TeraGrid User Portal v1.0: Architecture, Design, and Technologies

Maytal Dahan*, Eric Roberts*, and Jay Boisseau
Texas Advanced Computing Center
The University of Texas at Austin
Austin, Texas 78758
Email: {maytal,ericrobe,boisseau}@tacc.utexas.edu
*Equal Authorship

***Abstract*— The TeraGrid [1] is a grid computing project for building a world-class comprehensive distributed infrastructure for scientific discovery. TeraGrid comprises many heterogeneous systems that enable high performance computing, data management/storage, and scientific visualization, and access to scientific data collections from facilities across the country. TeraGrid uses advanced networking systems, software technologies, and operations and support activities to tie these fundamental resources together into a production cyberinfrastructure for science and research. Allocations, accounting, security, resource monitoring, consulting, and documentation are among the services that each TeraGrid Resource Provider (RP) implements to meet their site-specific needs while operating within the TeraGrid environment. The purpose of the TeraGrid User Portal [2] is to serve as a launch pad for new users and a control panel for current users by integrating RP resources, services, and information into a single web interface serving a national community of computational researchers.**

The first version of the TeraGrid User Portal [3] addresses the most fundamental issue for integrating TeraGrid resources: extension and integration of existing, centralized TeraGrid accounting and security services with a clear and comprehensive production portal interface. In addition, the portal provides simple access to existing TeraGrid user-centric information and services such as documentation, consulting, allocation request and renewal, and resource monitoring. This paper discusses the motivation, architecture, technology, and challenges used to build an enterprise portal for the TeraGrid. It also illustrates techniques, such as load balancing, redundancy, and user auditing, that are required to ensure performance, reliability, and traceability of the portal environment.

I. Introduction

The term user portal [4] is used in the computational research community to describe a web-based interface that provides access to targeted information, resources, and services for a community of users, scientists, and engineers. A grid user portal serves as a single interface for providing this access to a set of distributed resources integrated using grid-computing technologies. Thus, the TeraGrid User Portal is designed to serve as the main interface for providing information and services to users of TeraGrid resources.

The TeraGrid User Portal v1.0 presents detailed information about the specific resources on the TeraGrid, such as machine load, job queues, user documentation, and more. Equally importantly, the portal offers an easy, consistent, and unified account interface that enables users to manage the complexity of TeraGrid accounts, allocations, and distinguished names. In section three we describe the architecture and design behind the TeraGrid User Portal. Sections four and five describe the services currently being offered in the first production version of the TeraGrid User Portal and the technologies used to achieve this functionality. Section six describes the security infrastructure and section seven discusses the challenges encountered while developing and deploying the user portal. Finally, in section eight the paper discusses the future capabilities that will be included in upcoming releases of the User Portal.

II. Architecture

The basis for the User Portal architecture revolves around the web server. We use the Apache Tomcat HTTP web server and within the web server are the individual portlet web applications [5], authentication modules, the GridSphere portal framework [6], and the JSR168 [7] portlet container, which are all discussed in more detail in section five. The web server receives requests from clients and manages resources (such as database connection pools) used to communicate with backend services.

The individual portlets each depend on backend services including the TeraGrid Central Database (TGCDB), the GridPort Information Repository (GPIR) [8] web service, the MyProxy credential management [9] server, and the TeraGrid website. Figure 1 shows a detailed architecture diagram of the TeraGrid User Portal.

The portlet web applications in the User Portal make use of a common and powerful design pattern known as Model-View-Controller [17] (MVC). The MVC paradigm breaks things up into three separate pieces: the model, the view, and the controller. The architecture is designed to separate the application (model) from the way it is represented (view) and the portlet business logic (controller). This design enables rapid development and deployment of new portlet web applications. Using MVC we have developed a wide variety of interfaces to services described in the next section.

