# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

**Appendices**

# Acknowledgments

Grid computing is an extremely powerful, though complex, research tool. The development of the Grid Technology Cookbook is an outreach effort targeted at motivating and enabling research and education activities that can benefit from, and further advance, grid technology. The scope and level of information presented is intended to provide an orientation and overview of grid technology for a range of audiences, and to promote understanding towards effective implementation and use.

This first version of the Grid Technology Cookbook was initiated through startup support from SURA (Southeastern Universities Research Association) and the Open Science Grid, and brought to completion with additional funding through a U.S. Army Telemedicine & Advanced Technology Research Center (TATRC) grant to SURA. While this support was critical to the development of this first version, the Grid Technology Cookbook is a community-driven and participatory effort that could not have been possible without numerous contributions of content and peer review from the individuals listed here.

In addition, creating a first version of a work of this type can be particularly challenging. Everything from determining the initial outline, to integration of content, to review of final material begins as a grand vision that is then tempered by the realities of busy schedules, shifting priorities and complicated by deadlines. We especially appreciate the commitment and perseverance of all contributors to version 1, and look forward to building on this effort for version 2, as resources permit. If you would like to support or contribute to future versions of the Cookbook, please contact the co-editors.

## Sponsors

**SURA**
Southeastern Universities
Research Association
www.sura.org

Established in 1980 as a 501(c)3 membership association, SURA's membership is now comprised of 63 research universities located in 16 southern US states plus the District of Columbia. SURA's broad mission is to foster excellence in scientific research, to strengthen the scientific and technical capabilities of the nation and of the Southeast, and to provide outstanding training opportunities for the next generation of scientists and engineers. SURA maintains several active programs including; management of the DOE funded Jefferson National Laboratory, the SURA Coastal Ocean Observing and Prediction (SCOOP) program, a technology transfer and commercialization program, regional optical network development initiatives, and SURAgrid.

SURAgrid is a highly collaborative regional grid computing initiative that evolved from the NSF Middleware Initiative (NMI) Integration Testbed program that SURA managed as part of the NMI-EDIT Consortium funded by NSF Cooperative Agreement 02-028, ANI-0123937. The SURAgrid infrastructure has been developed over the past several years through investments by SURA and the growing number of universities that are active participants and contributors of computational resources to SURAgrid. To learn more about SURAgrid visit www.sura.org/SURAgrid.

**TATRC**
The Telemedicine and
Advanced Technology
Research Center
http://www.tatrc.org

The Telemedicine and Advanced Technology Research Center (TATRC), a subordinate element of the United States Army Research and Materiel Command (USAMRMC), is charged with managing core Research Development Test and Evaluation (RDT&E) and congressionally mandated projects in telemedicine and advanced medical technologies. To support its research and development efforts, TATRC maintains a productive mix of partnerships with federal, academic, and commercial organizations. TATRC also provides short duration, technical support (as directed) to federal and defense agencies; develops, evaluates, and demonstrates new technologies and concepts; and conducts market surveillance with a focus on leveraging emerging technologies in healthcare and healthcare support. Ultimately, TATRC's activities strive to make medical care and services more accessible to soldiers, sailors, marines, and airmen; reduce costs, and enhance the overall quality of military healthcare.

The USAMRMC's telemedicine program, executed by the TATRC, applies medical expertise, advanced diagnostics, simulations, and effector systems integrated with information and telecommunications enabling medical assets to operate at a velocity that supports the requirements of the Objective Force. The program leverages, adapts, and integrates medical and commercial/military non-medical technologies to provide logistics/patient management, training devices/systems, collaborative mission planning tools, differential diagnosis, consultation and knowledge sharing. These capabilities enhance field medical support by improving planning and enabling real time "what-if" analysis. Specifically, this program will:

- Reduce medical footprint and increases medical mobility while ensuring access to essential medical expertise & support
- Incorporate health awareness into battlespace awareness
- Improve the skills of medical personnel and units
- Improve quality of medical/ surgical care throughout the battlespace

**iVDGL**
International Virtual Data Grid Laboratory
www.ivdgl.org

The **iVDGL** (**i**nternational **V**irtual **D**ata **G**rid **L**aboratory) was tasked with establishing and utilizing an international Virtual-Data Grid Laboratory (iVDGL) of unprecedented scale and scope, comprising heterogeneous computing and storage resources in the U.S., Europe and ultimately other regions linked by high-speed networks, and operated as a single system for the purposes of interdisciplinary experimentation in Grid-enabled, data-intensive scientific computing.

