

Early Experiences with the EGrid Testbed

Gabrielle Allen, Thomas Dramlitsch, Tom Goodale,

Gerd Lanfermann, Thomas Radke, Ed Seidel

Max-Planck-Institut für Gravitationsphysik, Potsdam, Germany

{allen,thomasd,goodale}@aei-potsdam.mpg.de

{lanfer,tradke,eseidel}@aei-potsdam.mpg.de

Thilo Kielmann, Kees Verstoep

Vrije Universiteit

Amsterdam, The Netherlands

{kielmann,versto}@cs.vu.nl

Zoltan Balaton, Peter Kacsuk, Ferenc Szalai

MTA SZTAKI, Budapest, Hungary

{balaton,kacsuk,szferi}@sztaki.hu

Joern Gehring, Axel Keller, Achim Streit

Paderborn Center for Parallel Computing, Germany

{joern,kel,streit}@upb.de

Luděk Matyska, Miroslav Ruda

Aleš Křenek

Masaryk University

Brno, Czech Republic

{ludek,ruda,ljocha}@ics.muni.cz

Hans-Hermann Frese, Harald Knipp, André Merzky

Alexander Reinefeld, Florian Schintke

Konrad-Zuse-Zentrum für Informationstechnik

Berlin, Germany

{frese,knipp,merzky,reinefeld,schintke}@zib.de

Bogdan Ludwiczak, Jarek Nabrzyski, Juliusz Pukacki

Poznań Supercomputing and Networking Center, Poland

{bogdanl,naber,pukacki}@man.poznan.pl

Hans-Peter Kersken

DLR, Köln, Germany

Hans-Peter.Kersken@dlr.de

Giovanni Aloisio, Massimo Cafaro

University of Lecce, Lecce, Italy

{giovanni.aloisio,massimo.cafaro}@unile.it

Wolfgang Ziegler

Forschungszentrum Informationstechnik GmbH

Sankt Augustin, Germany

ziegler@gmd.de

Michael Russell

University of Chicago, Chicago, USA

russell@cs.uchicago.edu

Abstract

The Testbed and Applications working group of the European Grid Forum (EGrid) is actively building and experimenting with a grid infrastructure connecting several research-based supercomputing sites located in Europe. This paper reports on our first feasibility study: running a self-migrating version of the Cactus simulation code across the European grid testbed, including “live” remote data visualization and steering from different demonstration booths at Supercomputing 2000, in Dallas, TX. We report on the problems that had to be resolved for this endeavour and identify open research challenges for building production-grade grid environments.

1 Introduction

Grid computing, the deployment and utilization of distributed, network-connected computing resources, is increasingly drawing the attention of computational scientists around the world [12]. In the last few years, a lot of software infrastructure for Grid computing has been developed, particularly through projects such as Globus [11], Legion [17], and SNIPE [10]. Their systems take care of security mechanisms needed for authentication to various resources, program startup, data transfer mechanisms between machines, and services to monitor and report characteristics and status of the various resources on the Grid.

To date, most Grid software development has taken place in the US, with its rather advanced HPC infrastructure de-

veloped in the last decades. During the last 18 months, the *US Grid Forum* has been exploring various Grid computing settings and laying down possible standards and best practice scenarios. In parallel, a European Grid Forum, called the *EGrid*, began to organize itself in spring 2000, with its first meeting in Poznań, Poland. At that meeting various working groups were initiated, representing over a dozen European HPC centers from Germany, France, Italy, the Netherlands, the Czech Republic, England, Ireland, Scotland, Hungary, Poland, Greece, Denmark, and Sweden. In spite of differences in national computer networks, and a large diversity of computing platforms (e.g., machines from Japanese vendors are far more common than in the US), there is considerable enthusiasm among the EGrid members about the promise of Grid computing in Europe and worldwide, and about the ability of the various European groups beginning to make substantial contributions to this emerging global technology.