III. Services Supported

The TeraGrid User Portal presents many different types of information and services through a single interface by

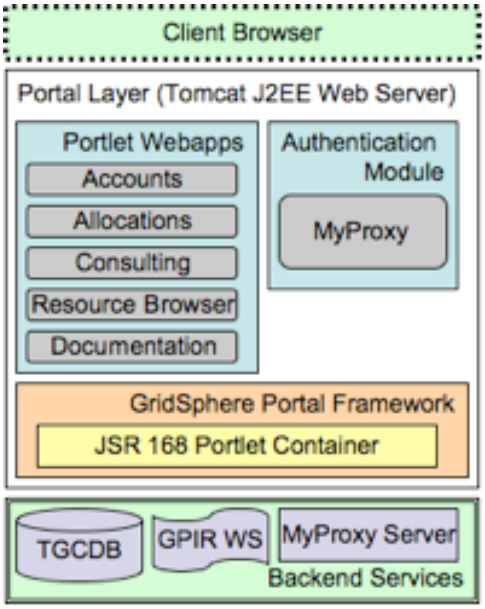

Fig. 1. TeraGrid User Portal Architecture Diagram

Fig. 2. Snapshot of production TeraGrid User Portal v1.0

using horizontal tab-based navigation. There are currently six main portal sections (tabs): Home, Resources, Documentation, Consulting, Allocations, and My TeraGrid. The Home section contains a login form, a feedback form, and information about the User Portal project. The Resources section includes a resource monitor that displays the current load, status, and job queues of all TeraGrid compute resources as well as other useful, more slow changing information about compute and visualization resources across TeraGrid. The Documentation section consumes and presents existing user documentation from the main TeraGrid website. The Allocations and Consulting sections provide information about how to apply for a TeraGrid allocation, and to request consulting support for using TeraGrid, respectively. Finally, the most personal section of the User Portal is the My TeraGrid section. It provides user-specific information to authenticated portal users about their project allocations and usage across TeraGrid, system accounts on TeraGrid resources, distinguished name (DN) registration, facilitates adding users to projects, and enables changing a users portal password. Figure 2 presents a snapshot of the TeraGrid User Portal home page showing the tabbed-navigation.

Given the large number of resources available for use in the TeraGrid, a user could have more than twenty system accounts to manage; the average user has about ten. Prior to the existence of the User Portal, a user was required to use command line interface (CLI) tools to monitor their allocation usage and generate and propagate their grid certificates. While these tools will always be useful to have in a command line environment, offering similar capabilities and presenting analogous information via a web interface have some advantages over CLI tools. One advantage is aggregating capabilities on different systems into a single interface; another is the superior formatting capability available through HTML, which allows structuring data into easy to read layouts and using aesthetic elements such as graphic images to make the information display more appealing and in some cases easier to interpret.

IV. Technologies Employed

In order to build and support the services on the TeraGrid User Portal a variety of technologies are required. The initial TeraGrid User Portal provides personalization, single sign on, and data aggregation in a consistent and comprehensive interface. This is achieved using the following technologies.

The base technology for the TeraGrid User Portal is portlets. Portlets are pluggable user interface components that are displayed and managed using a portal framework (which well describe shortly). A single portal page may have a variety of portlets displayed at one time and the same page may look different for different users. Specifically, the TeraGrid User Portal portlets are based on the JSR-168 Java Portlet specification. The specification sets a standard for interoperability of portlets between different portal frameworks and defines interfaces for security, customization, and presentation rendering. Portlets run within a portlet container, which manages each portlets lifecycle and environment. In addition to managing requests from the portal to display portlets it also stores persistent customization information per user, per portlet.

Once users access the TeraGrid User Portal using a web browser the portal framework receives the request and determines the list of portlets that are required to satisfy the request. Through the portlet container the framework invokes the required portlets and creates a single portal page composed of the HTML fragments generated by each portlet and returns that page to the browser. An example of this can be seen in Figure 3. It illustrates how two individual portlets are requested and displayed on the main My TeraGrid page.

As mentioned in the architecture section, the TeraGrid User Portal is built using the GridSphere portal framework,

