Using an End-To-End Demonstration in an Undergraduate Grid Computing Course

Mark A. Holliday
Dept. of Mathematics and Computer Science
Western Carolina University
Cullowhee, NC 28723
01-828-227-3951
holliday@cs.wcu.edu

Barry Wilkinson
Dept. of Computer Science
Univ. of North Carolina at Charlotte
Charlotte, NC 28223
01-704-687-8381
abw@uncc.edu

James Ruff
Dept. of Mathematics and Computer Science
Western Carolina University
Cullowhee, NC 28723
01-828-227-7245
jruff@cs.wcu.edu

ABSTRACT

In this paper, we describe a demonstration called an "End-to-End" demonstration developed for the 2005 offering of our grid computing course that was taught across the State of North Carolina in 2004 and 2005. This course was broadcast across the State of North Carolina using NCREN (North Carolina Research and Education Network) a televideo network. Eight universities and colleges participated in the course in 2004, which grew to 12 universities and colleges in 2005. We will briefly outline the course as the background information to the demonstration. We created the End-to-End demonstration to bring together the aspects of grid computing that are taught in the course.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems – *client-server, distributed applications, distributed databases, network operating systems.*

General Terms

Design, Experimentation, Security, Standardization.

Keywords

Grid Computing, Computer Science Education, Globus, Stateful Web Services, Sun Grid Engine, Load Sharing Facility.

1. INTRODUCTION

Grids are continuing to develop into one of the main forms of wide-area distributed computing. The academic community has developed courses on the topic at both the undergraduate and graduate levels. For the last two years, we have been developing and testing an undergraduate course for junior and senior level computer science and computer engineering majors.

In this paper, we report on our experience teaching the course across the State of North Carolina first in 2004 and again in 2005, with the focus on a specific demonstration developed for the 2005 offering. We took advantage of the state-wide televideo network called NCREN (North Carolina Research and Education Network) that links all 16 state institutions and a variety of private universities and research sites. The locations of the 16 state campuses are shown in Figure 1.

ACM SE'06, March, 10-12, 2006, Melbourne, Florida, USA

Figure 1. A map of the 16 state campuses of the University of North Carolina University system.

When the course was first taught (Fall 2004), eight institutions participated: Appalachian State University, Elon University, NC State University, University of North Carolina at Asheville, University of North Carolina at Greensboro, University of North Carolina at Wilmington, NC Central University, and Western Carolina University. Most lectures were presented from Western Carolina University, but some were presented from the University of North Carolina at Wilmington. Guest speakers gave presentations from MCNC (Microelectronics Center of North Carolina) and elsewhere. These guest speakers included internationally known grid computing experts Professor Dan Reed and Dr. Wolfgang Gentzsch. Professor Ian Foster also gave a taped presentation. These presentations exposed students to some of the best minds in the field. The course was organized around a series of grid computing programming assignments [4, 10].

We offered the course for a second time in Fall 2005 (on-going at the time of writing), and the number of participating sites increased to 12. New sites were: University of North Carolina, Chapel Hill, University of North Carolina at Charlotte, University of North Carolina at Pembroke, Winston-Salem State University, Lenoir Rhyne College, and Wake Technical Community College. This time, most lectures were presented from the University of North Carolina at Charlotte, but some were presented from University of North Carolina at Wilmington and Western Carolina University. We made changes based upon our experiences from the first offering. Significant changes were also made to the specific software used. In addition, we established collaborating Certificate Authorities across the participating sites to provide security. This major re-design of the course is reported here, notably the end-to-end (E2E) demonstration used to solidify students' understanding of the grid computing components.

Just as the course content is structured around the series of programming assignments, the programming assignments are structured around the end-to-end demonstration. In particular, the end-to-end demonstration is an example application illustrating the key components of a grid application. Early in the semester, we give a live demonstration of the end-to-end application, discuss its implementation in detail (including much of the code), and show the mapping between the end-to-end application and the course assignments.