The Grid computing infrastructure has been maturing during the last years. In particular, the Globus Toolkit is currently emerging as a common infrastructure across many US academic HPC centers. But still, Grid computing has yet to be widely deployed and used in production with a broad spectrum of applications. To address this problem, the EGrid Testbed and Applications Working Group was formed with the task of gaining “hands-on” experience with Grid computing. As a starting point, the Globus Toolkit was chosen to provide the Grid infrastructure across the EGrid, in order to evaluate it for use with demanding and innovative Grid applications. The primary application area is provided by Cactus [2], a powerful and generic simulation toolkit, developed primarily at the MPI-Gravitationsphysik in Potsdam, Germany, which has been used in many large scale applications (from black holes to parallel rendering) and Grid computing experiments. The goals of the EGrid working group are to deploy and test current Grid computing infrastructure, to tightly couple this infrastructure with real applications that are designed to exploit the many new opportunities enabled by Grid computing, to provide a testbed laboratory for Grid computing in Europe, and ultimately to bring user communities into the Grid world.

In this paper, we report on our early experiences of deploying a Grid infrastructure across the many sites in our international neighborhood, and on two key Grid computing experiments developed with the Cactus Computational Toolkit, uniquely taking advantage of a Grid computing environment.

2 EGrid: Past, Present, Future

The birth of a super-national initiative is often not a straightforward process. In the case of EGrid, it took three Europeans and a major North-American(!) conference

(HPDC-8 in August 1999) to bring out the idea of a pan-European initiative for grid computing. After some public announcements, the vision was taken up by the community in a surprisingly short time. The enthusiasm of the growing number of volunteers made the subsequent organizational meetings in Portland (November 1999) and Berlin (January 2000) a great success. At the first two official EGrid forums in Poznań (April 2000) and Munich (August 2000) we already counted more than 50 participants.

The success was anything but guaranteed, because of the lack of a coherent funding regime for long-term research collaborations in Europe. Most of the academic R&D is done by PhD students, who are typically employed on temporary contracts for just a few years. Even worse, several interested groups found it difficult to get the necessary money for traveling to the EGrid events.

While there is some hope for a better situation in the future, at the present time, all work in EGrid is voluntarily performed by the pure enthusiasm of the participants. EGrid is a community-driven initiative focusing on the promotion and development of grid technologies and applications by means of implementing “best practices” with an emphasis on rough consensus. EGrid comprises five working groups which focus on data management, resource management, programming models, testbeds, and performance analysis. The aims and goals of each working group are stated in their respective charters [9].

For grid computing, there is no boarder between nations. So it was only natural to merge the EGrid Forum with the successful U.S. Grid Forum and the Asia Pacific Grid initiative to create a *Global Grid Forum (GGF)*. This worldwide initiative has a total of ten working groups with more than 500 participants from 150 organizations in more than twenty countries. GGF working groups develop agreements and distill common practices in a growing community of people worldwide working on projects building grids.

3 Testbed Infrastructure

The sites participating in the EGrid testbed agreed to install a common set of grid and application software as an initial starting point for grid experiments. As already described, the Globus Toolkit and the Cactus Code were chosen to provide a grid infrastructure and application laboratory. In addition a Cactus Portal was installed at one site to provide easy deployment of Cactus across the testbed.

3.1 Globus Metacomputing Toolkit

The access and security infrastructure is currently based on the Globus Metacomputing Toolkit [13]. Globus is a well known middleware for grid computing that is being deployed at many US academic supercomputer sites, and

a growing number of sites worldwide. It is published under the Globus Toolkit Public License (GTPL). This public license allows free public and commercial use and redistribution of the Globus Toolkit (including its source code).

Globus is designed to offer features such as uniform access to distributed resources with diverse scheduling mechanisms, information service for resource publication, discovery, and selection, and enhanced performance through multiple communication protocols [15]. Additionally, Globus provides a uniform security infrastructure and command-line tools based thereupon for remote file management, information retrieval and job management (`gsi-ftp`, `gsi-ssh`, `globusrun`, `grid-info-search`, etc.) Libraries provide access to specific parts of the Globus infrastructure, like resource management (GRAM) or input/output (`globus-io`).

With these features Globus provides a good basis for the EGrid testbed. As described later, Globus was adapted to work with the CCS scheduling system of the HPCLine at PC². An earlier port to the `prun` scheduler [3] was used to run Globus on the Amsterdam DAS-Cluster. The open interfaces of Globus make such integration of diverse hardware possible.

3.2 Cactus Code and Computational Toolkit

Cactus (www.cactuscode.org) is an open source problem solving environment designed for scientists and engineers. It has a modular structure which easily enables parallel computation across different architectures and collaborative code development between different groups. Cactus originated in the academic research community, where it was developed and used over many years by a large international collaboration of physicists and computational scientists.