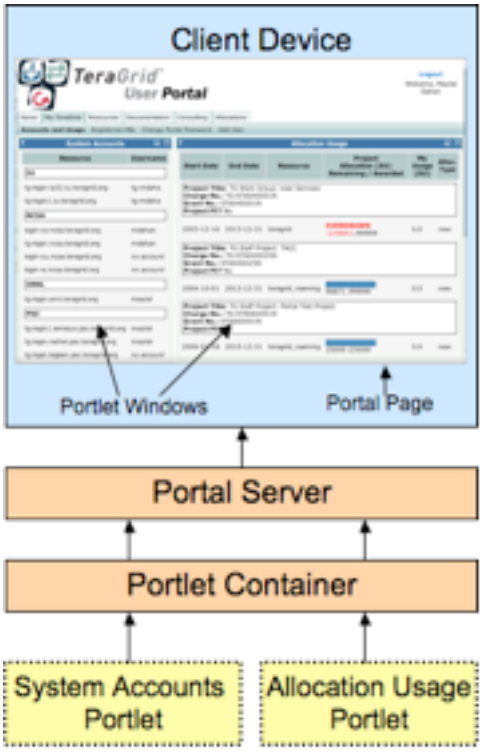

Fig. 3. My TeraGrid Portlet Page Request

which provides a JSR-168 compliant portlet container and a collection of core portal services such as login, logout, portlet layout, customization, and portlet management. GridSphere is an extensible open-source portal framework that enables us to develop and deploy TeraGrid-specific portlets.

In addition to a portal framework we needed a tool to automate building, configuring, deploying, and testing the portal. Rather than building such a system from the ground up we use an extensible, open source framework called Maven [10]. Maven provides a core set of plug-ins that handle a large set of standard software management operations including compiling java code, making java web application archives (.war files), recursive find/replace configuration, copying files, and more. However, Maven does not provide all of the functionality needed to manage the user portal. We have taken advantage of the plug-in architecture that Maven provides and built our own plug-in to manage all of the necessary portal management tasks that Maven does not provide. This plug-in allows us to easily configure and deploy the TeraGrid User Portal using simple, one-line commands and extended configuration. The plug-in enables us to overlay custom code that overrides the default behavior of GridSphere, deploy custom portlets, and greatly simplifies and centralizes portal configuration.

The initial code base for the TeraGrid User Portal used a portal toolkit known as GridPort 4 [11]. GridPort is based on the GridSphere portal framework and uses the custom maven plug-in discussed previously. GridPort enables rapid development of functional grid portlets and comprises a set of core grid portlet interfaces and services that provide a wide range of functionality. In addition to these core portlets, we have also developed many portlets specifically for the TeraGrid User Portal such as account and allocation management portlets but used GridPort as the starting framework for TeraGrid User Portal.

One interesting requirement for the User Portal involved the need to consume existing web interfaces as if they were part of the portal itself. One example is the documentation services that existed prior to the development of the User Portal. The documentation is not managed by the portal development team therefore modification and migration of documentation is beyond our control. In order to consume these external services we use a web element called an iframe, which allows external web pages to be presented in a window that is nested inside the portlet web page, making it appear as if it were part of the same page. This makes it very convenient to include external, non-portlet applications with little or no development. Therefore, we have developed our own general-purpose iframe portlet just for this purpose.

Furthermore, much of the functionality required in the portal requires database connections to the TeraGrid Central Database (TGCDB), which stores all account and allocation information for all TeraGrid users. An example of a portlet that requires database access is the system account portlet. A TeraGrid user can potentially have twenty different system accounts on TeraGrid resources, each with unique usernames. The goal of the system account portlet is to allow users to see their usernames on all systems for which they have an account. In order to retrieve the users system account information across TeraGrid the portlet needs to access the TGCDB. This is done using the Java Database Connectivity API [12], known as JDBC. JDBC enables access to relational databases through a simple interface and configuration.

Since many of the TeraGrid User Portal portlets depend on a database connection and we anticipated a large number of concurrent users accessing these portlets. We needed to consider the time and performance issues associated with instantiating a database connection. To optimize performance issues a database connection pool is utilized to manage the database connections. A connection pool is simply a pool of open database connections that are managed by a separate process in the web server. It allows applications requiring a database connection to simply ask for a connection, issue queries, and release the connection back to the pool without the instantiation overhead or destroying the connection. The connection pool can also dynamically grow and shrink based on the application demands for connections. For example, when a portlet application needs a connection it first requests an available connection from the connection pool. If one is not available (e.g. all connections are currently in use) then a new connection is created by the connection pool and given to the portlet application. A web server administrator can configure the pool with various settings to control the initial pool size, the maximum pool size, the number of available connections, and more. The specific tool being used for connection pooling

in the TeraGrid User Portal is the Apache Commons DBCP [13] package.