Our goal in establishing this laboratory was to drive the development, and transition to every day production use, of Petabyte-scale virtual data applications required by frontier computationally oriented science. In so doing, we seized the opportunity presented by a convergence of rapid advances in networking, information technology, Data Grid software tools, and application sciences, as well as substantial investments in data-intensive science now underway in the U.S., Europe, and Asia. Experiments conducted in this unique international laboratory influence the future of scientific investigation by bringing into practice new modes of transparent access to information in a wide range of disciplines, including high-energy and nuclear physics, gravitational wave research, astronomy, astrophysics, earth observations, and bioinformatics.

iVDGL experiments also provided computer scientists developing data grid technology with invaluable experience and insight, therefore influencing the future of data grids themselves. A significant additional benefit of this facility was that it empowered a set of universities who normally have little access to top tier facilities and state of the art software systems, hence bringing the methods and results of international scientific enterprises to a diverse, world-wide audience.

iVDGL was supported by the National Science Foundation.

**OSG**
Open Science Grid
www.opensciencegrid.org

The Open Science Grid is a national production-quality grid computing infrastructure for large scale science, built and operated by a consortium of U.S. universities and national laboratories. The OSG Consortium was formed in 2004 to enable diverse communities of scientists to access a common grid infrastructure and shared resources. Groups that choose to join the Consortium contribute effort and resources to the common infrastructure.

The OSG capabilities and schedule of development are driven by U.S. participants in experiments at the Large Hadron Collider, currently being built at CERN in Geneva, Switzerland. The distributed computing systems in the U.S. for the LHC experiments are being built and operated as part of the OSG. Other projects in physics, astrophysics, gravitational-wave science and biology contribute to the grid and benefit from advances in grid technology. The services provided by the OSG will be further enriched as new projects and scientific communities join the Consortium.

The OSG includes an Integration and a Production Grid. New grid technologies and applications are tested on the Integration Grid, while the Production Grid provides a stable, supported environment for sustained applications. Grid operations and support for users and developers are key components of both grids. The core of the OSG software stack for both grids is the NSF Middleware Initiative distribution, which includes Condor and Globus technologies. Additional utilities are added on top of the NMI distribution, and the OSG middleware is packaged and supported through the Virtual Data Toolkit.

The OSG is a continuation of Grid3, a community grid built in 2003 through a joint project of the U.S. LHC software and computing programs, the National Science Foundations' GriPhyN and iVDGL projects, and the Department of Energy's PPDG project.

To learn more about the  OSG we suggest you visit the Consortium section, OSG@Work, the Twiki and document repository

# Editors

**Mary Fran Yafchak**
Southeastern Universities
Research Association
IT Program Coordinator

Mary Fran Yafchak is the IT Program Coordinator for the Southeastern Universities Resource Association (SURA) and the project manager for SURAgrid, a regional grid initiative for inter-institutional resource sharing. As part of SURA's IT Initiative, she works to further the development of regional collaborations as well as synergistic activities with relevant national and international efforts. In current and past roles, Mary Fran has enabled and supported diverse initiatives to develop and disseminate advanced network technologies. She managed the NSF Middleware Initiative (NMI) Integration Testbed Program for SURA during the first three years of the NMI, in partnership with Internet 2, EDUCAUSE, and the GRIDS Center. She has led the development of several educational workshops for the SURA community, and previously designed and delivered broad-based Internet training as part of a start-up team for the NYSERNet Information Technology Education Center

(NITEC). Mary Fran holds a B.S. in Secondary Education/English from SUNY Oswego and an M.S. in Information Resource Management from Syracuse University.

**Mary Trauner**
SURA, ViDe
Senior Research Scientist, Consultant

Recently retired from her position as Senior Research Scientist at Georgia Tech, Mary Trauner is a consultant with several groups, including past Steering Committee chair of the Video Development Initiative (ViDe) and consultant with SURA on revision 1 of this resource and infrastructure support for SURAgrid.

