SimpleGrid Toolkit: Enabling Efficient Learning and Development of TeraGrid Science Gateway

Shaowen Wang Yan Liu
CyberInfrastructure and Geospatial Information Laboratory (CIGI)
National Center for Supercomputing Applications (NCSA)
University of Illinois at Urbana-Champaign
1-217-333-7608
shaowen@uiuc.edu, yanliu@uiuc.edu

Nancy Wilkins-Diehr
San Diego Supercomputer Center (SDSC)
University of California at San Diego
wilkinsn@sdsc.edu

Stuart Martin
Argonne National Laboratory
University of Chicago
smartin@mcs.anl.gov

ABSTRACT

Science gateways have become an important cyberenvironment modality to bridge domain applications and cyberinfrastructure, which is fundamental to make cyberinfrastructure integral and effective to the practice of advancing science and engineering. However, it remains a significant challenge to learn science gateway technologies and develop a science gateway because science gateway technologies are still actively evolving and often include a large number of sophisticated components. Some of these components must be designed to fit within the legacy methods through which scientists from a particular domain use conventional computation tools. This paper establishes a generic toolkit – SimpleGrid for learning and developing science gateways based on a service-oriented architecture using a component-based approach that allows flexible separation and integration of the components between domain applications and cyberinfrastructure in the process of learning and development of science gateways. The paper presents the design and implementation of SimpleGrid and illustrates our experience of using SimpleGrid in a tutorial to teach TeraGrid science gateway based on a well-known application (spatial interpolation) in the field of geographic information science.

Categories and Subject Descriptors

D.3.3 [**Programming Languages**]: Language Contructs and Features – *abstract data types, polymorphism, control structures.*

General Terms

Management, Design, Security, Human Factors, Standardization

Keywords

problem solving environments (PSE), science gateway, component-based software engineering, service-oriented architecture

Shaowen Wang, Yan Liu, Nancy Wilkins-Diehr, Stuart Martin, SimpleGrid Toolkit: Enabling Efficient Learning and Development of TeraGrid Science Gateway, International Workshop on Grid Computing Environments 2007, Nov. 11, 2007, Reno, NV, USA. An electronic version of this document will be available at http://casci.rit.edu/proceedings/gce2007

1. INTRODUCTION

The demand for high-level, easy to use, and powerful problem solving systems (PSE) for science has been recognized for decades [1]. Inadequate computing power, however, made such systems infeasible until the 1980s when serious work began [2]. Grid computing environments have been developed as a distributed computing infrastructure for coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations (VO) [3]. Consequently, for computationally-intensive applications, a Grid-based PSE can be viewed as the environment through which end-users exploit available Grid resources. The concept of a science gateway is a community-specific set of tools, applications, and data collections that are integrated together via a portal or a suite of applications, providing access to Grid-integrated resources [4]. The science gateway concept can be deemed as analogous to Grid-based problem solving environments, particularly from the point view of domain scientists.

Grid portal technologies have been used to manage the complexity of using distributed resources owned by collaborating users in Grid-based problem solving environments. A Grid portal is designed to provide users with a point of access to Grids. It is also defined as a type of PSE where a user can access Grid resources and services, execute and monitor Grid applications, and collaborate with other users [5]. Consequently, Grid portals are emerging as an important way to exploit and manage Grids when they are used to address scientific problems.

One challenge to developing Grid portals for science gateways lies in the necessity of using a number of evolving and sophisticated web portal technologies. Several evolving portal technologies are widely used, including GridSphere, Liferay, Sakai, and Jetspeed. Though these portal technologies are being developed to conform to standards such as JSR-168, different technologies have their specific implementations as well as strengths and weaknesses that developers must understand to build science gateways. Another challenge is to design and develop robust and reusable interfaces between applications and portals. Our research aims to address both challenges by establishing a generic toolkit – SimpleGrid that provides reusable software components [6] to encapsulate portal technologies and generalize the interfaces among applications, portals, and Grid middleware.

The design of SimpleGrid is consistent with service-oriented architecture within a component-based framework [7] and fully leverages the Java Commodity Grid (CoG) Kits that are based on the Globus Toolkit 4. We use the TeraGrid – a key element of the U.S. and world cyberinfrastructure as a cyberinfrastructure resource and service provider in SimpleGrid development and evaluation. Specifically, SimpleGrid is designed to provide domain scientists a user-friendly interface, allow scalable integration of Grid services, support standard-based portlet development, and enable streamlined access to cyberinfrastructure based on an easy-to-use component-based framework. SimpleGrid is sandwiched between cyberinfrastructure and portal-level capabilities. The SimpleGrid toolkit includes four basic components: SimpleCred for user credential management, SimpleRun for job execution support; SimpleTran for data transfer, and SimpleViz for visualization (Figure 1). These components are designed to be lightweight in terms of their programmability and extensibility to support efficient learning and development of science gateways.