The name Cactus comes from the design of a central core (or "flesh") which connects to application modules (or "thorns") through an extensible interface. Thorns can implement custom-developed scientific or engineering applications, such as computational fluid dynamics. Other thorns from a standard computational toolkit provide a range of computational capabilities, such as parallel I/O, data distribution, or checkpointing.

Cactus runs on many architectures. Applications, developed on standard workstations or laptops, can be seamlessly run on clusters or supercomputers. Cactus provides easy access to many cutting edge software technologies being developed in the academic research community, including the Globus Metacomputing Toolkit, HDF5 parallel file I/O, the PETSc scientific library, adaptive mesh refinement, web interfaces, and advanced visualization tools.

Cactus can easily be configured to use the Globus device for MPICH, meaning that applications using MPI for

communication (i.e. using the standard Cactus driver) are trivially Grid-enabled.

3.3 Cactus Code Portal

Although a grid-enabled version of Cactus can be easily compiled using the Globus MPICH communication device, it is not trivial to deploy Cactus, or any other application on the grid. Using multiple resources involves installing or compiling executables on different machines, keeping source code and parameter files consistent, managing authentication issues, and keeping up to date with software installations. More importantly, the end users of this technology must learn a new set of commands and a new computing paradigm.

To make grid computing more manageable for computational scientists, and intuitive for the end user, a *Cactus Portal* has been developed in the US a part of an NSF funded project for establishing an *Astrophysics Simulation Collaboration*. The Cactus Portal, implemented in a standard web browser, allows users to access all their grid-enabled computational resources through a single portal password. The portal shows all the resources a user has access to, which have the required grid software installed and deployed. Through the portal, users can select a machine, or a number of machines, on which a chosen version of Cactus is automatically installed, compiled, and executed either interactively, or through an automatically generated batch script. The portal contains options to compile and deploy a Globus enabled version of Cactus, optionally distributed across any number of machines.

3.4 Required Software

Equipping the testbed with the Globus Toolkit required all sites to install Globus 1.1.4, MPICH 1.2.1 with the Globus device MPICH-G2, and the Grid Security Infrastructure (GSI) tools GSIFTPD and GSISSHD. These packages require the prior installation of LDAP and the Secure Socket Layer (SSL). Note that although Globus 1.1.4 was installed, the libraries are only needed for compilation with MPICH-G2, and it was sufficient to deploy an earlier version of Globus. For locating the resources in the testbed, a *Grid Information Index Service* (GIIS) had to be set up, running on three redundant sites. Installation of the Cactus Code required, for the chosen applications, an installation of the HDF5 hierarchical data format of at least version 1.3.30 that included recent enhancements for streaming data.

Running a Cactus Portal for a testbed requires a server site to install a secure webserver and a Java servlet development kit. The EGrid portal used a secure Apache Web Server, and Tomcat.

To enable the testbed to be utilised in a short time scale, each site set up a local *demo* user, with all the testbed users being mapped to this demo user through the Globus gridmap files. This setup will be improved for future use of the testbed.

4 EGrid Testbed Working Group

The *Testbed and Applications* Working Group as part of the European Grid Forum aims to support the establishment of a European Grid testbed, particularly through a series of application experiments designed to fully exploit and test the Grid infrastructure. This testbed infrastructure provides resources to perform small to large scale Grid experiments as needed in the process of standardization, development and evaluation of Grid related software. This WG therefore represents a central part in the EGrid efforts to foster Grid related software developments.

The basic components of a Grid testbed infrastructure are distributed compute resources, in the broad sense of CPU cycles, storage facilities, and so on. These resources are connected by diverse network connections. Additionally, a Grid testbed needs to provide a unified access and security infrastructure, as well as appropriate programming environments for Grid aware software development. Finally, the testbed should include maintenance and support mechanisms for the software developers and end users working in the testbed environment.

The current efforts of the WG focused around the presentation at the Supercomputing 2000 Conference¹. This confronted the working group with the challenge to set up a pan-European testbed infrastructure in a very short time frame.

The remaining parts of this section give an overview of the sites taking part in this challenge, of the testbed infrastructure established until now, and of the Grid demonstration scenarios planned for the SC2000 event.

4.1 Testbed Participants

Figure 1 shows the sites currently involved in the EGrid testbed. The Albert-Einstein-Institut [1] in Potsdam uses an SGI Origin 2000 with 32 processors for HPC and visualization purposes. Additionally an SGI Infinite Reality II Engine is attached to the machine which allows interactive visualization of large volumes of data.