With an educational background in both computer science and atmospheric sciences, Mary's work has spanned "both worlds" to understand and model physical processes on large scale, parallel computing systems. She has also spent the last decade studying and deploying many digital video and collaborative technologies. Her most recent affiliations include the ViDe Steering Committee, the Internet2 Commons Management Team, the Georgia Tech representative to the Coalition for Academic Scientific Computation(CASC), and the Georgia HPC task force. Mary has participated in the development of a broad range of technology tutorials, user guides, and whitepapers including the ViDe Videoconference Cookbook, ViDe Data Collaboration Whitepaper, Georgia Tech HPC website and tutorials, and an interactive tutorial on building and optimizing parallel codes for supercomputers.

# Contributors

**Mark Baker**
University of Reading
Research Professor

Mark Baker is a Research Professor of Computer Science at the University of Reading in the School of Systems Engineering.

His research interests are related to parallel and distributed systems. In particular, he is involved in research projects related to the Grid, message-oriented middleware, the Semantic Web, Web Portals, resource monitoring, and performance evaluation and modelling. For more information, see http://acet.rdg.ac.uk/~mab/.

Mark and Dan Katz wrote the “Standards and Emerging Technologies” section.

**Russ Clark**
Georgia Institute of Technology, College of Computing
Research Scientist

Russ Clark's research and teaching interests include: real-time network management techniques, network visualization, and applications for wireless/mobile networks with the IP Multimedia Subsystem (IMS). He currently holds a joint appointment with the College of Computing and the Office of Information Technology Academic and Research Technologies Group (OIT-ART) at the Georgia Institute of Technology. This work includes a focus on network management in the GT Research Network Operations Center (GT-RNOC).

Russ received the PhD in Computer Science from Georgia Tech in 1995 and has extensive experience in both industry and academia.

**Gary Crane**
Southeastern Universities
Research Association
Director, IT Initiatives

Gary Crane is the Director of Information Technology Initiatives for the Southeastern Universities Research Association (SURA). Gary is responsible for the development of SURA’s information technology projects and programs (http://sura.org/programs/it.html), including SURAgrid, a regional grid development initiative and partnerships with IBM and Dell that are facilitating the acquisition of high performance computing systems by SURA members. Gary holds B.S.E.E. and M.B.A. degrees from the University of Rochester.

**Vikram Gazula**
University of Kentucky
Center for Computational
Sciences

Vikram Gazula is the Senior IT Manager for the Center for Computational Sciences at the University of Kentucky. He is responsible for the development and deployment of grid based projects and programs. He has more than 10 years of experience in HPC systems. His core interests are in the field of distributed computing and resource management of large scale heterogeneous information systems. He also manages various local and virtual technical teams deploying grid projects at HPC centers across the U.S.

Vikram holds an engineering degree in Computer Science from Kuvempu University, India and a Masters in Computer Science from the University of Kentucky.

**James Patton Jones**
JRAC, Inc.
President and CEO

Recognized internationally as an expert in HPC/Grid workload management and batch job scheduling, James Jones has contributed chapters to four textbooks, authored six computer manuals, written 25 technical articles/papers, and published several non-technical books. He served as co-architect of NASA s Metacenter (prototype Computational Grid) and co-architect of the Department of Defense MetaQueueing Grid, and subsequently assisted with the implementation of both projects. James managed the business aspects of the Portable Batch System (PBS) team from 1997 thru 2000. (PBS is a flexible workload management, batch queuing, and job scheduling software system for computer clusters and supercomputers. See also www.pbspro.com.) James created the Veridian PBS Products Dept in 2000, and in 2003 spun-out the profitable business unit, co-founding Altair Grid Technologies, the PBS software development company. He then served in worldwide technical business development roles, growing the global PBS business. In late 2005 James founded his third company, JRAC, Inc., publically focusing on HPC and Grid Consulting, and quitely developing the next "amazing killer app". (www.jrac.com)

**Hartmut Kaiser**
Louisiana State University
Center for Computation &
Technology (CCT)

After 15 interesting years that Hartmut Kaiser spent working in industrial software development, he still tremendously enjoys working with modern software development technologies and techniques. His preferred field of interest is the software development in the area of object-oriented and component-based programming in C and its application in complex contexts, such as grid and distributed computing, spatial information systems, internet based applications, and parser technologies. He enjoys using and learning about modern C programming techniques, such as template based generic and meta-programming and preprocessor based meta-programming.

