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.

Experiences in Teaching a Geographically Distributed Undergraduate Grid Computing Course

Barry Wilkinson[1], Mark Holliday[2], and Clayton Ferner[3]
[1] Department of Computer Science, University of North Carolina at Charlotte
[2] Department of Mathematics and Computer Science, Western Carolina University
[3] Department of Computer Science, University of North Carolina at Wilmington
abw@uncc.edu, holliday@wcu.edu, cferner@uncw.edu

1. Abstract

This paper describes the experiences of teaching a new undergraduate grid computing course to students across the State of North Carolina in Fall 2004. This course is one of the first at the undergraduate level and one of the first to have a large number of geographically distributed participating sites. New materials have been developed specifically targeted towards undergraduate students. We have developed a coherent set of grid computing programming assignments. All the materials developed are available on-line. This course also had internationally known guest speakers to provide state-of-the art presentations. Apart from developing grid computing materials, a number of issues arose in using the grid computing software and coordinating the activities, which are described here.

2. Introduction

Grid computing uses geographically distributed computers connected on the Internet collectively for high performance computing and resource sharing. It often involves computers from multiple organizations and crosses organizational boundaries and enables the creation of distributed teams (so-called virtual organizations). The goal of the work described in this paper is to develop new educational materials on grid computing suitable for undergraduates and to introduce a hands-on grid computing into the undergraduate curriculum. Since grid computing involves computers at multiple sites on the Internet, it was necessary to have cooperating sites. We took advantage of the existing state-wide North Carolina Research and Education Network (NC-REN) to present the lectures to geographically distributed sites. Computers were set up with grid software at major sites for students. This project required the cooperation of many people at eight universities. Funding was provided by the National Science Foundation and the University of North Carolina Office of the President. For the first offering of the course, existing computers were used at sites. However, at least five clusters of state-of the-art computers are being purchased for subsequent offerings of the course.

NC-REN which was used as the instructional medium is a telecommunications network that became operational in 1985 to interconnect universities, medical center, research institutions, and graduate centers in North Carolina. It provides multi-way, face-to-face video and audio communications, originally employing a combination of microwave links and land lines but now almost totally long distance digital fiber optic lines. Classroom “teleclass” facilities exist at each site, such as shown in Figure 1. Each student is provided with a microphone. Several video cameras are used so that the instructor and students at each site can hear and see each other, and participate in

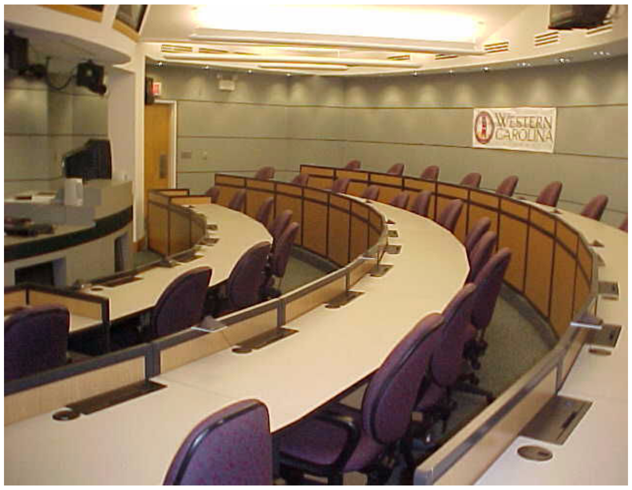

Figure 1. One teleclass classroom facility at Western Carolina University.

classroom discussions. All lectures are recorded so that students can view lectures again or catch-up on any they missed. The Fall 2004 grid computing course described in this paper originated at Western Carolina University (WCU) and broadcast on the NCREN network simultaneously to the University of North Carolina at Wilmington (UNC-Wilmington), University of North Carolina at Asheville (UNC-Asheville), Appalachian State University, North Carolina State University, the University of North Carolina at Greensboro (UNC-Greensboro), Elon University (a private institution), and North Carolina Central University.