The DAS (Distributed ASCI² Supercomputer) cluster [6] is located at four universities in the Netherlands (Vrije Universiteit in Amsterdam, University of Amsterdam, Delft

University of Technology, and Leiden University). It consists of 200 Pentium Pro nodes connected via Myrinet and uses `prun` [3] as the local queuing system.

The Supercomputing Center Brno [26] provides a SGI Origin 2000 with 40 processors and Infinite Reality Engine similar to the machine at the Albert-Einstein-Institut in Potsdam. The local scheduling system is NQE.

The SISTEC-DLR [8] provides its departmental compute server, a SUN E450 with Ultra-Sparc processors.

The Laboratory of Parallel and Distributed Systems of the MTA SZTAKI Computer and Automation Research Institute located in Budapest [20] provides their cluster with 29 nodes. The nodes have dual Pentium III 500 MHz FastEthernet. PBS is used as the local job manager.

The Poznań Supercomputing and Networking Center (PSNC) [24] provides three machines for the EGrid Testbed: a Power Challenge XL with 12 R8000 TFP processors, an Onyx2 Infinite Reality2 with 8 R10000 processors and a Cray T3E-900 with 8 processors. The Power-Challenge system is used for the primary GIISserver for the testbed.

The HPCLine Cluster from the Paderborn Center for Parallel Computing (PC²) [22] with 96 double processor Pentium II nodes is made available to the testbed. The locally developed queuing system is called CCS. It supports Advance Reservation in combination with Space Sharing.

The High Performance Computing Laboratory at the University of Lecce [19] provides a Compaq AlphaCluster ES40. The machine consists of 4 SMP nodes with 4 alpha EV67 cpus each.

The SP2 at the GMD-SCAI (Institute for Algorithms and Scientific Computing) [16] is used for the EGrid testbed activities. It currently consists of 34 thin nodes and 3 wide nodes based on a standard POWER2 architecture RS/6000 processors and uses an EASY job scheduler.

The Konrad-Zuse-Zentrum für Informationstechnik Berlin [27] provides its Cray T3E LC 384 with 392 processors for parallel applications for the testbed. For SC2000 a maximum of 136 processors can be allocated for one job. The local scheduling system is based on NQE.

4.2 Testbed Networks

The testbed network is based on the national high speed networks in each partner country and the European high speed research network TEN-155. The national high speed networks include: B-WIN which is maintained by the DFN³ [7], Polish national research and education network POL34/155 [23] maintained by the PSNC, national computer network for education and research in the Netherlands - SURFnet [21], CESNET [4] - research and education network in Czech Republic. All these networks are the par-

¹November 4 - 10 2000, Dallas, <http://www.sc2000.org/>

²Advanced School for Computing and Imaging

³Deutsches Forschungsnetz



Figure 1. EGrid Testbed Participants

ticipants of the TEN-155 [5], which represents the Current European Research Networking.

As the testbed is just starting its activities, the partners did not yet establish dedicated transatlantic connections to the US research networks, like Abilene or vBNS+. Thus, only the shared Dante and SURFnet connections were available for the demo at Supercomputing 2000.

4.3 SC 2000 Scenarios

The basic testbed infrastructure as described above can already be used by application developers and end users. This fact has been demonstrated by the EGrid Testbed and Application WG during the Supercomputing 2000 Conference.

At this conference, a computational science application framework, Cactus, will be used to perform a number of innovative simulation experiments that exploit the EGrid infrastructure, distributing single simulations across the entire computing environment. Such simulations test nearly every aspect of the Grid infrastructure, and put it to practical use. The fact that this is possible is demonstrated by the success

of the basic setup of the European Grid Testbed in an amazingly short period of time (roughly 10 weeks).