As you can see above, the portal requires a wide and extensive variety of technologies. All these are crucial to the development and production release of the TeraGrid User Portal v1.0. The next section describes the security issues and related security technologies required to develop the User Portal.

V. Security

One of the crucial requirements in the TeraGrid User Portal is user authentication, authorization, and account management. The TeraGrid User Portal uses a single sign-on model to secure access to many disparate services. The main authentication service is composed of many different geographically distributed services that all work together to provide single sign-on access. The services that comprise this master authentication service include Kerberos [14], a Certificate Authority [**?**], and a grid security service called MyProxy credential management service. Below we describe each of these services individually as well as how they work together.

Kerberos is a strong network authentication protocol. By not allowing passwords to be transferred over the wire, Kerberos provides a very secure form of authentication, one that is suitable for securing sensitive information for such a large distributed project as TeraGrid. Kerberos provides high availability and failover through replicated, geographically distributed service instances and thus, scales to the volume of users that we expect to eventually be part of the project.

A certificate authority generally acts as a trusted third party that issues long and short-term digital certificates. Digital certificates form the basis for public key infrastructure, which is the security infrastructure used by many pieces of grid software, such as the Globus Toolkit [16]. TeraGrid uses the Globus toolkit as a core infrastructure piece for performing grid-related tasks on distributed resources. Globus uses short-term certificates called proxies that users use to authenticate to the various grid services, including Globus.

The MyProxy online credential repository is a remote server used for storing and retrieving proxies. Normally, users possess long-term certificates and use those to generate short-term proxy certificates. If a user wished to use a short-term proxy generated on system A on system B they would normally have to manually copy that proxy from system A to system B using a file transfer tool (e.g. ftp, scp, etc). To address this problem, MyProxy allows users to delegate, or store, proxies to the server for later use elsewhere. This increases user mobility between systems, allowing them to retrieve a proxy from any machine with MyProxy client tools, which are part of the core Globus software that is installed on all production grid resources in TeraGrid. MyProxy also supports pluggable authentication modules (PAM), which allows other PAM-compliant authentication services to be used for authentication.

By combining all of these security technologies together, we have deployed a very reliable authentication service that can authenticate users and issue short-term proxy certificates on the fly. Kerberos and certificate authority software plug into MyProxy using its PAM interface, allowing user credentials (i.e. username and password) to be authenticated by Kerberos, and a certificate issued from the Certificate Authority, but all hidden behind the MyProxy service interface. The GridPort toolkit contains a MyProxy authentication module that plugs into GridSphere and therefore this service addresses our security requirements quite well. This service is also a big advantage to users because it does not require them to manage long-term certificates, an unwanted security burden on users. The portal can use this service to quickly authenticate a user and store the generated proxy in that users portal session for later authorization to other services.

VI. Releasing a Production Portal

There were many important issues to consider when releasing the first production TeraGrid User Portal. The initial concerns were ensuring that the server and services could handle the load from portal visitors. Also, it is important to be able to log not only how many user visitations to the web portal but also do more fine grain auditing of what users visited what portals and portlet functionality. This section discusses how these two objectives were completed and how they will continue to be refined throughout development of the TeraGrid user portal. In order to handle the possible multitude of visits to the user portal we have implemented load balancing for Apache Tomcat. Load balancing distributes the server load by switching to different nodes or servers based on the policy defined. When a user visits the user portal, one of the servers is chosen to execute the request. Load balancers are the single point of entry that direct traffic to various servers or clusters. The load balancing also enables to redirect load if a underlying server is down or code is being updated and pushed to production. This way the portal service is not interrupted if a single server is down.

In addition to managing user visits and balancing stress on the server we also address the issue of auditing user visits. For monitoring web server statistics a package known as AWStats is used. This monitors how many visits the portal gets a month, what operating systems are used, browser types, etc. This offers valuable insight as to what type of user we are getting, what browser and OS are most commonly used, and how many unique visits the portal is receiving. In addition to monitoring web statistics we are also implementing portlet filters. These are snipits of code that filter information about each portlet request to a log file using log4j. By mining the log files on a regular basis this will give us insight as to what components of the portal is being used most often, by what users, etc. In addition to monitoring web visit and portlet requests we are also tracking both scheduled and unscheduled downtime to ensure that the TeraGrid user portal is up and reliable and meets the expectations of both users and stakeholders.