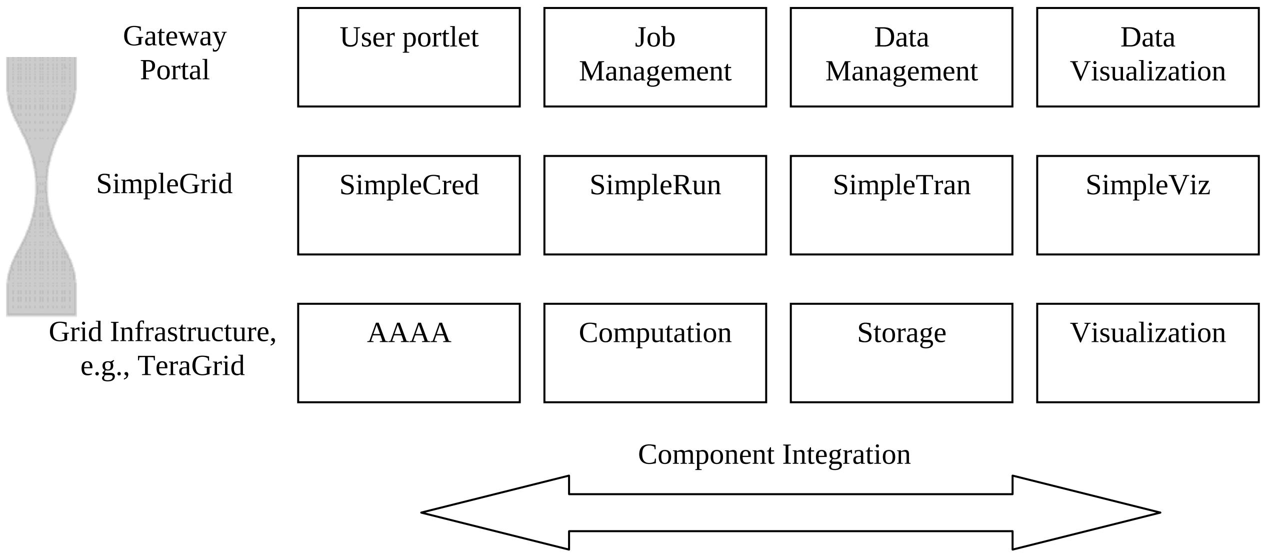


Figure 1. SimpleGrid components

In the remainder of this paper, section 2 describes the motivation and targeted use of the SimpleGrid toolkit. Section 3 presents SimpleGrid architecture. Section 4 illustrates the implementation of each SimpleGrid toolkit component. Section 5 introduces an application called spatial interpolation in geographic information science (GIScience) that is used to demonstrate how SimpleGrid interfaces with domain applications. Finally, section 6 draws several conclusions, based on which some future research directions are summarized.

2. MOTIVATION

SimpleGrid Toolkit is designed to lower the barriers and costs for developing and learning science gateways. Currently, the barriers and costs are high mainly because:

- there is a significant gap between Grid technologies and application problem solving environments;
- Grid and web portal technologies are complex, and still rapidly evolving;
- Grid technologies have been developed to focus on enabling resource sharing and federation whereas the development of problem solving environments requires extensible, programmable, reusable, application-oriented software components that support customizable access to Grid and VO capabilities.

SimpleGrid's application-oriented components provide a set of APIs that allow science gateway developers to concentrate on developing problem solving environments tailored to the needs and working styles of scientists. SimpleGrid toolkit integrates the most recent advancements in existing generic Grid middleware and portal technologies. Such integration facilitates science gateway development and learning. Through the use of SimpleGrid, science gateway developers benefit from avoiding the overhead of managing changing portal technologies and controlling the details of Grid and VO capabilities such as Grid credential management and job execution.

3. ARCHITECTURE

SimpleGrid architecture includes four major components that provide external interfaces to science gateway environments (Figure 2-A).