Figure 2 shows a map of the universities in the University of North Carolina System with the participating sites identified. Elon University is not shown, where faculty were in attendance. Also four students initially enrolled at UNC-Charlotte although this site dropped the course for financial reasons. (Some sites have to pay for receiving classes.) Final student demand was still high. Over 40 students enrolled in the course including two from Cape Fear Community College attending at UNC-Wilmington. Most students were senior undergraduate Computer Science students. Students received credit at their home institution for the course. Several faculty were in attendance and some lectures were given by faculty from UNC-Wilmington.

Near the end of the course, internationally known guest speakers (Professor Daniel Reed and Dr. Wolfgang Gentzsch) gave presentations to the class from different sites. One taped presentation by Ian Foster was given. One NC State University graduate student gave a presentation on his work on grid computing and Chuck Kelser from the Microelectronics Center of North Carolina also gave a presentation on legal issues.

Computers were set-up with grid software at three major sites (WCU, UNC-Wilmington, and NC State University) where there was a significant number of students registered (around 13 at each of these sites). At other sites where there were only one or two students, students were

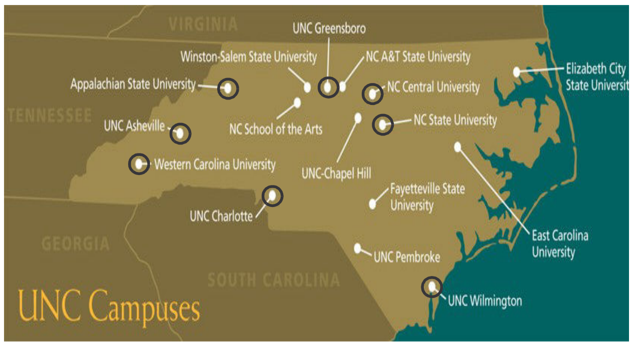

Figure 2. The University of North Carolina campuses with participating sites marked.

given accounts on WCU computers. We installed components of the National Science Foundation NMI grid software package [12] at each site, notably Globus, Condor-G, and MPICH-G2. In addition, a workflow editor called Grid-Nexus was installed and used in the latter part of the course.

An extensive home page was developed that provides links to detailed instructions for installing and using the software, and teaching materials (slides, assignments, etc.) [13]. WebCT was used for submission of programming assignments, and for quizzes. We will discuss how we used the software, WebCT and the home page in the class in subsequent sections.

The paper is organized as follows: In Section 3, we describe the course structure including the topics covered and assignments. In Section 4, we describe experiences of teaching the course, starting with how we evaluated the course and then the problems we encountered. In Section 5, we briefly outline the future work, which includes creating an on-line version of the course. Conclusions are given in Section 6.

3. Course Structure

The course is designed primarily for upper-level undergraduate Computer Science students, although graduate students were accommodated by providing extra work. The software used required knowledge of Linux, Java, and C. Most upper-level undergraduate Computer Science students have this knowledge or can quickly learn it.

There is no course textbook. No suitable course textbook exists. All materials are prepared specially for the course in the form of Powerpoint slides and Word documents. Links are provided to classic materials [8], [9], which are set as reading assignments. Reference grid computing books for the course include [2], [3], [4], and [5]. On-line books include [6] and [7].

The topics covered in the course are listed below:

- Review of Internet technologies
- Introduction to grid computing
- Web services
- Grid services
- Security, Public Key Infrastructure
- Open Grid Services Architecture (OGSA)
- Globus (version 3.2 in Fall 2004)
- Condor-G
- MPI and grid-enabled MPI (MPICH-G2)
- GridNexus workflow editor
- Grid computing applications

More details of these topics are given in Tables 1 - 6, separated by those parts done before each assignment.

Table 1. Weeks 1 - 3

Topic	Details
Grid Computing	Virtual organizations, computational grid projects, grid computing networks, grid projects in the US and around the world, grid challenges.
Internet Technologies	IP addresses, HTTP, URL, HTTP, XML, Telnet, FTP, SSL.
Web Services	Service-Oriented Architecture, service registry, XML documents, XML schema, namespaces, SOAP, XML/SOAP examples, WSDL, portType, message definition, WSDL to code, WSDL from code.
Assignment 1	"Simple" Web service Java programming assignment. Tomcat environment, Axis, JWS facility.