The complexity of grids makes it is easy to lose sight of the overall picture. Upon reviewing our first offering of the course, we concluded that we should take further steps early in the semester to make clear the overall nature of grid computing and how the individual assignments and course topics fit into that overall framework. We choose to demonstrate a complete, but as simple as possible, example of a use of a grid. By "complete" we mean an application that starts at the typical front-end that the user would access all the way to a typical back-end that provides computational and data resources.

We feel that the end-to-end demonstration and how it integrates with and enhances the rest of the course is noteworthy and should be considered by other instructors developing grid computing courses. The end-to-end demonstration was designed to reinforce the course assignments by using the same grid features as the assignments. The end-to-end demonstration was also designed to balance two competing goals: the goal of being a realistic, complete application and the goal of being simple enough that it can be understood by students early in the semester and explained in its entirety.

This paper describes the end-to-end application, how it achieves the above goals, and how it provides a framework for the assignments, and indirectly, for the entire course. The next section of this paper provides an overview of the course assignments and the course in general. The third section describes the end-to-end application. In the fourth section, we discuss the relationship of the end-to-end application to the course assignments. In the last section, we discuss our experiences and then we conclude.

2. BASIC COURSE STRUCTURE

Though there is some proprietary grid software, the grid environment is one in which open source software based on open standards is the predominant case. In general, the operating system used is Linux, although some work can be done by students on a Windows system. Java is the main programming language. The principal grid computing package we use is from the Globus project. The Globus Toolkit is the reference implementation for the core grid software [5]. Related packages come from the Apache project for web services, which we consider first in the course. Software distributions, most notably the National Science Foundation Middleware Initiative (NMI) distribution [6], serve as a central location for many grid computing packages that provide all the functionality needed in a grid.

Consequently, our course uses this combination of software (Java, Linux, Apache projects, the Globus Toolkit, and the NMI distribution) for our grid and explains the grid-specific features of the software, specifically some of the software in the Globus Toolkit. For the first course offering, we used the Globus Toolkit 3.2. This year Globus Toolkit 4.0 became available and we redid all of the software to include it. This was a significant undertaking because version 4.0 completes the integration of grid services and web services. Because of the complexity and rapid evolution of the software, it was necessary for us to develop a detailed installation guide. This guide was used by the faculty at each of the five sites that have clusters with the grid software installed. The resulting consistency across the clusters ensured the assignments worked the same at each site.

We chose to use a bottom-up approach for the course organization. Unlike other courses (such as those developed by the GridForce project [7]), the assignments and lectures start with the lowest level concepts (i.e. web services) and build up to higher levels of software. Each assignment has two halves. In the first half the student is guided through the steps of running a version of the software being demonstrated. In the second half the student modifies or extends the provided client and server software and then repeats the steps. The lectures for the course cover the material needed for each assignment. The lectures also include introductory material motivating the use of grids and providing a historical context.

2.1 Assignment 1: Web Services

A key feature of grid software is that a grid is implemented using web services, which are based on a set of well-defined standards. A web service is often implemented using a Java servlet, which in turn requires the presence of a servlet container. Our Assignment 1 is based upon [1] and provides the Apache Tomcat servlet container, and some of the source files for an example web service. The student uses Apache Axis to generate the remaining source files including the Web Services Definition Language (WSDL) file for the service. WSDL is explained. The source file for a client of the web service is provided and reviewed. The students compile both the client and the server sources, deploy the service, and execute the client and the service. The students then have to extend the functionality of the web service, extend the client to test the new functionality and then repeat the previous steps.

2.2 Assignment 2: Web Services with Resources

Grids use a version of web services that implements the Web Services Resource Framework (WSRF). WSRF defines a means of representing a stateful web service by a combination of a web service with one or more resources and their resource properties. Assignment 2 is a modification of the one in Bojo Sotomayor's tutorial [8] and is similar in structure to Assignment One. The web service with its associated resource, however, has state. Also, the Globus Toolkit is used to provide the servlet container.