**Daniel S. Katz**
Louisiana State University
Assistant Director for Cyberinfrastructure Development
Associate Research Professor

Daniel S. Katz is Assistant Director for Cyberinfrastructure Development (CyD) in the Center for Computation and Technology (CCT), and Associate Research Professor in the Department of Electrical and Computer Engineering at Louisiana State University (LSU). Previous roles at JPL, from 1996 to 2006, include: Principal Member of the Information Systems and Computer Science Staff, Supervisor of the Parallel Applications Technologies group, Area Program Manager of High End Computing in the Space Mission Information Technology Office, Applications Project Element Manager for the Remote Exploration and Experimentation (REE) Project, and Team Leader for MOD Tool (a tool for the integrated design of microwave and millimeter-wave instruments). From 1993 to 1996 he was employed by Cray Research (and later by Silicon Graphics) as a Computational Scientist on-site at JPL and Caltech, specializing in parallel implementation of computational electromagnetic algorithms.

His research interests include: numerical methods, algorithms, and programming applied to supercomputing, parallel computing, cluster computing, and embedded computing; and fault-tolerant computing. He received his B.S., M.S., and Ph.D degrees in Electrical Engineering from Northwestern University, Evanston, Illinois, in 1988, 1990, and 1994, respectively. His work is documented in numerous book chapters, journal and conference publications, and NASA Tech Briefs. He is a senior member of the IEEE, designed and maintained (until 2001) the original website for the IEEE Antenna and Propagation Society, and serves on the IEEE Technical Committee on Parallel Processing's Executive Committee, and the steering committee for the IEEE Cluster and IEEE Grid conference series.

Dan and Mark Baker wrote the "Standards and Emerging Technologies" section.

**Gurcharan Khanna**
Rochester Institute of Technology
Director of Research Computing

Gurcharan has a special interest and expertise in innovative collaboration tools, the social aspects of technologically connected communities, and the cyberinfrastructure required to support them. He started the first Access Grid nodes at RIT and Dartmouth College. He is a member of the ResearchChannel Internet2 Working Group and helped start the Internet2 Collaboration SIG. He serves as a member of the Board and Chair of the Middleware Group of the NYSGrid, an advanced collaborative cyberinfrastructure for supporting and enhancing research and education.

Gurcharan is currently Director of Research Computing at Rochester Institute of Technology, reporting to the Vice President for Research. He provides the leadership and vision to foster research at RIT by partnering with researchers to support advanced research technology resources in computation, collaboration, and community building. Gurcharan created the Interactive Collaboration Environments Lab housed in the Center for Advancing the Study of Cyberinfrastructure at RIT, as a teaching and learning, research and development, practical application, and evaluative studies lab.

Gurcharan was a Member of the Real Time Communications Advisory Group, Internet2 from 2005-2006. He was formerly Associate Director for Research Computing at Dartmouth College. He has served as a consultant on several grant proposals to design and implement multipoint collaborative conferencing systems and twice as a panelist for the NSF Advanced Networking Infrastructure Research Program (2001-2002).

His background includes teaching in the Geography Department and supervising the UNIX Consulting Group in Academic Computing at the University of Southern California from 1992-1995 and teaching and research at the University of California, Berkeley from 1980-1992, where he received his Ph.D. in anthropology.

**Bockjoo Kim**
University of Florida
Assistant Scientist,
Department of Physics

Bockjoo Kim completed his undergraduate work at Kyungpook National University and received his MS and PhD (High Energy Physics) from the University of Rochester in 1994. His research career includes positions at the University of Rochester, University of Hawaii, Fermilab, Istituto Nazionale di Fisica Nucleare (Italy), Seoul National University, and now the University of Florida. He is a member of the CMS (Compact Muon Solenoid) team.

**Warren Matthews**
Academic & Research Technologies
Georgia Institute of Technology
Research Scientist II

Warren Matthews is a research scientist II in the Office of Information Technology (OIT) at the Georgia Institute of Technology.

He helps to run the campus network, the Southern Crossroads (SOX) gigapop and the Southern Light Rail (SLR). He also works with other researchers to coordinate international networking initiatives and chairs the Internet2 Special Interest Group on Emerging NRENs.

Since obtaining his PhD in particle physics, he has been active in many areas of network technology. His current interests include network performance, K-12 outreach and bridging the Digital Divide.