Table 2. Weeks 3 - 4

Topic	Details
Grid Service	Concepts, differences to Web services, stateful/stateless/transient/non-transient, OGSA, OGSI, grid service factory, WSRF.
Assignment 2	"Simple" grid service Java programming assignment. Globus 3.2 environment.Tools: Ant.

Table 3. Weeks 4 - 6

Topic	Details
Security	Secure connection, authorization requirements, symmetric and asymmetric key cryptography, non-repudiation, digital signatures, certificates, certificate authorities, X.509 certificate.
Globus: Part 1	Basic structure (version 3.2), grid service container, service browser, Globus Resource Allocation Manager (GRAM), job submission with managed-job-globusrun, Grid Security Infrastructure (GSI), Globus certificates, simpleCA, proxies, creating a proxy.
Globus: Part II	Resource management, Master Managed Job Factory Service (MMJFS), more on managed-job-globusrun. Resource Specification Language (RSL and RSL-2), syntax, examples.
Assignment 3	Submitting a Job to the Grid, GT3 managed-job-globusrun, job specified in RSL-2.

Table 4. Weeks 6 - 7

Topic	Details
Globus: Part III	Information Directory Services, LDAP, resource discovery.
Schedulers and resource brokers	Condor, submit description file, DAGMan, checkpointing, Class-Ad, Condor-G, other systems.
Assignment 4	Submitting a Condor-G Job.

Table 5. Weeks 7 - 9

Topic	Details
High performance computing (HPF)	Grand challenge problems, parallel computing, potential speed-up, types of parallel computers, shared memory, message-passing, programming.
Parallel Programming	Techniques suitable for a Grid, embarrassingly parallel computations, Monte Carlo, parameter studies, sample "big" problems.
Cluster Computing	Basic message passing techniques, history, system software, programming models, synchronous and asynchronous message passing, message tags, collective routines.
MPI	Process creation, communicators, unsafe message passing, point-to-point message-passing, blocking, non-blocking, communication modes, collective communication, running an MPI program on a cluster.
Grid-enabled MPI	MPI-G2 internals, mpirun command, RSL script.
Assignment 5	Running a simple MPI-G2 program.

Table 6. Weeks 10 - 12

Topic	Details
Workflow	Concept, examples.
GridNexus I	GridNexus UNC-W GUI for Workflow management.
GridNexus II	Implementation of GridNexus, JXPL. Presented by Professor Jeff Brown.
Grid portals	Grid portals, purpose, application-based portals, historical examples, GPDK, Gridport, Hotpage, NCSA Alliance portal, DOE Fusion Grid portal, etc., portal implementation, portlets, JSR 168, OGCE portal.
Assignment 6	GridNexus assignment.

The course is designed as a true hands-on bottom-up course with carefully crafted and documented programming assignments that build a body of knowledge and skills. The assignments are interspersed in the course at appropriate places when the related topics are being covered in the lectures, as indicated in Tables 1- 6. Brief details of the assignments are given in Tables 7 - 11.

Assignments start with an assignment 1: The design and implementation of a web service. A web service is introduced as the first assignment because grid computing today is based upon grid services which are an extension of web services. Assignment 1 is based upon [1]. Assignment 2 calls upon the student to implement a simple grid service and a client to use the service. This assignment is based upon [11].

Table 7. Assignment 1

Objective: Learn how to write, deploy, and use a web service, using Apache Tomcat Java Servlet container and tools.	
1	Create a simple math web service (Java Web Service file).
2	Generate several required files from jws file and compile.
3	Create client source file and compile.
4	Execute client to access service.
5	Extend service functionality and test.

