41518

author: Martin Placek
Martin.Placek@siemens.com

Gridbus: Grid Authentication

Step 1: Broker Login.

Hostname:	169.254.226.16
Username:	gmonitor
Password:	••••••••

Telnet Status: OK.

Step 2: Grid Authentication Paramters.

Grid Password:	•••

grid-proxy-init: OK

Figure 6: G-Monitor Grid Authentication.

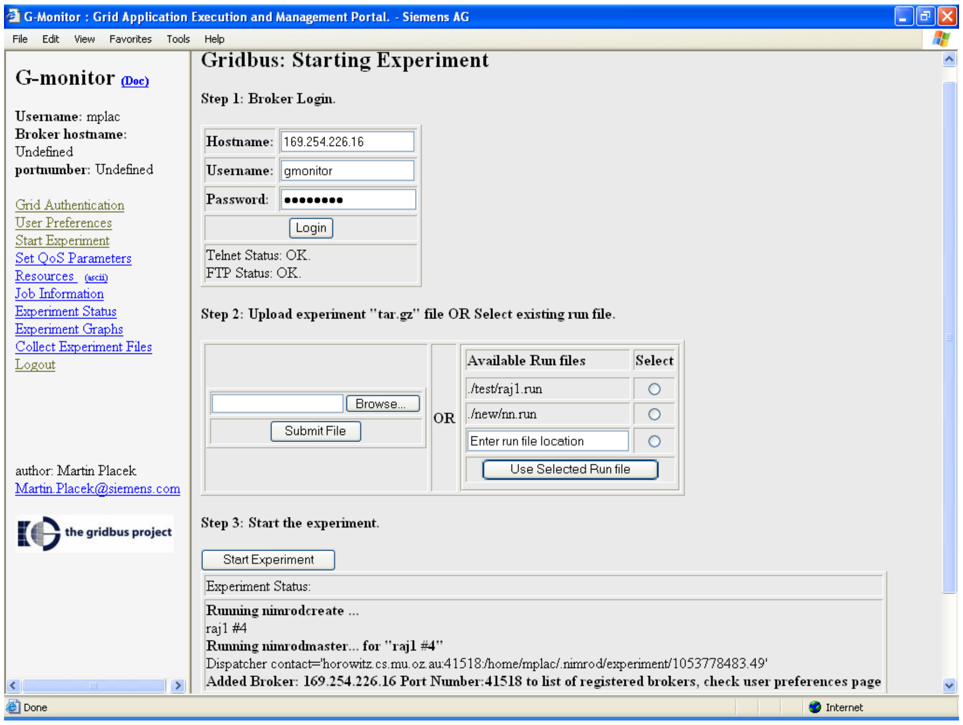
5.3 Start Experiment

The user has the ability to start experiments using the “Start Experiment” page [Figure 7]. This page has been divided into the following three steps:

1. *Broker Login:* The user is required to provide the hostname and user account of the broker they wish to start the experiment on. When the user selects “Login”, G-Monitor will check to see that the broker is up and that the user account is valid. Following this it provides feedback to the user on both these accounts

2. *Experiment Files:* In this step the user is required to either upload their experiment files or select existing experiment files which are present on the broker. Upon doing so the user will be provided with feedback as to the experiment files which will be used when starting the experiment

3. *Starting the Experiment:* Once the user clicks on the “Start Experiment” button, G-Monitor will issue Nimrod-G commands on the broker to start the experiment with the experiment files uploaded/selected. The output of these commands is output to the screen ensuring the user is kept up-to-date with the experiment. If the experiment has been started successfully a broker entry will be automatically made in the user’s preference screen [Figure 5]. The user is then able to actively monitor the experiment and use all the monitoring and management functionality provided by G-Monitor


G-Monitor : Grid Application Execution and Management Portal. - Siemens AG
File Edit View Favorites Tools Help
Done
Internet

G-monitor (Doc)

Username: mplac
Broker hostname: Undefined
portnumber: Undefined

Grid Authentication
User Preferences
Start Experiment
Set QoS Parameters
Resources (ascii)
Job Information
Experiment Status
Experiment Graphs
Collect Experiment Files
Logout

author: Martin Placek
Martin.Placek@siemens.com

the gridbus project

Gridbus: Starting Experiment

Step 1: Broker Login.