VII. V1.0 Challenges

One of the biggest challenges involved in developing a User Portal for TeraGrid has been the complexity of integrating the

authentication and database infrastructure to support a constantly growing user community of more than four-thousand users. The portal relies on many geographically distributed services for security, documentation, systems data, and more. Tying these distributed resources together in a consistent and scalable manner has been difficult. A comprehensive plan to map all the systems and resources the user portal depends on was crucial in ensuring completeness and reliability.

Another challenge was anticipating and simulating the load and usage that the portal would encounter for a user community as large and continually growing as TeraGrid. For database queries by the interfaces that contact the TGCDB, the portal server has been configured to use database connection pooling to reduce this overhead as much as possible. Furthermore, we ran performance load tests to verify that the web servers and services offered through the portal would achieve a reasonable level of responsiveness (approximately three second maximum response time) while supporting a reasonable average concurrent usage volume (approximately one hundred concurrent users). The portal currently performs acceptably but we are always tuning different aspects and looking for performance bottlenecks to fix in order to increase the performance as much as possible. In addition to performance tuning, load-balancing techniques to distribute the request load across multiple web servers and failover systems to manage network outages or system failures have also been deployed and configured.

VIII. Future Plans

The TeraGrid User Portal v1.0 implements a significant portion of features based on user requirements, but this is just a small set of the planned features. There are many new features and improvements requested by users during the testing and deployment phase. These features include a portal consulting interface to help better direct help desk tickets; to list and integrate TeraGrid data collections and science gateways resources; to further integrate documentation and user services. In addition to expanding with new features, there are a number of existing TeraGrid services and capabilities that will be architected for integration into the User Portal. These longer-term plans include: event notification services; interactive grid services such as remote file transfer and remote job submission; interactive remote visualization, and more. These features are currently being planned, architected, and developed and will be released in subsequent versions of the TeraGrid User Portal.

IX. Conclusions

The objective of the TeraGrid User Portal is to serve the national community of computational scientists by integrating resources, services, and information into a single web interface. Developing this comprehensive interface requires a complex architecture and a wide variety of technologies as described above. While the challenges for developing the initial version were significant, it has been clearly apparent from user response that the portal is providing great benefits for users navigating the heterogeneous TeraGrid resources. As additional services get integrated into the portal, the value and impact will continue to increase and should become an indispensable resource for all TeraGrid users.

References

[1] TeraGrid, URL: http://www.teragrid.org
[2] E. Roberts, M. Dahan, J. Boisseau. TeraGrid User Portal: An Integrated Interface for TeraGrid User Information and Services TeraGrid 06.
[3] TeraGrid User Portal, URL: http://portal.teragrid.org
[4] M. Thomas, S. Mock, J. R. Boisseau. Development of Web Toolkits for Computational Science Portals: The NPACI HotPage. HPDC 2000: 308-309
[5] Portlet Definition, URL: http://en.wikipedia.org/wiki/Portlets
[6] J. Novotny, M. Russell, O. Wehrens GridSphere: An Advanced Portal Framework GridSphere Project Website, URL: http://www.gridsphere.org
[7] JSR-168 Portlet Specification, URL: http://jcp.org/en/jsr/detail?id=168
[8] GridPort Information Repository (GPIR) Web Service, URL:
[9] J. Basney, M. Humphrey, V. Welch. The MyProxy Online Credential Repository. Software: Practice and Experience, Volume 35, Issue 9, July 2005: 801-816
[10] Apache Maven Project, URL: http://maven.apache.org/
[11] The GridPort 4 portal toolkit, URL: http://gridport.net
[12] Java Database Connectivity API, URL: http://java.sun.com/javase/technologies/database.jsp
[13] Apache Commons DBCP, URL: http://jakarta.apache.org/commons/dbcp/
[14] Kerberos, URL: http://web.mit.edu/kerberos
[15] Definition of Certificate Authorities, URL:
[16] Globus Toolkit, URL: http://www.globus.org
[17] Description of the Model-View-Controller, URL: http://en.wikipedia.org/wiki/Model-view-controller/