Table 8. Assignment 2

Objective: Learn how to write, deploy, and use a grid service, using Globus 3.2 Toolkit.	
1	Run name change script.
2	Create a math grid service with several methods.
3	Define grid service interface using GWSDL.
4	Write deployment descriptor.
5	Build and deploy grid service.
6	Create client source file and compile.
7	Start Globus container on a free port.
8	Execute client to access service.
9	Extend service functionality and test.

Table 9. Assignment 3

Objective: Learn how to submit jobs in Globus 3.2.	
1	Start Globus container on a free port.
2	Start a proxy process (obtain proxy certificate).
3	Submit a pre-existing job using the Globus managed-job-globusrun command. This command uses an RSL-2 job description file.
4	Create your own job, compile and submit.
5	Write and deploy your own grid service accessed by multiple identical clients started using the managed-job-globusrun command.

Table 10. Assignment 4

Objective: Learn the Condor-G scheduler and Globus Resource Allocation Manager.	
1	Start a proxy process (obtain proxy certificate).
2	Check status of Condor pool.
3	Create a Condor job description file for a simple job and submit a job.
4	Check status of job and manage job using Condor commands.
5	Write and test a computationally intensive Java job according to specification.
6	Submit job to a Condor Java universe and test.
7	Submit job to a Condor Globus universe and test.

Table 11. Assignment 5

Objective: Learn how to write and submit MPI programs to a grid using MPICH-G2.	
1	Start a proxy process (obtain proxy certificate).
2	Create and compile a simple MPI program.
3	Execute MPI program on two machines.
4	Add communication to MPI program and re-execute program.
5	Write, compile and execute an MPI ping program.
6	Write, compile and execute an MPI program for numerical integration using broadcast and reduce operations.

Table 12. Assignment 6

Objective: Learn how to create a GridNexus workflow that uses a web service and a grid service.	
1	Download GridNexus and install on your computer.
2	Check that your service from assignment 1 and 2 are deployed. If not, deploy them, or use the deployed services provided.
2	Create a web service workflow.
3	Create a grid service workflow.
4	Create a workflow that uses both web and grid services.
5	Extend service functionality and test.

Assignment 3 uses the Globus job submission service, GRAM, to submit jobs that use a grid service. This assignment asks the student to submit multiple simultaneous jobs representing shoppers. The grid service holds the inventory of an item being bought by the shoppers. The assignments so far require quite detailed command line actions and editing of complex files such as deployment description files.

Assignment 4 continues with job submission but using the Condor-G scheduler to submit jobs. (Globus does not have a fully fledged job scheduler.) This assignment introduces students to the job description language used with Condor-G, and job dependency graphs. The next assignment, assignment 5, asks students to execute MPI programs on the grid, using MPICH-G2. Although tested and ready to go, this assignment was skipped in the Fall 2004 course because of lack of time.

Moving to a higher level tool, assignment 6 uses a workflow editor called GridNexus developed by UNC-Wilmington [10]. By the end of assignment 5, the student has an understanding of how the internals of a significant part of a grid environment works. It is now time for the student to view the grid from the perspective of an end user. There are two areas of development that focus on user interaction with the grid: portals and workflow editors. Both types of interfaces are GUI-based. However, portals are web-based interfaces, whereas workflow editors are applications that allow the user to describe interactions between computation components using directed graphs (workflow). Figure 3 shows a chemistry example of a workflow generated by GridNexus that involves several processing steps of a molecule.

In assignment 6, the students are asked to use GridNexus to construct a workflow that uses their web service from assignment one and their grid service from assignment two. First, the students must start the Globus container and verify that their services are working. Then they are required to configure a generic web service module in GridNexus that communicates with their web service. This step requires nothing more than providing the URL of the web service WSDL. The students perform a similar step to configure a module to communicate with their grid service. However, configuring a module for a grid service requires a few more steps. In particular, the students must download the JAR files associated with their grid service and provide the URL of the factory service, the stub class, and the interface class.