Hostname:	169.254.226.16
Username:	gmonitor
Password:	••••••••

Login

Telnet Status: OK.
FTP Status: OK.

Step 2: Upload experiment "tar.gz" file OR Select existing run file.

Browse... Submit File

OR

Available Run files	Select
./test/raj1.run	○
./new/nn.run	○
Enter run file location	○

Use Selected Run file

Step 3: Start the experiment.

Start Experiment

Experiment Status:

Running nimrodcreate ...
raj1 #4
Running nimrodmaster... for "raj1 #4"
Dispatcher contact='horowitz.cs.mu.oz.au:41518:/home/mplac/.nimrod/experiment/1053778483.49'
Added Broker: 169.254.226.16 Port Number:41518 to list of registered brokers, check user preferences page

Figure 7: G-Monitor Grid Authentication.

5.4 Real/time graphs

The user is able to access the experiment graphs [Figure 8] by clicking on the "Experiment Graphs button". These graphs are generated and updated for experiments that are currently being monitored and/or have been selected to be monitored in the background from the "User Preferences" page [Figure 5]. The two graphs provided are as follows:

- *Job Numbers Graph:* Plots the number of jobs ready, running, completed and failed against time. Enables the user to get an idea as to how their experiment is progressing as a whole.
- *Percentage Graph:* Plots the percentage of completed jobs and budget spent against time. Provides the user with an insight into the likelihood of their experiment completing within budget parameters.

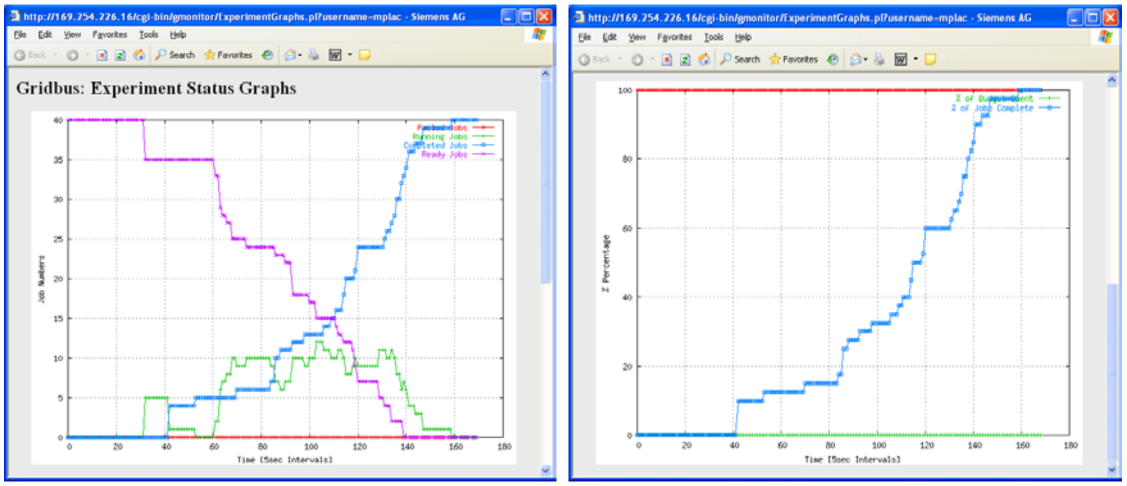


Figure 8: Job Numbers and Percentage complete Graphs from PRAGMA demo.