**Shawn McKee**
University of Michigan
Assistant Research Scientist, School of Information

**Russ Miller**
State University of New York, Buffalo
Distinguished Professor, Computer Science and Engineering

**Jerry Perez**
Texas Tech University
Research Associate
High Performance Computing Center

Jerry Perez is a Research Associate for the High Performance Computing Center (HPCC) at Texas Tech University. His experience also includes adjunct teaching in Management Information Systems, Grid Computing, Computer Programming, and Systems Analysis for Wayland Baptist University. He has 5 years corporate experience as Senior Product Engineer Technician at Texas Instruments. He holds a Bachelors of Science in Organizational Management, an M.B.A. and is concluding work on his Ph.D in Information Systems at Nova Southeastern University (NSU). Jerry has authored or co-authored several papers on the implementation of grids to support a variety of specific application areas including: Sybase Avaki Data Grid, parallel Matlab, grid enabled SAS, SRB Data Grid, parallel graphics rendering, theoretical mathematics, cryptography, digital rights, grid security, physics applications, bioinformatics data solutions, computational chemistry, high performance computing, and engineering simulations. Other synergistic activities include: sole designer, developer, deployer, and manager of a multi-organizational campus-wide compute grid at TTU (TechGrid); lead for deployment of commercial grid technologies with TTU Business, Physics, Computer Science, Mass Communications, Engineering, and Mathematics departments; Director of Distance Learning Technology video technology group for HiPCAT (High Performance Computing Across Texas) Consortium; collaboration in SURAgrid (Southeastern Universities Research Association Grid), including contribution to the white paper, SURAgrid Authorization/Authorization: Concepts & Technologies, and Chair of the SURAgrid grid software stack committee. Jerry is an international grid lecturer who leads grid talks to discuss development and deployment of desktop computational grids as well as Globus based regional grids. Jerry s most recent grid talks were presented at Sybase TechWave in Las Vegas, GGF 12, OGF18 and 19; Tecnológico de Monterrey in Mexico City; EDUCAUSE Regional Conference; and he was invited to give a one day seminar about building and managing campus grids at the EDUCAUSE National Conference 2007 in Seattle Washington.

**Ruth Pordes**
Executive Director, Open Science Grid
Associate Head, Fermilab Computing Division

Ruth Pordes is the executive director of the Open Science Grid  a consortium that was formed in 2004 to enable diverse communities of scientists to access a common grid infrastructure and shared resources. Pordes is an associate head of the Fermilab Computing Division, with responsibility for Grids and Communication, and a member of the CMS Experiment with responsibility for grid interfaces and integration. She has worked on a number of collaborative or  joint  computing projects at Fermilab, as well as been a member of the KTeV high-energy physics experiment and an early contributor to the computing infrastructure for the Sloan Digital Sky Survey. She has an M.A. in Physics from Oxford University, England.

**Lavanya Ramakrishnan**
Indiana University, Bloomington
Graduate Research Assistant

Lavanya Ramakrishnan’s research interest includes grid workflow tools, resource management, monitoring and adaptation for performance and fault tolerance. Lavanya is currently a graduate student at Indiana University exploring multi-level adaptation in dynamic web service workflows in the context of Linked Environments for Atmospheric Discovery(LEAD). Previously, she worked at the Renaissance Computing Institute where she served as technical lead on several projects including Bioportal/TeraGrid Science Gateway SCOOP, Virtual Grid Application Development Software(VGrADS). Lavanya is also co-PI of the NSF NMI project - A Grid Service for Dynamic Virtual Clusters that is investigating adaptive provisioning through container-level abstractions for managing grid resources.

**Jorge Rodriguez**
Florida International University
Assistant Professor, Physics

Dr. Jorge L. Rodriguez is a Visiting Assistant Professor of Physics at Florida International University in Miami Floria. His research interest include Grid computing and the physics of elementary particles. He is currently a member of the Compact Muon Solenoid (CMS) experiment at the Large Hadron Collider at CERN and works on Software and Computing as a member of the USCMS collaboration.

Previously, Jorge served as Deputy Coordinator for the International Virtual Data Grid Laboratory (iVDGL) and was a senior member in the Grid Physics Network (GriPhyN) project. GriPhyN and iVDGL together with other U.S. and European Grid and application communities formed Grid3, one of the first large scale international computational grids. The effort lead directly to the Open Science Grid (OSG) where Jorge also served as Co-Chair in several OSG committees.