- SimpleCred provides the capabilities to manage user and VO authentication and authorization and to support auditing and accounting. Myproxy is integrated within SimpleCred while GridShib is being investigated to support attribute-based authorization.
- SimpleRun automates the execution through a workflow system that is developed to interface with both Web services Globus GRAM (Grid Resource Allocation and Management) and Pre-Web service GRAM. The capabilities of auditing and recovering computation jobs can be integrated within SimpleRun.

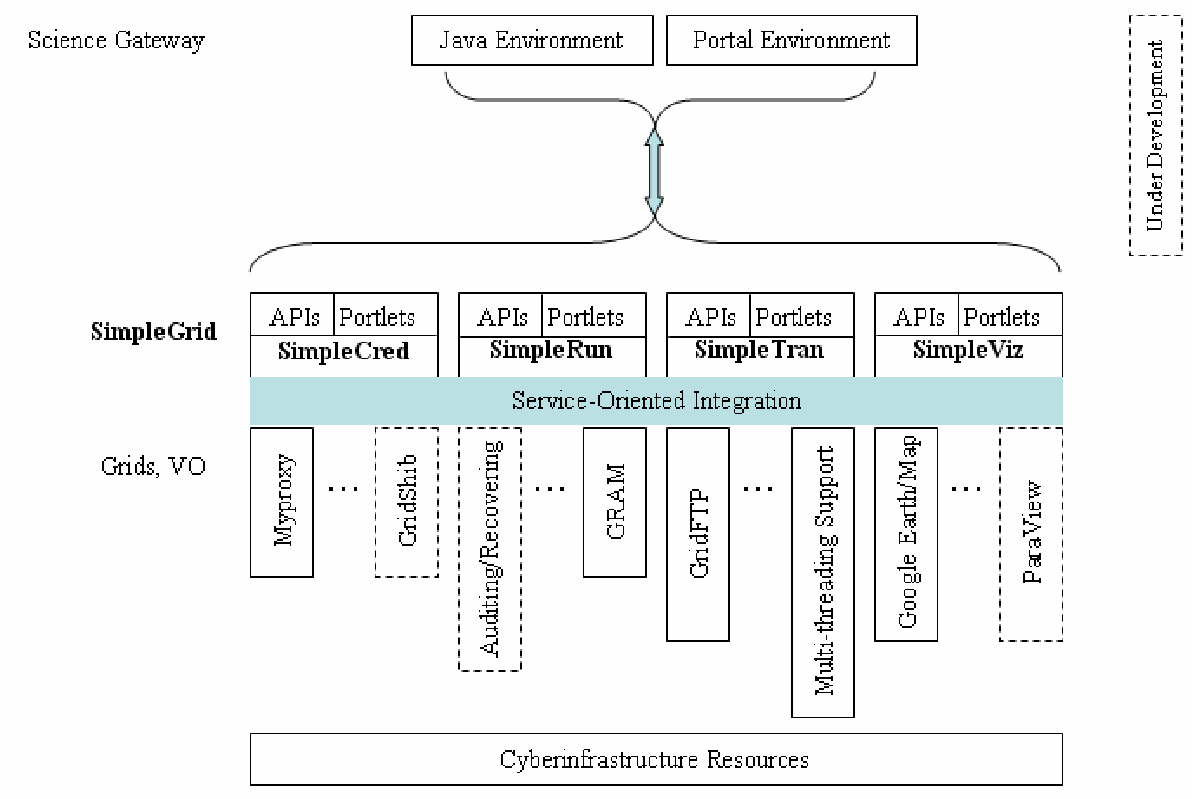


Figure 2-A. SimpleGrid architecture – external interfaces

- SimpleTran allows flexible file management that is implemented by leveraging GridFTP with multi-threaded support. The multithreading support enables users to initiate multiple non-blocking GridFTP transfers.
- SimpleViz customizes and integrates several visualization tools such as Google Earth/Map and ParaView [8].

These four components have both API and portlet interfaces to science gateway applications. For learning how to develop science gateways, portlet interfaces would be sufficient while Java APIs provide more flexibility to specify the capabilities of accessing cyberinfrastructure resources in science gateway development.

SimpleInfo is an internal component serving as the information system of SimpleGrid, which does not directly provide any external interface to client applications. SimpleInfo uses the Modular Information Provider [9] to gather and manage information from the aforementioned four components. SimpleGrid can interface with Grid information services such as Globus MDS-4 by leveraging the capabilities of the Modular Information Provider. SimpleRun, SimpleTran, and SimpleViz rely on SimpleInfo to discover information about the status of computation, data transfer, and visualization using credentials from SimpleCred (Figure 2-B). Interactions among SimpleRun, SimpleTran, and SimpleViz may take place depending on the input-output requirements of a particular workflow.