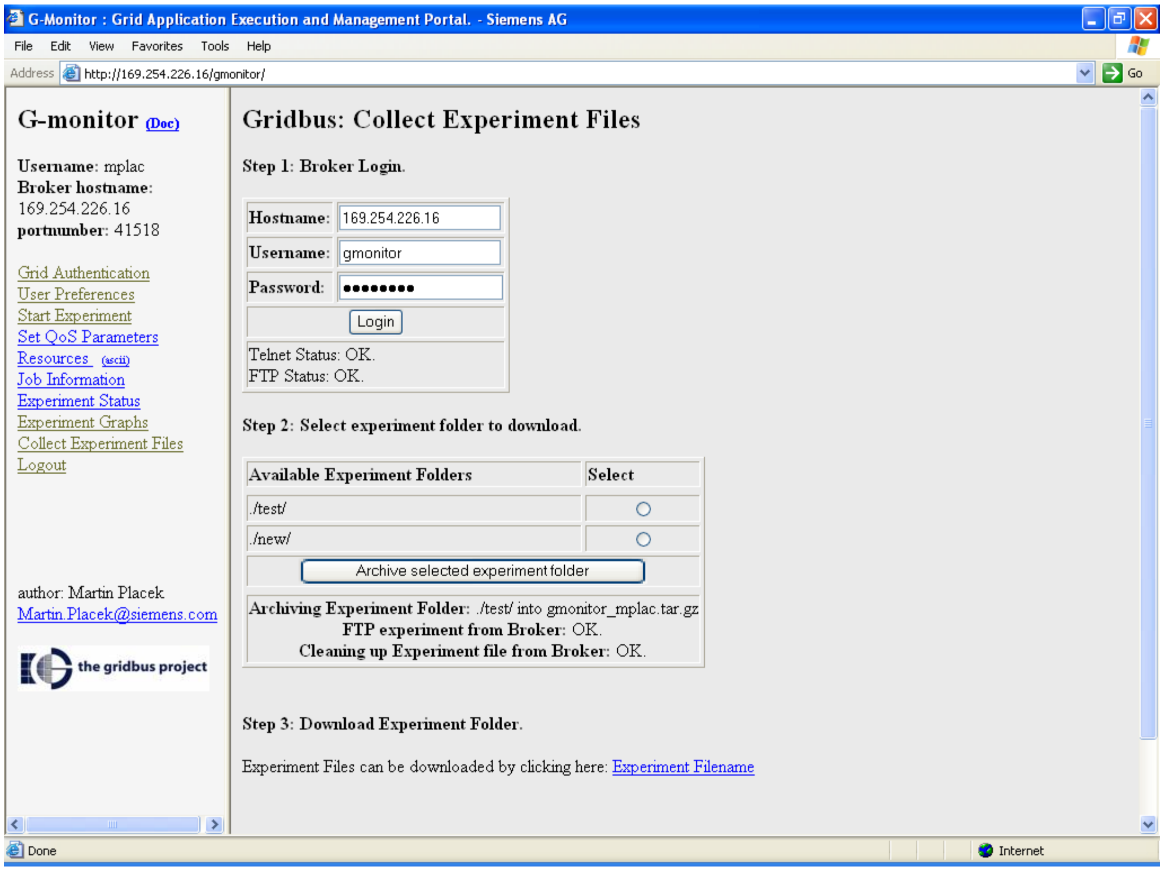
5.5 Collect Files

Upon completion of the experiment the user may need to download their output files. The "Collect Experiment Files" page [Figure 9] can be used to download the files from the broker. This process behind downloading experiment files from the broker server has been divided into the following three steps:

- *Step 1 – Broker Login:* The user is required to provide the hostname and user account of the broker they wish to download the experiment files from. Upon providing the broker details the user will be presented with a list of directories of all the experiments present under that user account.