Jorge was also the facilities manager for the University of Florida CMS Tier2 Center. The University of Florida Tier2 Center was one of the first and the largest prototype Tier2 Center in the country. It together with Caltech, UC San Diego and Fermi Lab were instrumental in developing the ongoing and successful U.S. Tier2 program which supports computing for the CMS and OSG application communities.

Jorge was born in Havana Cuba and now lives in South Florida with his wife and two kids. He teaches physics, exploits Grid computing for research in elementary particles and has time for little else.

**Alain Roy**
University of Wisconsin-Madison
Associate Researcher, Condor Project

Alain Roy is the Software Coordinator for Open Science Grid, where he leads the effort to build the VDT software distribution. He has been a member of the Condor Project since 2001. He earned his Ph.D. from the University of Chicago in 2001, where he worked on quality of service in a grid environment with the Globus project.

In his spare time he enjoys playing with his children and baking bread. He has trouble keeping his desk clean and hopes that this is a sign of the great complexity of his work instead of inherently disorganized thought. He has a secret desire to visit Pluto one day.

(Note from editor: Alain is a great cook and teacher, in the strict sense of the words. See his instructions for making crepes at Making Crepes.)

**Mary Trauner**
SURA, ViDe
Senior Research Scientist, Consultant

Recently retired from her position as Senior Research Scientist at Georgia Tech, Mary Trauner is a consultant with several groups, including past Steering Committee chair of the Video Development Initiative (ViDe) and consultant with SURA on revision 1 of this resource and infrastructure support for SURAgrid.

With an educational background in both computer science and atmospheric sciences, Mary’s work has spanned "both worlds" to understand and model physical processes on large scale, parallel computing systems. She has also spent the last decade studying and deploying many digital video and collaborative technologies. Her most recent affiliations include the ViDe Steering Committee, the Internet2 Commons Management Team, the Georgia Tech representative to the Coalition for Academic Scientific Computation(CASC), and the Georgia HPC task force. Mary has participated in the development of a broad range of technology tutorials, user guides, and whitepapers including the ViDe Videoconference Cookbook, ViDe Data Collaboration Whitepaper, Georgia Tech HPC website and tutorials, and an interactive tutorial on building and optimizing parallel codes for supercomputers.

**Judith Utley**
HPC and Grid Systems Analyst
IS Professional

Judith Utley is an information systems professional with 22 years experience in HPC systems analysis and administration, including 13 years with HPC and Linux cluster integration. Ms. Utley was co-lead for the NASA Metacenter project. She was a key member of the NASA Information Power Grid (IPG) project team, evaluating and modifying as needed state-of-the-art grid infrastructure toolkits to work well in the established production environment and contributing to grid plans, tutorials, user support and training. Ms. Utley, as a member of the IPG project, provided feedback to outside grid developers. Ms Utley was also the coordinator of this persistent NASA grid among eight NASA sites, training new grid administrators as new sites joined the NASA grid as well as represented IPG as a consultant to emerging grids. Ms Utley established the Production Grid Management Research Group in the Global Grid Forum (now the Open Grid Forum) and chaired this group for over three years. Her project management experience includes managing both local and distributed virtual technical teams as well as planning and coordinating international workshops in grid technology management. Ms. Utley has experience in business planning, marketing, sales, and technical consulting working with both government and commercial customers. Ms. Utley also contributed significantly to the commercialization of the PBS Pro product.

**Art Vandenberg**
Georgia State University
Director, Advanced Campus Services
Information Systems & Technology

Art Vandenberg has a Masters degree in Information & Computer Sciences from Georgia Institute of Technology, where he held various research, support and development roles from 1983 to 1997. As Director of Advanced Campus Services at Georgia State University, he evaluates and implements middleware infrastructure and research computing. Vandenberg was the Project Manager for Georgia State University’s Y2K inventory, analysis and remediation effort that included all of Georgia State’s business and students systems and processes, information technology and campus facilities. Vandenberg was the lead for Georgia State’s participation in the National Science Foundation Middleware Initiative (NMI) Integration Testbed Program (Southeastern Universities Research Association sub-award to NSF Contract #ANI-0123837) Supporting Research and Collaboration through Integrated Middleware. The NMI Integration Testbed was part of NFS’ overall effort to disseminate practices and solutions software for collaboration, directories, identity management and grid infrastructure. Vandenberg’s work with the NMI Testbed lead to the architecture and deployment of formal identity management practices for Georgia State. Current activities include grid middleware and collaboration with faculty researchers on high performance computing and grid infrastructure. Art is an active participant with SURA and the regional SURAgrid project. Art is co-PI with Professor Vijay K. Vaishnavi on an NSF Information Technology Research grant investigating a unique approach to resolving metadata heterogeneity for information integration and is a member of the Information Technology Risk Management Research Group at Georgia State.