2.3 Assignment 3: Job Submission

At this point the student has the basic understanding of how to define, create, deploy, and use a web service suitable for a grid. In Assignment 3 we focus on how to use a particular, very important, such service: the service that is used to submit a job. This service is called the Globus Resource Allocation Manager (GRAM). The student is first introduced to the Resource Specification Language-2 (RSL-2), which is also the XML-based language that GRAM uses.

The student then follows the sequence of steps needed for job submission using the executable form of a provided program as the job. A major part of this sequence involves using the Grid Security Infrastructure (GSI), which is a type of PKI (Public Key

Infrastructure) using mutual authentication via X.509 certificates, certificate authorities, proxy-based delegation, and the Secure Socket Layer (SSL) protocol. The sequence of steps involves not merely job submission, but also monitoring of the progress of job execution using GRAM commands. After successful job submission of the provided job, the students write their own job that meets specified functionality and repeats the previous steps.

2.4 Assignment 4: Job Scheduling

The Globus Toolkit only provides part of the software needed for a grid environment. Assignment 4 is the first assignment that goes beyond the Globus Toolkit. In particular, GRAM provides only basic job submission. The job manager that comes with GRAM for Linux execution hosts simply invokes the fork and exec system calls to run the job. Typically in a grid environment, each site is actually a computer cluster. There is an instance of GRAM running on the master node of each cluster that is connected via an adapter to a job scheduler for that particular cluster. A number of cluster schedulers are available and can be used with GRAM so long as an adapter between GRAM and that scheduler has been developed.

Assignment 4 introduces the student to the two cluster schedulers used in our grid and how to submit jobs to them through GRAM. Our two cluster schedulers are the free, open source Sun Grid Engine (SGE) and the commercial Load Sharing Facility (LSF). The adapter between GRAM and SGE was developed by Gridwise Technologies and one of the institutions that is part of our grid, MCNC. The adapter between GRAM and LSF comes with LSF from Platform Computing.

2.5 Assignment 5: Workflow Editor

By the end of Assignment 4, the student has an understanding of how the internals of a significant part of a grid environment work. Assignment 5 turns to the user's viewpoint. In a grid, the user is authenticated via a grid portal and then uses the portal to specify which resources to execute within a workflow. The workflow is then submitted and the user can monitor the progression of the workflow and use visualization to examine the results when the workflow completes. We illustrate part of the user's experience by examining the workflow editor developed at the Department of Computer Science at UNC-Wilmington [2].

3. THE END-TO-END APPLICATION

The end-to-end application was developed to provide an example shown and discussed early in the semester that provides a framework for all the assignments and in turn, for the course as a whole. Thus, we had several goals: include features of all the assignments involving the grid internals; illustrate a realistic application that includes all the components of a grid application; and be simple enough to be understandable to a student just starting to learn about grid computing. We feel that the application we developed achieves these goals and therefore is of interest to the wider community.

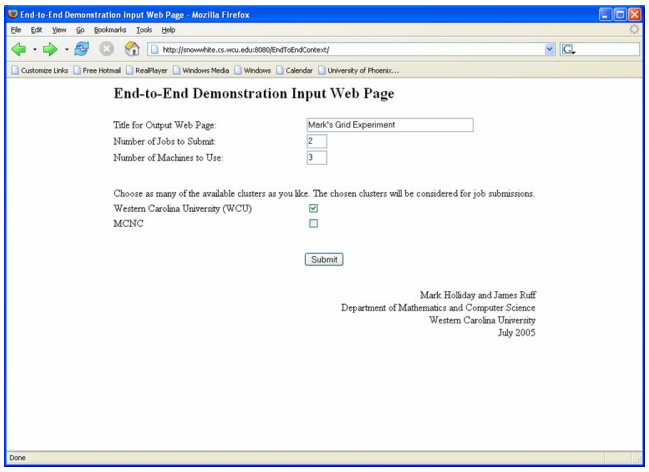

Figure 2. The input web page for the End-to-End demonstration.