Once these modules have been configured and the students have tested them to verify that they work properly, the students must put them together into a more complex

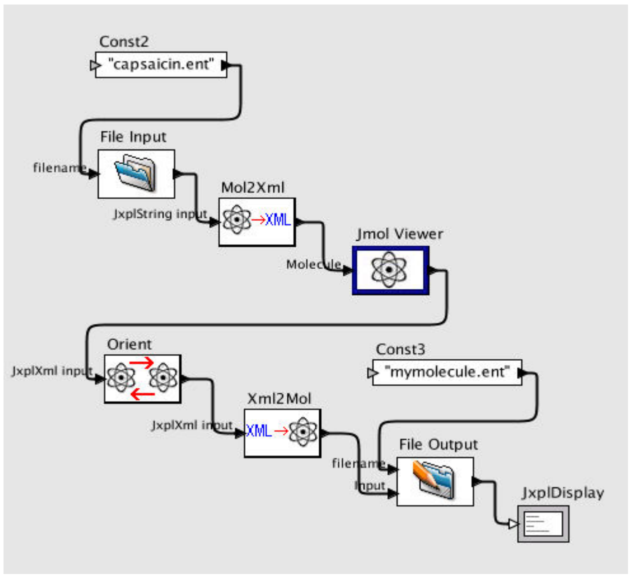

Figure 3. A workflow example using Grid-Nexus.

workflow. The students have to construct a workflow that 1) gets the value of their grid service counter, 2) squares it using their web service, 3) adds the result back to their grid service counter, 4) gets the new value of their grid service counter, and 5) displays the result. Although the functionality of the above workflow may seem trivial, the point of this exercise is for the student to use web and grid services from a higher-level point of view. Contrast the steps that would be required to perform this workflow using only the steps and tools of assignment one and two.

Very detailed instructions are provided for each assignment. They begin with step-by-step instructions to achieve tasks, and end with tasks that require the knowledge learned in the earlier tasks. More details concerning these assignments can be found in [13].

Guest presentations occurred after week 12, while student completed their last assignment. Presentations were:

- Professor Ian Foster: Taped presentation "The Grid: Beyond the Hype," by Ian Foster, Argonne National Laboratory and University of Chicago, (originally given at Duke University, Sept. 14th, 2004).
- Wolfgang Gentzsch, Managing Director, MCNC Grid Computing and Networking Services. Title of presentation "Grid Computing in the Industry"
- Professor Daniel A. Reed, Chancellor's Eminent Professor, Director of Institute for Renaissance Computing, University of North Carolina at Chapel Hill, Duke University, and NC State University. Title of presentation: "Grid computing: 21st Century Challenges."
- Chuck Kesler, Director, Grid Deployment and Data Center Services, MCNC. Title of presentation: "Security Policy, Legal, and Regulatory Challenges in Grid Computing Environments."
- Sammie Carter, NCSU graduate student. Title of presentation: "Wolfgrid: the NCSU community supercomputer and Datamation Sort using OpenMP and MPI."

4. Experiences

Feedback for this course was done in several ways. First, a brief paper evaluation form was handed out on several occasions throughout the course. This evaluation form asked three basic questions: First, was the presentation in the distance education classroom easy to follow. Second, was the pace of instruction in the distance education classroom appropriate. Finally, the student were asked whether both local and remote participants were able to take part in the lesson.

A much larger on-line, end-of course evaluation form was produced with a large number of questions designed to obtain feedback on the delivery of the course, the course content and the student perception of the course. This online course evaluation form was prepared by Barbara Heath (evaluator for the project). The course evaluation form can be found at http://firewin.bear.uncw.edu/survey/TakeSurvey.asp?SurveyID=3L37p3612p4KG.

The mode of instruction was through prepared Powerpoint slides. This mode of instruction has some intrinsic disadvantages. For example, it generates a rather one-sided and staid approach to teaching. It can bore students if they do not have sufficient opportunity to interact with the instructor. It also tends to lead to excessive material. The slides commonly consist of bulleted items that are discussed but the discussion is not documented. Students simply watch the slides pass by. On the positive side, slides provide the ability to present animation, graphics and images.