- *Step 2 – Select Experiment folder to download:* The user is able to select the experiment folder they wish to download into. Upon clicking the "Archive selected experiment folder" the experiment folder will be archived using tar, compressed using gzip and ftp'd to the G-Monitor Server. The user is provided with feedback of this process

- *Step 3 – Downloading the experiment file from the G-Monitor server.* The user is provided with a link to the file and can download the experiment compressed archive by clicking on the "Experiment Filename" link


G-Monitor : Grid Application Execution and Management Portal. - Siemens AG
File Edit View Favorites Tools Help
Address http://169.254.226.16/gmonitor/
Go

G-monitor (Doc)

Username: mplac
Broker hostname: 169.254.226.16
portnumber: 41518

Grid Authentication
User Preferences
Start Experiment
Set QoS Parameters
Resources (ascii)
Job Information
Experiment Status
Experiment Graphs
Collect Experiment Files
Logout

author: Martin Placek
Martin.Placek@siemens.com

the gridbus project

Gridbus: Collect Experiment Files

Step 1: Broker Login.

Hostname:	169.254.226.16
Username:	gmonitor
Password:	••••••••
	Login

Telnet Status: OK.
FTP Status: OK.

Step 2: Select experiment folder to download.

Available Experiment Folders	Select
./test/	○
./new/	○

Archive selected experiment folder

Archiving Experiment Folder: ./test/ into gmonitor_mplac.tar.gz
FTP experiment from Broker: OK.
Cleaning up Experiment file from Broker: OK.

Step 3: Download Experiment Folder.

Experiment Files can be downloaded by clicking here: Experiment Filename

Figure 9: Collect Experiment Files.

6 Conclusion and Future Work

In this paper we have identified the need for Web based portals that provide a light weight, ubiquitous, easy to use interface which enable the execution and management of applications in a Grid environment. We outlined the enhancements to G-Monitor, which allow the user to start and manage experiments of application jobs on Global Grids. Unlike related systems, G-Monitor provides web-based interface to high-level Grid services (e.g. the Nimrod-G broker). This means that users of G-Monitor delegate the responsibility of selection of appropriate resources for execution of their application jobs to the specific broker. We have used G-Monitor successfully in a number of Grid demonstrations including the PRAGMA-4 workshop and SC2002 [16] conference.

The initial version of G-Monitor was designed specifically around monitoring Grid experiments, while providing only limited experimental control. The new set of enhancements provides a multi-user interface whereby they are able to upload and start their experiment, manage and monitor their experiment using graphs which are dynamically updated, and finally collect the resultant experiment files. The user is also able to authenticate their grid proxy privileges using G-Monitor. G-Monitor has evolved into a complete solution, removing the need for the user to rely on other third party software to upload their experiment files, or the need to manually login to the broker for the purpose of starting their experiment, collecting their experiment files and authenticating their grid proxy privileges.

Some possible future developments for G-Monitor are as follows:

- Enhance security by providing the user with the option of using SSH and SCP services, instead of Telnet and FTP respectively.
- Build tailored component to G-Monitor for a specific applications (e.g. dynamic job allocation based on results of prior completed jobs – as might be useful in Neural Networks).
- Research the possibility of providing a developers toolkit enabling users to customize G-Monitor to their specific needs themselves.
- Provide ubiquitous access to the Grid by supporting clients that originate from mobile and handheld devices by extending G-monitor presentation layer.

Availability

The G-Monitor software with source code can be downloaded from the Gridbus Project website:

`http://www.gridbus.org/`

Acknowledgement

We would like to thank our colleagues from Sydney, Canberra, Adelaide, and Melbourne universities for their contribution to the Belle Analysis Data Grid testbed, which also served as an experimental platform for G-monitor web portal demonstration. We thank Emmett Sheppard and Anthony Sulistio for their invaluable comments on the paper.

The Grid nodes (eServer) shown in Figure 4 are provided by IBM for the testbed. The Solaris server running GMD (Grid Market Directory) is provided by Sun Microsystems. Support from Michael Monty and Benjamin Khoo (IBM Singapore) during the preparation of PRAGMA demonstration is acknowledged.

References

[1] Apache Web Server, http://www.apache.org/

[2] D. Abramson et. al., *Using Application Programming Interface*, Chapter 9, EnFuzion Manual, 2002. Available at: http://www.csse.monash.edu.au/cluster/enFuzion/api.htm

[3] I. Foster and C. Kesselman (editors), *The Grid: Blueprint for a Future Computing Infrastructure*, Morgan Kaufmann Publishers, USA, 1999.