3.1 Overview of the Demonstration

The first thing we show the students is a live demonstration by entering the URL

http://snowwhite.cs.wcu.edu:8080/EndToEndContext.

A web page with a form will appear as shown in Fig. 2. The inputs are:

- Title for output web page: This is an arbitrary character string that will appear on the output web page.
- Number of jobs to submit: For each cluster used, this is the total number of jobs that will be run on that cluster. This number of jobs will be distributed over the number of machines specified.
- Number of machines to use: For each cluster, this is the number of machines on which the jobs will be executed if there are enough jobs. For example, if the number of jobs to submit is 2 and the number of machines to use is 3, then in fact, two machines will be used since there are two jobs. If the number of jobs to submit is 3 and the number of machines to use is 3, three machines will be used with each machine running one job.
- The cluster check boxes: If a cluster's check box is checked (selected), the number of jobs specified above will be run using the number of machines specified above. If both clusters are selected, then both clusters run the complete number of jobs and use up to the specified number of machines.

The grid application executes and a dynamically generated web page is displayed. The output web page will include all of the student's input values as well as the output of the /bin/hostname command as run on several machines in the WCU cluster as shown in Figs. 3 and 4.

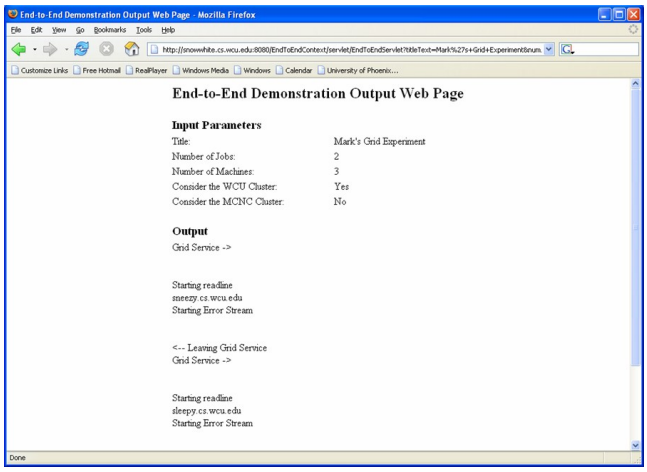

Figure 3. The top of the output web page of the end-to-end demonstration.

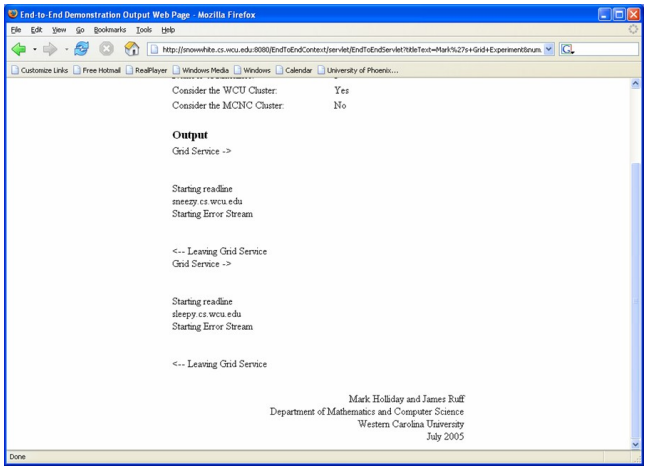

Figure 4. The bottom of the output web page of the end-to-end demonstration.

3.2 The Sequence of Events

Figure 5 diagrams the grid application parts being demonstrated as well as the sequence of events that occur.