Another concern of the instructors was meeting students outside class to help solve programming issues. Having instructors at two sites (UNC-W and WCU) allowed students at these sites to receive help from an instructor. We also employed two talented teaching assistants at WCU (Jeff House and James Ruff) who were responsible for handling day-to-day problems that frequently occurred throughout the course. Email was used continually to communicate with individual students at all sites.

One site visit was conducted at NC State University near the end of the course to talk with students there. This site visit coincided with one of the guest presentations (Professor Reed). The site visit to NC State University confirmed the view that Powerpoint slides alone do not provide sufficient opportunity for two-way dialogue and can be less interesting over the full course than interspersed with other teaching strategies. However, simply writing on a white board would not work as one might do in a non-broadcast course as lighting makes it difficulty for the cameras to pick up writing. Similarly, writing by hand on a white board on a desk viewed by an overhead camera, although better for image pick-up also is difficult for students to see and discern the handwriting. The normal classroom techniques do not work well. It appear that the best approach is a mixture of techniques, and not all Powerpoint slides. It is important to maintain interest by engaging remote students especially. Students at NC State University suggested more demonstrations of grid computing, possibly tied to the assignments.

As described earlier, assignments were set at various places in the course to match the topic being presented. Each assignment required students to do predefined tasks such as deploying pre-written services and executing clients but also to test their own web services and grid services and client using the services. In a typical programming course, one would ask for a write-up of the work, the sources files, and maybe a demonstration of the programs. Source files would be tested to prove that the programs actually worked. If they did not work, the instructor would be able to provide valuable feedback to the student. Demonstrations by the students are useful to verify that the students did the work and understood what they did. With this course however, it is not possible to ask for demonstrations as students are at various remote sites. Therefore, each student was asked to submit a write-up that included screen shots at various stages of the assignment, submitting the document as a Word document.

At first, these documents were submitted by students through a web-based submission system called Hermes (see http://www.cs.wcu.edu/~hermes) that was developed in-house by the Department of Mathematics and Computer Science at WCU several years ago. Hermes is used for various programming courses within the department including the Java based Freshman programming courses and an HTML/Javascript course for non-majors. Hermes allows students to upload files of their choice. The files are deposited in folders identified by the course name, assignment and student name within the instructor's home directory.

However, Hermes is somewhat inconvenient for several reasons. Most notable is that there is no convenient mechanism to identify students from different institutions as occurs in this course. Hermes time-stamps submissions by attaching an increasing number to the submission file name. From that, one can tell the most recent submission. It also appends a time-stamped header to the submission and creates other files. The low level nature of the submission system makes it quite difficult to use for this course, and after assignment 2, WebCT was used.

WebCT was found quite adequate for assignment submission although again there was no direct way of identifying students for different institutions. Indeed, WebCT organizes students alphabetically. To group students by institution, one could add a prefix to student names identifying the institution. Knowing the student's institution became important when assessing the assignments, as different sites had different problems with the computers and software.

In fact, only WCU had a complete working set of software for all the assignments, and this was achieved partly because the software had been installed over the summer, and the very significant technical problems could be resolved. The other sites had to load the various software components during the semester. It is a challenging exercise to install Globus 3.2 and make it work continuously with many students.

Globus itself poses very significant problems for using in a grid computing course. Globus is not designed as a mult-user/developer tool and we had to provide a work-around to get so many students to use it simultaneously. Students create grid services and deploy them in a Globus container to make the grid service available. Since all students will deploy their services into the same container, it is necessary to have unique names for every service that is deployed. Sam Daoud, a WCU student, created a script that takes the student username and replaces "assignments" with "yourusernameAssignments," where yourusername is the username the student used to log on. Students need to run the script only once to make the necessary changes. The details of the script can be found in the assignment 2 write-up, see [13].

Another way to use Globus is to have each student install their own copy of Globus on their computer. UNC-Wilmington actually provided each student in the course with a loaner laptop for loading GridNexus and if they wished, Globus. The students had full use of their laptop for the duration of the grid computing course, returning the laptop at the end of the course.