Furthermore, the testbed environment will be used by various groups of application programmers to demonstrate its usability for Grid aware software development and testing. There are two classes of applications, both based on Cactus, that have been used for these purposes. The first involves distributing a single simulation across multiple sites, using MPICH-G2 to distribute the computation across the sites. This style of simulation has been demonstrated with Cactus many times in past experiments. The second is a much more innovative simulation type, that is able to adapt itself *dynamically* to a real live Grid environment, which will have characteristics that change in time. For that, the Cactus toolkit application will be equipped with the ability to investigate the current Grid environment, and take advantage of other Grid resources that it discovers. A simulation that can do this will be able to exploit a Grid environment in many ways, from spawning off related simulations to redistributing its load across different resources as appropriate. In this specific instance, the Cactus simulation will query the central GIIS server that has information about the

state of the entire EGrid, find appropriate resources, and migrates itself autonomously from one compute resource to any available other one. We call this “THE CACTUS WORM”. The Cactus Worm is a demonstration of the basic building blocks needed to exploit a Grid environment, but also serves a very practical purpose itself. Grand Challenge style applications, which usually consume computing times in order of many hours or days, often run out of compute time, or run into maintenance shutdowns, and need to be checkpointed and restarted on alternative resources by hand. The Worm concept allows the application to discover free and available resources in the testbed environment by itself, and to autonomously migrate the running application to these resources.

To achieve this functionality, a resource discovery library from the University of Lecce [19] is used to allow Cactus to query the testbed information services, which provides information about current resource status and availability. Some simple brokerage is done to determine useful and spare resources, based on, available processors, load, memory, etc. Then, using the established Globus security infrastructure, the application spawns a new instance on the remote host. This new instance is contacted, and the “old” one creates a checkpoint, but then streams the checkpointed data over the network to the new instance, which in turn uses this information to bootstrap, and continues the simulation at the point it was left off by the original version. With this procedure, an exact copy of the simulation program is then running on the new resource.

The Cactus toolkit allows one to track and watch its simulation process by querying simulation status and data sets online. In this case, Cactus operates as a web and data server, and provides these information in HTTP and HDF5 formats. In order to allow external and stationary tools to follow the movement of the application over the testbed, some basic contact information like current hostname and port numbers are published by the bootstrapping Cactus into the testbed information service, and can be queried there centrally.

This complex scenario is using numerous Grid related features of the testbed: security infrastructure, information services, remote job control, distributed resource utilization, remote data access, and some more. That makes it a very interesting scenario to demonstrate the potentials of Grid computing, and also to demonstrate the successful exploitation of Grid technologies by the EGrid community.

5 Lessons Learned

Besides the problems encountered during the installation at the individual sites, a GIIS service for the EGrid testbed had to be established.

5.1 Establishing a GIIS server

GIIS Backend

We started from the Globus 1.1.4 GIIS implementation. It provides the basic functionality, however it requires manual insertion of all the internal nodes between the root of the tree (o=GRID) and the roots of individual resources that are registered with the GIIS. If the internal nodes are not inserted correctly, the tree is not browsable by a sequence of one-level search requests (used by LDAP browsers). In the testbed environment, where sites of different countries and organizations are registered, this configuration is difficult to manage and likely to contain errors.

We upgraded using the patch released by the Globus developers in September 2000. Unfortunately, this introduced a more serious bug causing search misses on several fairly frequent search patterns. We succeeded in tracking down the problem and submitted a fix that was further improved by the Globus developers [14]. The current version of the GIIS backend does not exhibit problems of this sort.

Platform Stability

During testing on SGI Origin 2000 (IRIX 6.5) we observed that the `slapd` process was trapped rather frequently in a loop waiting for a non-existent child, resulting in a fail of the entire GIIS. The problem was avoided by moving the service to Linux (2.2.16, debian/potato).

Response Time

Response time of the GIIS depends strongly on the current state of its cache. In the case of a cache hit, it responds in about 2–7 seconds. At the other extreme, if the cache entries have expired and the search scope requires all the testbed nodes to be queried, the time may rise to as much as 10 minutes.

Application Specific Information

We found that the current GRIS/GIIS implementation is rather inflexible with respect to adding custom information. The CACTUS WORM application publishes certain information on its current state. One way to do this is for a master Cactus Server to provide the information, but in some cases one would also like to publish information about the state of the Cactus job directly to some GRIS/GIIS. This was implemented by providing an LDIF generating script that has to be statically added to the GRIS configuration on all involved resources, with obvious performance and management drawbacks. This approach has been reported to be used by other testbeds as well. Future releases of Globus are expected to provide better support for publishing such information.

5.2 Site Reports

Most sites managed to install and deploy the grid and application software with only minor problems. Exemplarily reports are given for two of the sites in more detail.