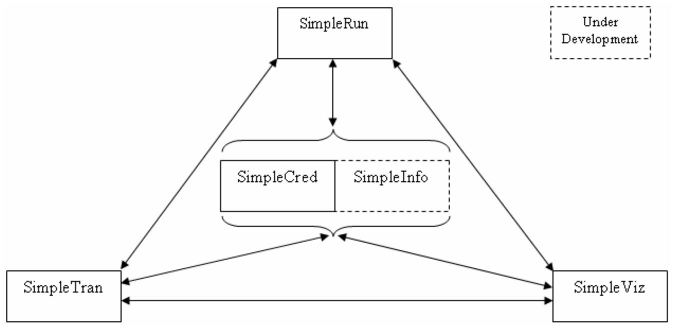


Figure 2-B. SimpleGrid architecture – internal interactions

4. IMPLEMENTATION

SimpleGrid toolkit serves the purpose of supporting both learning and development. As a learning environment SimpleGrid shortens the learning curve of comprehending science gateway technologies; as a development environment, SimpleGrid simplifies science gateway development process. The toolkit uses Globus Toolkit for Grid access capabilities, leverages Java CoG kit as a Grid access programming interface, and employs GridSphere as a gateway portal server. The accommodation of various Grid services and web portal technologies is accomplished using a service-oriented approach (Figure 2-A). A set of APIs and reusable portlet components are developed based on similar principles and technologies employed in the development of GISolve – a TeraGrid GIScience Gateway [10].

4.1 Efficient Learning and Development

Because science gateway developers are faced with significant complexity of technologies, spanning from understanding Grid middleware, programming support for Grid access, to web services and interface development. SimpleGrid implementation aims to be “simple”.

The learning of Grid middleware and web portal technologies using SimpleGrid is contextualized in prototyping a TeraGrid science gateway using a representative scientific computing application. Through such a concrete experience, developers can easily switch to production gateway development by following the principles learned and reusing SimpleGrid components. The exercise includes learning Grid credential management, data transfer, job submission, and visualization. The learning process is carried out in three stages: command-line, Grid-enabled java application development, and portal development. Each stage fulfills the same requirements of using cyberinfrastructure resources for scientific computing tasks. Command-line uses Globus commands; Grid-enabled application uses SimpleGrid Java APIs; and portal development uses both SimpleGrid Java APIs and portlet components and interfaces.

SimpleGrid Toolkit is packaged to enable simple deployment through *ant*. Currently, SimpleGrid toolkit uses GridSphere, Tomcat, and Globus Toolkit as external supporting environments. Instructions are also available to setup Eclipse environment for efficient web application debugging.

Currently, Grid and portal technologies integrated within SimpleGrid Toolkit include:

- Grid security: Grid certificate, Globus-based authentication, MyProxy;
- Grid data management: GridFTP;
- Grid job submission and management: GRAM, WS-GRAM;
- Grid-enabled application development: Java CoG Kit, Globus APIs;
- Science gateway development: GridSphere portlet container, JSR-168 java Portlet APIs;
- Portlet rendering: GridSphere Visual User Interface / JSP, Velocity.

4.2 SimpleGrid APIs

SimpleGrid APIs enable automating the access to cyberinfrastructure for a large number of scientific computing tasks and provide a programming interface for domain scientists who are generally not interested in all the details of Grid and VO technologies. SimpleGrid APIs are tailored to web portal contexts, and can be directly reused to learn and compose science gateways. Through the APIs, the complexity of managing Grid capabilities is hidden while application-oriented interfaces are presented. Combined with portlet components that provide object storage, SimpleGrid APIs allow developers to compose application workflows, and support Grid-services via Globus WS-Core and/or Apache Axis. WS-GRAM, Globus auditing, and GISolve Grid services are such examples. SimpleInfo supports Grid resource discovery and scheduling, and is under development. We describe several example methods of SimpleCred APIs and overview API implementations of other components as follows.

4.2.1 SimpleCred

SimpleCred APIs have several methods to instantiate and manage a Grid proxy.