The most critical factor for success of a hands-on undergraduate grid computing course is the software. All the software must be operational well in advance. Each student account requires a "certificate." We set up 35 student accounts on the WCU system, called student1 through student35 and tested the accounts before allocation to students. Even so, it is possible for students to cause significant disruption to the system inadvertently. For example, system problems can occur if a user asks for a new certificate while still having a valid certificate. On occasion, we have resorted to re-booting system automatically in the early morning hours to clear out any unfortunate situations.

5. Future Work

Since the course was recorded on tape, it is feasible to convert the recording to streaming-video/audio and faculty at both Elon University and NC State University have already experimented with doing this with the course video tapes. We plan to convert the whole course onto a form that can be viewed on-line.

The course will be offered again in Fall 2005, and before then, we will be revising the assignments, and especially developing ways to make the software more stable under student conditions. The revised course will include University of North Carolina at Charlotte (UNC-C) as one of the originating sites. University of North Carolina at Charlotte was not one of the participating sites for the first offering. We will develop more grid materials and assignments that use multiple grid nodes.

A follow-on course is also being planned in which students apply grid computing techniques to application areas and subjects learned in other senior level courses, such as image processing, graphics, artificial intelligence, and parallel algorithms. This follow-on course, entitled "research seminar," will be available to students at all participating sites.

We have already received interest for other universities about the course and we make our materials freely available for all to use. We hope to give grid computing training workshops for faculty in the future.

6. Conclusions

As with many advancing topics in Computer Science, first graduate students were exposed to grid computing in topics and seminar type courses. There are several such examples of graduate level grid computing courses in the USA and around the world. The grid computing course described in this paper is one of the first truly at the undergraduate level. It is also perhaps the first to have a large number of geographically distributed participating sites. New materials have been developed targeted towards undergraduate students. We have developed a coherent set of programming assignments. The guest speakers provided state-of-the art presentations with insight not usually found in undergraduate courses.

We see grid computing as becoming a common course in the undergraduate Computer Science curriculum. It was alluded to in the IEEE/ACM Computing Curriculum 2001 under the topic "net-centric computing."

7. Acknowledgements

This work is supported in part by the National Science Foundation under grant DUE 0410667 and grants from the University of North Carolina Office of the President. The authors are also grateful to Jeff House and James Ruff, Computer Science majors at WCU, who worked as research assistants on the project, installing software, helping on the course with student problems, and development of assignments.

8. References

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

[2] Abbas, A., *Grid Computing: A Practical Guide to Technology and Applications*, Charles River Media, Inc., Hingham, MA, 2004.

[3] Berman, F., G. C. Fox, and A. J. G. Hey editors, *Grid Computing Making the Global Infrastructure a Reality*, Wiley, Chichester, England, 2003.

[4] Foster, I., and C. Kesselman editors, *The Grid: Blueprint for a New Computing Intrastructure 2nd edition*, Morgan Kaufmann, San Francisco, CA, 2004.

[5] Joseph J., and C. Fellenstein, *Grid Computing*, Prentice Hall/IBM Press, Upper Saddle River, NJ, 2004.

[6] Ferreira L., et al, *Introduction to Grid Computing with Globus*, IBM Redbooks, 2003, http://www.redbooks.ibm.com/redbooks/pdfs/sg246895.pdf.

[7] Ferreira, L., et al, *Globus Toolkit 3.0 Quick Start*, IBM Redbooks, 2003, http://www.redbooks.ibm.com/redpapers/pdfs/redp3697.pdf.

[8] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *Int. J. Supercomputer Applications*, 2001. http://www.globus.org/research/papers/anatomy.pdf.

[9] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," http://www.globus.org/research/papers/ogsa.pdf.

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

[11] B. Sotomayor, "The Globus Toolkit 3 Programmer's Tutorial," http://www.casa-sotomayor.net/gt3-tutorial/multiplehtml/index.html.

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

[13] B. Wilkinson and C. Ferner, "Grid Computing course home page," http://www.cs.uncc.edu/~abw/ITCS4010F05.