Czech MetaCenter

The SGI Origin 2000 from the Supercomputing Centre of Masaryk University is included in the Czech grid-computing activity – project MetaCenter [25]. Since the security infrastructure of this project is based on Kerberos, translation from X.509 certificates to Kerberos tickets had to be written to support Kerberos authentication into batch systems and distributed filesystem AFS. Also GSI-SSH collided with Kerberos patches to standard SSH. Since this problem was encountered at several sites, it was decided to run GSI-SSH on a separate port on all sites of SC2000 demo testbed.

To prevent the queuing of parts of distributed runs it was decided to use jobmanager-fork for launching distributed runs. However, support of two batch systems (NQE, PBS) was also tested. Four relatively short shell scripts for job maintenance (submit, remove, poll, status query) had rewritten – it was not possible to use the supplied versions of these scripts without fundamental modifications.

Small problems with support of native SGI MPI occurred during Cactus installation. MPICH-G2 supports using native MPI implementations and cooperation of versions with and without native MPI support, but compilation of Cactus had to be done very carefully to allow this feature. Similar situation is with cooperation of 32 and 64 bit MPI implementations – it was verified using MPICH examples that these versions can cooperate together, but at the time of writing this report we are still not able to run 32 and 64 bit versions of Cactus together.

PC²- Paderborn Center for Parallel Computing

The HPCLine cluster with 96 nodes is managed by CCS [18] (Computing Center Software), which was developed at the PC² and is now used for everyday work. Unique features of CCS are the support for Advance Reservation together with Space Sharing. Thus requests can potentially get queued (e.g. if all nodes are in use), which induces problems for distributed runs with other sites using time sharing. By doing reservations for the projected time period of the distributed run this potential drawback is solved.

Since the Globus Toolkit did not support CCS, a new interface had to be written. Globus communicates to local queuing subsystems via four shell scripts for job submit, job remove, job poll, and getting the current status of job queues. These scripts have to be implemented only once

for each batch subsystem and therefore they are partially undocumented, which certainly complicates the task of adapting them to CCS or any other new queuing system. In addition to the standard features of CCS like starting a batch job, reservations can be done and previously done reservations are usable by users via Globus commands.

The installation of all other required software packages and tools proceeded without any further problems. Due to the configuration of a cluster with a workstation as frontend and the nodes as backends a GSI-SSH-connection is only permitted to the frontend-machine. This is sufficient, because the filesystem is shared between the nodes and the frontend via NFS. For login to individual nodes `rlogin` is used, which requires no password and thereby suits well with the concept of certificate based login on the frontend.

At the beginning minor problems were based on a special feature of CCS. Generally all output to `stdout` and `stderr` is written in 10k blocks (and of course after job completion) for better performance. Cactus starts a user interface for simulation control as a `httpd` server and prints the URL to `stdout`. Now this information is kept in the buffer which makes the user think that Cactus does not work. To fix that, CCS was modified to return output immediately.

Further problems were based on the security infrastructure of clusters. Cactus is started on each cluster node and thereby communicates with other Cactus instances at other sites via MPICH-G2 point-to-point socket communication. Firewalls or portmappers at all sites must be configured properly to accept direct connections from other machines of the testbed by adding the IP-numbers to the appropriate configuration files.

During the startup of the distributed run, all involved machines receive the FQHN (Full Qualified Host Name) of the others from MPICH-G2. Entries generally consist of an IP number and the FQHN (hostname + domainname).

Now in some sites the domainname is left out e.g. due to system administration history. Therefore only the hostname is transferred, which is obviously not enough to contact a machine. The solution to that is to use an environment variable `GLOBUS_HOSTNAME` containing the FQHN. Note that in a cluster this variable must be set on every node individually.

6 Conclusions

In this paper we presented some of the lessons learned by EGrid's Testbed Working Group during the process of setting up one of the first pan-European grids. The fact that this could be established from scratch during a period of ten weeks, across a dozen sites in many countries, ending with a highly complex, working Grid testbed with real applications, is certainly one of the most important results of this initiative. It is very clear that simply installing the in-

frastructure at each site is far from sufficient to enable a working Grid: close coordination of the sites, and thorough testing with a real application designed to exploit the Grid resources, has been essential in making it work.

It also became clear that many of the problems we encountered only become visible in a grid scenario of this scale and not in smaller installations between just a few sites or within one single institution. This is the reason why large testbed installations are required to accompany the future development of grid technologies, and at least as importantly, why they must be tested thoroughly with working applications that exploit many aspects of the installed Grid infrastructure. It should also be considered that the focus of research in this area is shifting away from basic software infrastructure towards higher level services. Therefore, it is advisable to have at least some testbeds like the one described in this paper as permanent installations.