**Mary Fran Yafchak**
Southeastern Universities Research Association
IT Program Coordinator

Mary Fran Yafchak is the IT Program Coordinator for the Southeastern Universities Resource Association (SURA) and the project manager for SURAgrid, a regional grid initiative for inter-institutional resource sharing. As part of SURA’s IT Initiative, she works to further the development of regional collaborations as well as synergistic activities with relevant national and international efforts. In current and past roles, Mary Fran has enabled and supported diverse initiatives to develop and disseminate advanced network technologies. She managed the NSF Middleware Initiative (NMI) Integration Testbed Program for SURA during the first three years of the NMI, in partnership with Internet 2, EDUCAUSE, and the GRIDS Center. She has led the development of several educational workshops for the SURA community, and previously designed and delivered broad-based Internet training as part of a start-up team for the NYSERNet Information Technology Education Center (NITEC). Mary Fran holds a B.S. in Secondary Education/English from SUNY Oswego and an M.S. in Information Resource Management from Syracuse University.

**Katie Yurkewicz**
Fermi National Accelerator Laboratory
Editor

Katie Yurkewicz was the founding editor of Science Grid This Week, a weekly newsletter about U.S. grid computing and its applications to all fields of science. In November 2006, she launched International Science Grid This Week, an expanded version of the original newsletter that informs the grid community and interested public about the people and projects involved in grid computing worldwide and the science that relies on it. In addition to editing SGTW and iSGTW, Katie worked in communications for the Open Science Grid until December 2006. Katie, who holds a Ph.D. in nuclear physics from Michigan State University, is now the US LHC communications manager at CERN in Geneva, Switzerland.

# Trademarks

Globus™, Globus Alliance™, and Globus Toolkit™ are trademarks held by the University of Chicago.

Sun® and Grid Engine® (gridengine®) are registered trademarks held by Sun Microsystems, Inc.

IBM® and Loadleveler® are registered trademarks held by the IBM Corporation.

The Internet2® word mark and the Internet2 logo are registered trademarks of Internet2.

Shibboleth® is a registered trademark of Internet2.

caBIG and cancer Biomedical Informatics Grid are trademarks of the National Institutes of Health

# Use of this material

This work is the intellectual property of SURA and the authors. Permission is granted for this material to be used for non-commercial, educational purposes with the stipulations below. To disseminate or republish otherwise requires written permission from SURA. (Please use the Contact Page for this purpose.)

- Incorporation of all or portions of the Cookbook into other electronic, online or hard copy works is not allowed.
- The online Cookbook may not be mirrored or duplicated without written permission of SURA, but all or portions of it may be linked to from other works as long as credit and copyright are clearly noted at the point of the link in the referencing work.
- Reproduction of the Cookbook as a whole is allowed in hard copy or offline electronic versions for non-profit educational purposes only and provided that this copyright statement appears on the reproduced materials and notice is given that the copying is by permission of the authors.

# Preface

## Why this guide?

Many universities and research organizations are actively planning and implementing Grid technology as a tool to enable researchers, faculty and students to participate more broadly in science and other collaborative research and academic initiatives. However, there are numerous technologies, processes, standards and tools included under the "Grid umbrella" and understanding these various elements, as well as their likely evolution, is critical to the successful planning and implementation of grid-based projects and programs. This community-driven "Grid Technology Cookbook" is intended to educate faculty and campus technical professionals about the current best practices and future directions of this technology to enable effective deployment and participation at local, regional and national levels.