- *load():* Load a local valid proxy file via Globus APIs (equivalent to globus-proxy-init command);
- *logon()*: Fetch the proxy file by contacting a MyProxy server defined by SimpleGrid configuration (equivalent to myproxy-logon command).
- *get()* method tries to load() first. If it failed, *logon()* method is called. The result of *get()* is a standard GSSCredential object instance stored as a variable. The *export()* method saves a GSSCredential instance to a file for later retrieval. Using *get()* method, gateway developers can easily implement an automatic proxy renewal function in their science gateways. Therefore, little effort is required in Grid credential management.

SimpleCred functions are also made available through a portlet interface based on GridSphere (Figure 3). This user interface used a template to develop attribute-based authorization component based on the GridShib [11]. In the case that a gateway uses a community account, SimpleGrid supports an instantiated SimpleCred object that can be shared among all portal users.

4.2.2 SimpleTran

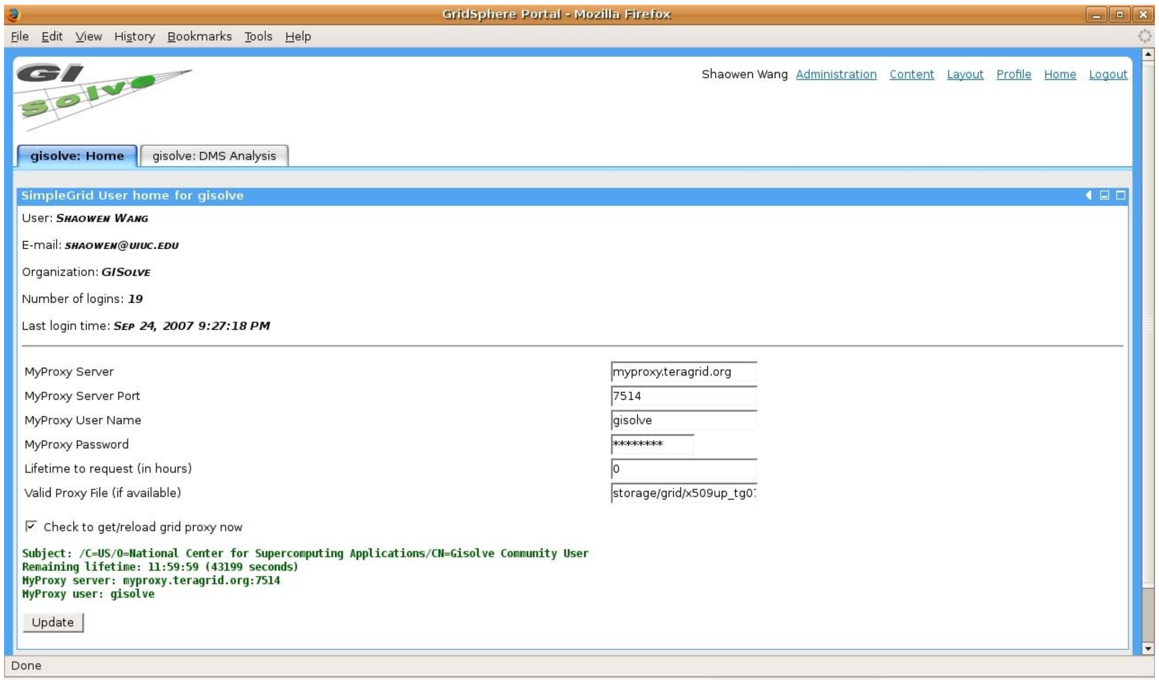
This set of APIs are built directly on Java CoG kit by adding multi-threaded support with which portal servers are able to return immediately after starting GridFTP transfer instances. Therefore gateway users experience responsive transfer actions. SimpleTran authenticates control and/or data channels with a SimpleCred instance, and is used to transfer input and output datasets and visualization results between cyberinfrastructure resources.

4.2.3 SimpleRun

SimpleRun APIs support both Pre-WS GRAM and WS GRAM with a unified interface method execute(). Jobs are submitted using a batch mode to get immediate return with a job handle from execution threads. getStatus() method takes a job handle as its argument and retrieves job status. Job description file is application-specific. SimpleRun depends on SimpleTran for input and output dataset transfers. An application computation process is uniquely identified and recorded based on user and job information. Each user has its own space at portal side for dataset uploading, result downloading, and intermediate information storage. Such user-specific information management is particularly useful for remote Grid resources in their management, if a gateway uses a Grid community account.