The demonstration involves up to three machines in addition to the machines in the clusters. The sequence of events and the machines used are as follows:

1. The user machine has the web browser running on it. In a production-quality grid, the web page that would be loaded into the web browser would be a grid portal. A grid portal is a sophisticated web page that allows complex user interaction. For example, the Open Grid Computing Environment (OGCE), is a widely-used software tool for developing grid portals. Our demonstration uses a simplified analogue of a grid portal, which is a web page that contains a form. However, the basic technique (HTML, forms, web server, servlet) is the same. Initially in the demonstration, the web browser will download the web page. Information is entered in the form and the Submit button is pressed. This will cause the information in the text fields and check boxes to be sent by the web browser to the web server in an HTTP request. The HTTP request specifies that the Java servlet EndToEndServlet is to be used to service the HTTP request.

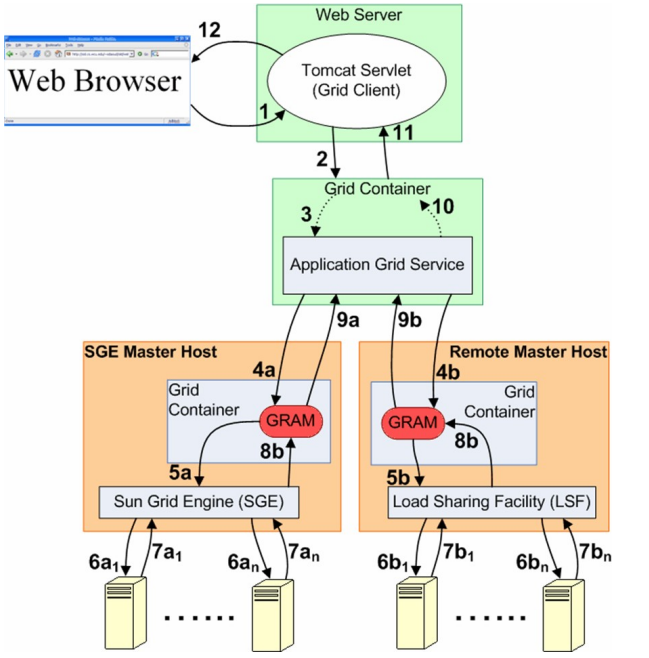

Figure 5. The parts of a grid application and the sequence of events initiated by the user.

2. The web server machine uses the Apache Jakarta Tomcat web server since it is both a HTTP server and a servlet container. A servlet provides a way for a web server to generate a web page dynamically. The web server receives the HTTP request including the information in the form's text fields and check boxes and invokes the Java servlet EndToEndServlet. The servlet extracts the form information, which appears to it as parameter values from the HTTP request. The servlet passes the information by a regular method call to the EndToEndClient object, which is on the same machine as the web server and the servlet.

3. The EndToEndClient object is an application-specific client for the EndToEndService web service. The grid client finds the location of the web service and remotely calls a method of the grid service. The parameters of the remote method call include the values that the grid client received from the servlet, which in turn came from the form in the web page.

4. The application, user-developed web service, EndToEndService, runs on a third machine. This web service runs within a grid container that is part of the Globus Toolkit. The code of the user-developed web service encodes the particular sequence of accesses to computational and data resources of the grid that are needed to perform the action requested by the grid client. The information provided by the grid client in the remote method invocation is taken into account in determining what grid resources are accessed.

5. The sequence of accesses to computational and data resources of the grid can be complex and data-dependent. In fact, it is a workflow. In general, the resources of a grid are controlled by one or more organizations with each organization maintaining clusters at one or more sites. A cluster has computational resources (the machines of the

cluster) and data resources organized as files or databases on storage devices. The user-developed web service spawns a series of jobs with each job being submitted to a machine on some cluster. That job uses that machine for computing and accesses the machine's storage devices for data.

6. For the example demonstration, we have chosen to use a very simple workflow. Our user-developed web service accesses resources of two organizations: Western Carolina University (WCU) and Microelectronics Center of North Carolina (MCNC). Both WCU and MCNC have clusters that are accessible by the grid service. The web service submits a series of jobs to each cluster. For the demonstration, a trivial job is used (the /bin/hostname command). However, that job could be replaced by a job that is arbitrarily complex with respect to both computing and data access.