[4] I. Foster and C. Kesselman, Globus: A Metacomputing Infrastructure Toolkit, *International Journal of Supercomputer Applications*, Volume 11, Issue 2, Sage Publications, USA, 1997.

[5] A. Grimshaw and W. Wulf, *The Legion Vision of a Worldwide Virtual Computer*, Communications of the ACM, vol. 40(1), January 1997.

[6] NeuroGrid Project, http://www.gridbus.org/neurogrid/

[7] R. Buyya, D. Abramson, and J. Giddy, *An Economy Driven Resource Management Architecture for Global Computational Power Grids*, Proceedings of the 2000 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2000), Las Vegas, USA., June 2000.

[8] The Gridbus Project, http://www.gridbus.org

[9] M. Thomas, S. Mock, J. Boisseau, M. Dahan, K. Mueller, D. Sutton, *The GridPort Toolkit Architecture for Building Grid Portals*, Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing, Aug 2001.

[10] NPACI HotPage -- https://hotpage.npaci.edu/

[11] Sun, *Sun Grid Engine Portal*, http://www.sun.com/solutions/hpc/pdfs/TCP-final.pdf

[12] G. Ma and P. Lu, *PBSWeb: A Web-based Interface to the Portable Batch System*, Proceedings of the 12th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS), Las Vegas, Nevada, U.S.A., November 6-9, 2000. http://www.cs.ualberta.ca/~pinchak/PBSWeb/

[13] G. Aloisio, M. Cafaro, P. Falabella, C. Kesselman, R. Williams, *Grid Computing on t he Web using the Globus Toolkit,* Proceedings of the 8th International Conference on. High Performance Computing and Networking Europe (HPCN Europe 2000), Amsterdam, Netherlands, May 2000.

[14] J. Novotny, The Grid Portal Development Ki*t, Journal of Concurrency and Computation: Practice and Experience (CCPE)*, Volume 14, Issue 13-15, Wiley Press, Nov.-Dec., 2002.

[15] Global Grid Testbed Collaboration - http://scb.ics.muni.cz/static/SC2002/

[16] SC 2002, International Conference on High Performance Networking and Computing, November 16-22, 2002. http://www.sc-conference.org/sc2002/

[17] R. Buyya, S. Date, Y. Mizuno-Matsumoto, S. Venugopal, and D. Abramson, *Economic and On Demand Brain Activity Analysis on Global Grids*, Technical Report, Dept. of Computer Science and Software Engineering, The University of Melbourne, Australia, February 2003.

[18] A. Natrajan, A. Nguyen-Tuong, M. Humphrey, M. Herrick, B. Clarke, and A. Grimshaw, The Legion Grid Portal**,** *Journal of Concurrency and Computation: Practice and Experience (CCPE)*, Volume 14, Issue 13-15, Wiley Press, Nov.-Dec., 2002.

[19] R. Buyya, *Economic-based Distributed Resource Management and Scheduling for Grid Computing*, Thesis, School of Computer Science and Software Engineering, Monash University, Australia, April 2002.

[20] M. Placek and R. Buyya, *G-Monitor: A Web Portal for Monitoring and Steering Application Execution on Global Grids*, Proceedings of the International Workshop Challenges of Large Applications in Distributed Environments CLADE 2003, IEEE Computer Society Press, USA, June 2003.

[21] PRAGMA Demonstration, http://epp.ph.unimelb.edu.au/epp/grid/pragma.html

[22] PRAGMA, Pacific Rim Applications and Grid Middleware Assembly, The 4th Workshop, June 5-6, 2003 http://www1.qpsf.edu.au/pragma/index.html

[23] Experiment Particle Physics group, The University of Melbourne, http://epp.ph.unimelb.edu.au/epp/

[24] Grid computing prototype demonstrated in Melbourne, The Age, June 5, 2003, http://www.theage.com.au/articles/2003/06/05/1054700318561.html

[25] GD Graphics library, http://www.boutell.com/gd/