4.2.4 SimpleViz

Currently, SimpleViz provides a local implementation via JFreeChart and Google Earth/Map based visualization. This local visualization is threaded. For large-scale visualization requiring intensive data and computing, Grid-based visualization technologies, e.g., ParaView on TeraGrid, are being integrated.


GridSphere Portal - Mozilla Firefox
File Edit View History Bookmarks Tools Help
GI Solve
Shaowen Wang Administration Content Layout Profile Home Logout
gisolve: Home
gisolve: DMS Analysis
SimpleGrid User home for gisolve
Update
Done

User: Shaowen Wang

E-mail: shaowen@uiuc.edu

Organization: GISolve

Number of logins: 19

Last login time: Sep 24, 2007 9:27:18 PM

MyProxy Server	myproxy.teragrid.org
MyProxy Server Port	7514
MyProxy User Name	gisolve
MyProxy Password	********
Lifetime to request (in hours)	0
Valid Proxy File (if available)	storage/grid/x509up_tg0

[✓] Check to get/reload grid proxy now

```
Subject: /C=US/O=National Center for Supercomputing Applications/CN=Gisolve Community User
Remaining lifetime: 11:59:59 (43199 seconds)
MyProxy server: myproxy.teragrid.org:7514
MyProxy user: gisolve
```

Figure 3. A user portlet based on SimpleCred

4.3 Portlet Components and Interfaces

SimpleGrid adopts standard-based portal development, i.e., JSR-168 Portlet APIs. These APIs define a set of specifications to make portlets deployable to different portlet containers. Portlet development in SimpleGrid complies with the Portlet APIs standard. Therefore, Porlets in SimpleGrid can be deployed into different portlet containers.

However, conventions for inter-portlet communication, object persistence, and response rendering are not defined in portlet APIs, which means that multiple technology choices can be made. SimpleGrid is not bound with any single technology. Instead, it provides alternative technologies based on different development interests. Take response rendering for example, GridSphere Visual UI is a JSP-based portlet interface technology tightly coupled with GridSphere; Velocity portlets separate code and markup languages following the Model-View-Control mechanism; AJAX achieves efficient communications between portal and browser. To support them and other technologies (e.g., JavaServer Face (JSF) and Web 2.0), SimpleGrid implements a pluggable component-based framework to link services between Portlet APIs and specific technologies. SimpleGrid keeps and updates the state information of application-level workflow, portal page rendering, and computing and data transfer status online in portlet session memory and offline in databases.

5. EXPERIENCE OF USING SIMPLEGRID

In this section, we describe our experience of using SimpleGrid to teach a hands-on tutorial of developing TeraGrid science gateways based on a spatial interpolation analysis in GIScience. This hands-on tutorial was presented as part of the "Building Blocks for Science Gateways Tutorial[1]" in the TeraGrid'07 conference.

5.1 Spatial Interpolation Analysis on TeraGrid

Two-dimensional spatial interpolation is widely used in GIScience to predict the trends of environmental and social variables. Spatial interpolation often involves a nearest-neighbor search procedure that is computing intensive for large spatial datasets and/or high-resolution interpolation. In the tutorial, we used a fast two-dimensional spatial interpolation algorithm called DMS (Dynamically Memorized Search). A real-world spatial data analysis based on this algorithm can be viewed as a parameter-sweeping application.

The overall process of performing a DMS analysis on TeraGrid in the SimpleGrid hands-on tutorial is summarized as follows.

1. Request an individual or community account on TeraGrid. In this tutorial, TeraGrid training accounts were used;
2. Install DMS executables on three TeraGrid sites: UC (University of Chicago), NCSA (National Center for Supercomputing Applications), and SDSC (San Diego Supercomputing Center);
3. Prepare a dataset on a local machine. SimpleGrid package provides a sample dataset in

[1]http://www.teragridforum.org/mediawiki/index.php?title=TG07_Gateway_Tutorial

${SIMPLEGRID_DIR}/simplegrid/webapp/storage/samples/;

4. Transfer a specified dataset to a TeraGrid site (e.g., NCSA);
5. Submit a Grid job to the specified TeraGrid site with a parameter value;
6. The submitted job is scheduled to be executed on one compute node on the specified TeraGrid cluster;
7. When the job is finished, the analysis result is written into the data directory of DMS installation on the TeraGrid cluster;
8. Transfer the result back to the local machine; and
9. Visualize the result using the DMS visualization tool which is a Java-based application in ${SIMPLEGRID_DIR}/simplegrid/lib/viz/.