7. Each job request is sent to one cluster. Each cluster has a master node and some number of other nodes. The cluster master node is running a web service called WS-GRAM (Web Services Grid Resource Allocation Manager). WS-GRAM is part of the Globus Toolkit; it is not application-specific. WS-GRAM provides a uniform interface to software (in our case, EndToEndService) or a person (at the command line) that submits a job to different clusters. Different clusters may be using different cluster batch schedulers. For example, the WCU cluster is using the Sun Grid Engine (SGE) cluster batch scheduler and the MCNC cluster is using the Load Sharing Facility (LSF) cluster batch scheduler.

8. WS-GRAM on the master node of a cluster uses adapter software to call the batch scheduler for that cluster. That batch scheduler is running on the node of the cluster which the batch scheduler views as the master node (which usually is, but does not have to be, the same node as the node WS-GRAM is running on). The adapter software is called by WS-GRAM, invoking what WS-GRAM calls the job manager for that batch scheduler. For example, the WCU cluster uses an adapter developed by Gridwise Technologies and MCNC to access SGE. The MCNC cluster uses an adapter to access LSF.

9. The cluster batch scheduler assigns the job to one or more machines. The job runs on the specified machines using their computing and data resources and generates output. That output is encoded as a character string and returned to the user-developed web service (via the cluster batch scheduler master software and then WS-GRAM).

10. The user-developed web service collects the output from all the jobs and generates a string that is their concatenation. That string is returned to the grid client.

11. The grid client passes that string to the servlet. The servlet then adds HTML to that string to generate a web page. That web page is returned to the HTTP server which then sends it to the web browser.

12. The web browser displays the output web page. From the viewpoint of the user, the user enters information in the web page text fields, clicks the Submit button, and then an output web page appears.

The end-to-end demonstration as described so far appears to allow unlimited access to the clusters. This is not the case; the version described has several means to control access (including only being deployed during a demonstration) In addition, the version that has been described is actually a subset of the complete end-to-end demonstration.

The complete version has a password-protected login form as the input web page. When the user enters a username and password and clicks the Submit button, a servlet is invoked on the web server that validates the username and password. If the username and password are invalid, a web page announcing that fact is dynamically generated and returned to the browser. If the username and password are valid, the servlet generates the web page that we have shown as the input web page; that is, the web page with the form for the number of jobs submitted, the number of machines used, and which clusters are to be used. The remainder of the complete version is the same as the version we have described.

4. THE RELATIONSHIP TO THE ASSIGNMENTS

Our explanation of the demonstration consists of four parts.

- The design of the front-end. The front-end means the HTML of the web page with the form, how pressing the Submit button causes the information in the text fields to be sent to the web server, how the web server calls the Java servlet, how the servlet retrieves the information about the text fields, how the servlets generate the output web page, and how the string representing the output web page goes back to the web browser via the web server. This material is discussed in the end-to-end course handout since none of the assignments addresses these aspects of a grid.

- A user-developed web service, EndToEndService, and a user-developed client for that web service, EndToEndClient. We discuss and review the source files for both the client and the service. We also go over the Web Service Definition Language (WSDL) file for the service. This maps to the first two course assignments.

- An important pre-existing web service provided as part of the Globus Toolkit, WS-GRAM. We discuss the role of WS-GRAM and the code in our user-developed web service, EndToEndService, calls WS-GRAM. This maps to the third course assignment.

- WS-GRAM provides a uniform interface that user-developed web services can use to submit jobs to multiple clusters even though each cluster may be using a different cluster job scheduler. For example, the WCU cluster uses the SGE cluster scheduler and the MCNC cluster uses the LSF cluster scheduler. Each different type of cluster scheduler must have an adapter between it and the instance of WS-GRAM running on that cluster. This part of the end-to-end demonstration maps to the fourth course assignment.