We have also sought to illustrate a few concepts and techniques that we feel will become important in Grid computing in the future. In particular, we showed that if a simulation code is endowed with the ability to query the central GIIS server, allowing it to discover resources available to it, very interesting capabilities are immediately possible. If the code is also able then to communicate with other simulations, it can then take advantage of a highly dynamic and changing Grid environment in novel ways.

In this case we illustrated the possibilities with an example application, which we called the “Cactus Worm”. The Worm is able to run on a machine in the Grid, query the GIIS to find out what resources are available to it, choose one, start up a remote process on another machine, stream the contents of memory to that remote process, register the new location with a central server, and shut down the original process. At each stage of the migration from site to site, the simulation can be interactively connected to, and controlled and visualized. Although such “worm” capabilities are useful in themselves, they more importantly illustrate future Grid scenarios where simulation codes can dynamically adapt themselves to their environment, take advantage of new resources as they are needed, and report the current state to a central server. We find these capabilities to be very exciting indeed!

7 Acknowledgements

It is a pleasure to acknowledge the help and advice of many of our colleagues around the world who helped to make this work possible. Many people from NCSA, Argonne, and ISI provided their expertise and energy, most notably Greg Daues, Steven Fitzgerald, Ian Foster, Shridhar Gullapalli, Shelley Henderson, Carl Kesselman, Scott Koranda, Jason (“Billy Bob”) Novotny, John Shalf, and Von Welch. We are also grateful to the folks at Manch-

ester, Stuttgart, and Sun, who graciously provided support and space from within their booths at SC2000 during the demonstrations.

References

- [1] <http://www.aei-potsdam.mpg.de/>.
- [2] G. Allen, W. Benger, T. Goodale, H. Hege, G. Lanfermann, A. Merzky, T. Radke, and E. Seidel. The Cactus Code: A Problem Solving Environment for the Grid. In *Proc. High Performance Distributed Computing (HPDC-2000)*, pages 253–260. IEEE Computer Society, 2000.
- [3] H. Bal et al. The Distributed ASCI Supercomputer Project. *Operating Systems Review*, 34(4):76–96, Oct. 2000.
- [4] <http://www.cesnet.cz/>.
- [5] <http://www.dante.net/ten155/>.
- [6] <http://www.cs.vu.nl/das/>.
- [7] <http://www.dfn.de/>.
- [8] <http://www.sistec.dlr.de/>.
- [9] <http://www.egrid.org/>.
- [10] G. E. Fagg, K. Moore, J. J. Dongarra, and A. Geist. Scalable Network Information Processing Environment (SNIPE). In *SC’97*, Nov. 1997. Online at <http://www.supercomp.org/sc97/proceedings/>.
- [11] I. Foster and C. Kesselman. Globus: A Metacomputing Infrastructure Toolkit. *Int. Journal of Supercomputer Applications*, 11(2):115–128, 1997.
- [12] I. Foster and C. Kesselman, editors. *The GRID: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 1998.
- [13] <http://www.globus.org/>.
- [14] http://www.globus.org/mail_archive/discuss/threads.html.
- [15] The Globus Project. *Globus Quick Start Guide*, Dec. 1999.
- [16] <http://www.gmd.de/SCAI/>.
- [17] A. S. Grimshaw, W. A. Wulf, and the Legion team. The Legion Vision of a Worldwide Virtual Computer. *Commun. ACM*, 40(1), January 1997.
- [18] A. Keller, M. Brune, and A. Reinefeld. Resource management for high-performance PC clusters. *Lecture Notes in Computer Science*, 1593, 1999.
- [19] <http://www.informatica.unile.it/laboratori/lab-hpc/>.
- [20] <http://www.lpds.sztaki.hu/>.
- [21] <http://www.surfnet.nl/en/>.
- [22] <http://www.uni-paderborn.de/pc2/>.
- [23] <http://www.man.poznan.pl/pol34/english/>.
- [24] <http://www.man.poznan.pl/>.
- [25] M. Ruda and L. Matyska. Security Infrastructure of the MetaCenter. In *ISThmus 2000 - Research and Development for the Information Society*, pages 419–426, May 2000.
- [26] <http://www.ics.muni.cz/scb/>.
- [27] <http://www.zib.de/>.