This process was demonstrated in three ways: TeraGrid command-line tools for individual use, SimpleGrid APIs to automate the access to cyberinfrastructure resources, and a SimpleGrid portlet to enable community access to the analysis using GridSphere.

5.2 DMS Analysis Portlet

A DMS portlet was created to provide a web interface for performing DMS analysis. In this portlet, a DMS analysis process goes through the selection of a dataset and parameter, job creation, dataset transfer, job execution on TeraGrid, result transfer, and local visualization. A "*next*" button and text area are used to initiate each stage and show the status of states respectively. For job execution and data transfer, DMS portlet uses the loaded proxy from the user portlet (Figure 4) via the inter-portlet communication mechanisms that are described in section 4.3.

To illustrate alternative technologies that can be used to develop portlets that are compliant with the JSR-168 standard, the User portlet employs GridSphere-based method, i.e., *ActionPortlet* and JSP-based Visual UI while the DMS portlet employs Apache Velocity-based method.

5.3 Summary of the Hands-on Tutorial Experience

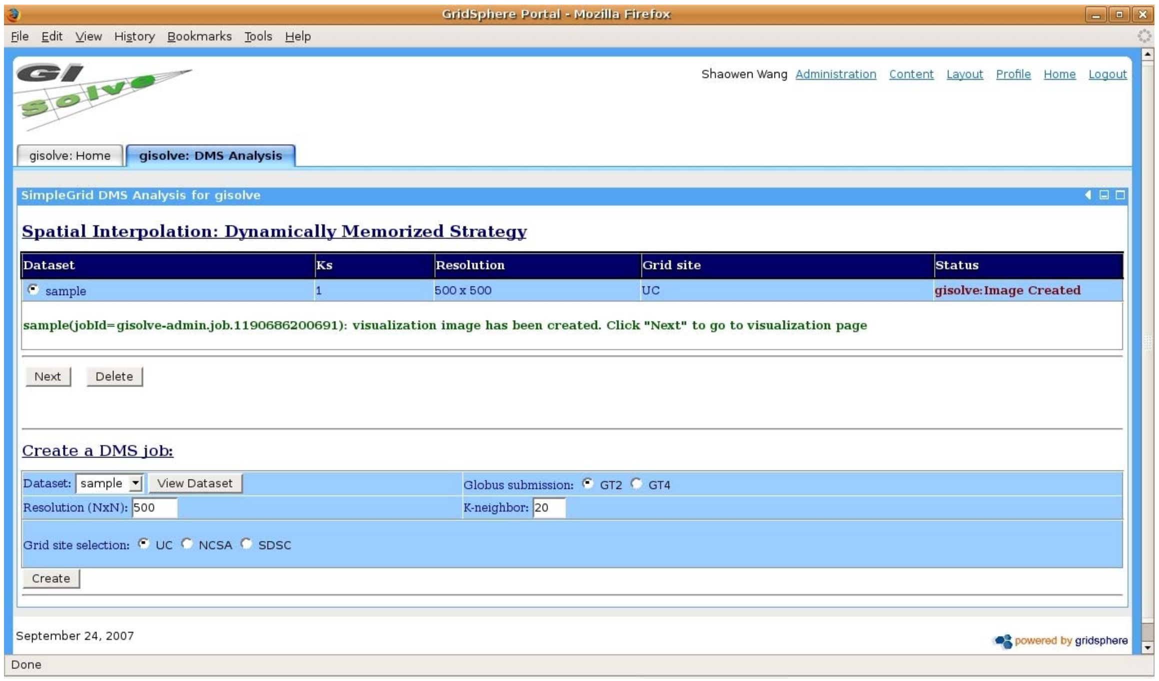
The hands-on tutorial had 16 participants with various levels of software development experience and Grid computing knowledge. After two and half hours, all participants including those with minimum Java programming knowledge were able to master the SimpleGrid APIs for the DMS analysis, and to successfully set up a portlet for the analysis in a GridSphere portal server.

6. CONCLUDING DISCUSSION

The SimpleGrid toolkit makes an abstraction of generic Grid middleware services and hides the complexity of evolving web portal technologies by tailoring to application requirements for developing PSE. This abstraction enables science gateway developers to concentrate on developing PSE by working on reusable and extensible software components. SimpleGrid Java APIs and portlet support simplify science gateway development

and help overcome the learning curve of science gateway technologies.