5. CONCLUSIONS

An important feature of the course is that the course experiences mirror true grid computing. The course structure is an example of wide-area distributed computing with students and faculty from many universities attending at distributed locations. Furthermore, the students are using an actual grid involving clusters at multiple sites both for the end-to-end-demonstration

and for their assignments. The large number of universities and colleges with students and faculty participating in our two course offerings, as well as student course evaluations, indicate that our course design is fulfilling a need. Moreover, the initial feedback that we have received about the end-to-end demonstration is that the students are finding it helpful as a concrete example integrating the concepts they are learning through the programming assignments and lectures.

We have described the course structure of an undergraduate grid computing course. The course is interesting at several levels. It is taught in a grid environment with students at many sites, and common grid software is deployed over multiple distributed clusters. The actual lecture content and sequence of programming assignments are carefully integrated. The end-to-end demonstration, the main focus of this paper, provides a simple, but complete and realistic example of an application that starts with the user interface front-end, includes both user-developed and pre-existing web services, and reaches the back-end of the computational and data resources of the clusters. Such a demonstration provides a framework integrating the concepts of the course lectures and the practice of the course programming assignments. More information about the end-to-end demonstration and other course materials is available [3, 9].

6. ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation under grants DUE 0410667 and 0533334 (with additional funding) and two grants from the University of North Carolina Office of the President. One of the principal tenets of grid computing is the concept of a virtual organization, a team of people geographically distributed and working together toward a common goal computing sharing resources. To be able to teach this course, we have been supported by a large team of people, distributed at many sites, who have installed complex software and supported the students. We have been constantly amazed at how this virtual organization has worked together to provide the infrastructure for the course. We are grateful to all the members of our virtual organization, especially Steve Thorpe and Chuck Kessler (MCNC), and Clayton Ferner (UNC-Wilmington), the co-instructor of the course.

7. REFERENCES

[1] Apon, A., Mache, J. Yara, Y., and Landrus, L., "Classroom Exercises for Grid Services," *Proc. Linux Charter Institute Int. Conf. on High Performance Computing*, Austin, TX, USA, May 2004.

[2] GridNexus home page, www.gridnexus.org.

[3] Holliday, M. and Ruff, J., "The Western Carolina University Grid Project", http://sol.cs.wcu.edu/~certauthority/.

[4] Holliday, M., Wilkinson, B., House, J., Daoud, S., and Ferner, C., "A Geographically-Distributed, Assignment-Structured Undergraduate Grid Computing Course," In *Proc. SIGCSE 2005 Technical Symposium on Computer Science Education*, St. Louis, Missouri, February 23 - 27, 2005.

[5] Li, M. and Baker, M., *The Grid: Core Technologies*, John Wiley, 2005.

[6] National Science Foundation Middleware Initiative, http://www.nsf.org/toolkit/.

[7] Ramamurthy, B., "GridForce: A Comprehensive Model for Improving the Technical Preparedness of our Workforce for the Grid", *Int. Workshop on Grid Education (CCGrid04)*, April 2004, Chicago, IL. USA.

[8] Sotomayor, B., *The Globus Toolkit 4: Programming Java Services*, Morgan Kaufman, 2005.

[9] Wilkinson, B. and Ferner, C., "ITCS 4010 Grid Computing course home page," University of North Carolina at Charlotte and University of North Carolina at Wilmington, http:// www.cs.uncc.edu/~abw.

[10] Wilkinson, B., Holliday, M., and Ferner, C., "Experiences in Teaching a Geographically Distributed Undergraduate Grid Computing Course," *Workshop on Collaborative and Learning Applications of Grid Technology and Grid Education, IEEE International Symposium on Cluster Computing and the Grid (CCGrid2005*), Cardiff, UK, May 9 - 12, 2005.