The SimpleGrid architecture complies with service-oriented architecture within a component-based framework, and thus is open and extensible to integrate evolving Grid middleware


GridSphere Portal - Mozilla Firefox
File Edit View History Bookmarks Tools Help
GI Solve
Next Delete
View Dataset
Create
powered by gridsphere
Done

Shaowen Wang Administration Content Layout Profile Home Logout

gisolve: Home | gisolve: DMS Analysis

SimpleGrid DMS Analysis for gisolve

Spatial Interpolation: Dynamically Memorized Strategy

Dataset	Ks	Resolution	Grid site	Status
● sample	1	500 x 500	UC	gisolve:Image Created

sample(jobId=gisolve-admin.job.1190686200691): visualization image has been created. Click "Next" to go to visualization page

Create a DMS job:

Dataset: sample

Globus submission: [✓ GT2] GT4

Resolution (NxN): 500

K-neighbor: 20

Grid site selection: [✓ UC] NCSA SDSC

September 24, 2007

Figure 4. Spatial interpolation portlet

services such as the RFT (Reliable File Transfer) service from the Globus Toolkit. SimpleGrid toolkit paves the way to rapidly prototype science gateways, and to lower the barriers to leverage comprehensive Grid environment technologies such as the OGCE (Open Grid Computing Environments) [12].

We used SimpleGrid to give a hands-on tutorial on TeraGrid science gateway to a group of developers with various levels of knowledge about programming and Grid computing. All developers were able to successfully prototype a science gateway using a well-known application in GIScience during the two and half hours hands-on tutorial. The tutorial participants were also able to understand the basic, but essential components needed to build a TeraGrid science gateway and learned a scalable way to develop and share their applications in a gateway portal. The code in SimpleGrid is component-based, and thus can be directly extracted for a generic science gateway development. Related technologies are carefully referenced in SimpleGrid so that gateway developers could directly use sophisticated capabilities that are made available at the levels of generic Grid services and various portal technologies.

Future work will integrate additional generic Grid services such as those for scheduling and performance prediction of job execution and data transfers; continue to enhance the SimpleInfo component; develop a generic and simple workflow system that tie SimpleGrid components together; and investigate ways of recording and managing data provenance based on the interactions of SimpleRun, SimpleTran, and SimpleViz components.

7. ACKNOWLEDGMENTS

This research was supported in part by the National Science Foundation through the TeraGrid Grid Infrastructure Group project award # 0503697 and a computation resource allocation award: TG-SES070007N.

8. REFERENCES

[1] Culler, G., and Fried, B. 1963. An on-line computing center for scientific problems. IEEE Pacific Computer Conference (1963), 221.

[2] Rice, J., and Boisvert, R. 1985. Solving Elliptic Problems Using ELLPACK. Springer-Verlag, New York.

[3] Foster, I., Kesselman, C., and Tuecke, S. 2001. The anatomy of the Grid: enabling scalable virtual organizations. International Journal Supercomputer Applications (2001), 15(3).

[4] Wilkins-Diehr, N. 2007. Science Gateways – Common Community Interfaces to Grid Resources. Concurrency and Computation: Practice and Experience (2007), 19(6), 743-749.

[5] Fox, G. C., Gannon, D., and Thomas, M. 2003. Overview of Grid computing environments. Grid Computing, Making the Global Infrastructure a Reality (2003), edited by R. Berman, G. Fox, and T. Hey, John Wiley & Sons, West Sussex, England.

[6] The Common Component Architecture Forum (2007). http://www.cca-forum.org/ (September 2007).

[7] Schmidt, R., Benkner, S., Brandic, I., and Engelbrecht, G. 2007. Component oriented application construction for a Web service-based Grid. Concurrency and Computation: Practice and Experience (2007), 19(5), 637-650.

[8] ParaView. http://www.paraview.org/New/index.html.

[9] Wang, S., Shook, E., Padmanabhan, A., Briggs, R., Pearlman, L. 2006. Developing a modular information provider to support interoperable Grid information services. In the Proceedings of Grid and Cooperative Computing - GCC 2006: The Fifth International Conference (2006), IEEE Computer Society, 448-453.

[10] GISolve, TeraGrid GIScience Gateway. 2007. http://www.gisolve.org (September 2007).

[11] GridShib. 2007. http://gridshib.globus.org/ (September 2007).

[12] OGCE. 2007. http://www.collab-ogce.org/ogce2/ (September 2007).