There is immediate need within the advanced scientific application community for effective resources and references that illustrate the planning, deployment and usage of grid technologies. Supporters of the Grid Cookbook include recognized grid experts from various communities and organizations including SURAgrid, the Open Science Grid, the Louisiana State University Center for Computation and Technology (CCT), and the European Enabling Grids for E-Science (EGEE) project. Writing and review teams have been (and continue to be) drawn from these known supporters and also through a continued open Call for Participation to insure that this Grid Cookbook is broadly vetted, relevant, and useful.

The Grid Cookbook is made available freely over the Internet as an online-readable document and in hard copy at a small fee for cost recovery. The Grid Cookbook has been designed to serve as both a reference and a model for grid technology education (such as preparatory reading for seminars and classes); reproduction for non-profit educational purposes will be granted to encourage and increase dissemination. We also encourage its use to leverage the development and creation of additional educational opportunities within the community.

## Who is the audience?

This cookbook has been developed with three, possibly overlapping, audiences in mind:

| | |
|---|---|
| 🟩 | Beginners, higher level administrators, those just curious |
| 🟦 | Programmers or those ready to consider using grid services |
| 🟪 | Those considering or responsible for building a grid (for the first time) |
| 🟨 | General material of interest to all readers of the Cookbook |

This cookbook has been designed from general to specific, from introductory to advanced. The early sections provide a general introduction of the material. Later sections give actual programming examples and generic (and eventually real) installation examples. Depending on your experience level, here are some guidelines on sections that may be of most interest to you:

| | | |
|---|---|---|
| 🟨 | Acknowledgements | Please don't miss this section! Read up on who had a hand in writing and producing this resource. |
| 🟨 | Preface | This section covers the why, who, and how of getting the most out of your reading of the Cookbook. |
| 🟩 | Introduction | We start from the beginning with what a grid is, an overview of how grids work, what resources you're likely to find on a grid, and who can access grid resources. |
| 🟩 | History, Standards & Directions | Where are the standards? We discuss this in light of well-known initiatives that are developing standards in foundational areas such as grid architecture, scheduling, resource management, data access, and security. |
| 🟩 | What Grids Can Do For You | We describe the payoffs you will see using grids: access to resources, performance improvements, speedup of results, and collaboration enhancements. We also highlight trends in computational and networked services offered via grids and describe a future view of a ubiquitous "grid of grids". |
| 🟩 | Grid Case Studies | We present several examples of applications that benefit from the use of grids along with overviews of some multi-purpose grid initiatives. Both of these are intended to give you ideas on how such benefits can be realized within your own computational strategies. |
| 🟩 | Current Technology for Grids | We give an overview of the typical components found in grid architectures from user interface, to resource discovery and management, to grid system administration and monitoring. Pointers to popular grid products in each area are included. |
| 🟦 | Programming Concepts & Challenges | We present guidelines on how to work with specific grid services and toolkits, including programming examples. Scheduling resources, job submission (and monitoring and management), data access, security, workflow processing and network communications are covered. |
| 🟪 | Joining a Grid: Procedures & Examples | This section includes overviews of two grid initiatives that are open to new participants and provide an environment for peer-to-peer learning and support. In future versions of the Cookbook, we hope to add more detail on designing your own grid and grid-to-grid integration. |
| 🟦 | Typical Usage Examples | This section walks through several examples to show variety among grid applications and approaches to workflow and user interface. |
| 🟨 | Related Topics | Other related things are helpful, if not important, in understanding and deploying grids. Networks form the virtual bus that interconnects grid nodes. Knowing how to plan your manpower is key. These things can be found here. |

🟨 My Favorite Tips — This section provides an interactive space for readers to share tips and techniques for successful grid design, development and use.

🟨 Glossary — A number of excellent glossaries for grid technologies exist. We offer links to those resources as well as any additional terminology required for the use of this resource.

🟨 Appendices — In this section, we provide a full bibliography plus valuable peripheral topics such as resources for further reading and reference, links to grid software distributions, links to mailing lists and other interactive forums, and a brief introduction to benchmarks and performance.

# How to use this guide?

You should find this cookbook fairly straightforward to navigate. But lets go over a few of its features and tools:

### Toolbar

First, you are likely to notice the toolbar where you will find the usual suspects:

**Home** Return to the cookbook home or cover page.

**Previous** Go to the previous section of the cookbook (relative to where you are.)

**Next** Go to the next section of the cookbook (relative to where you are.)

**Print** Find out how to get a printed copy of the cookbook.

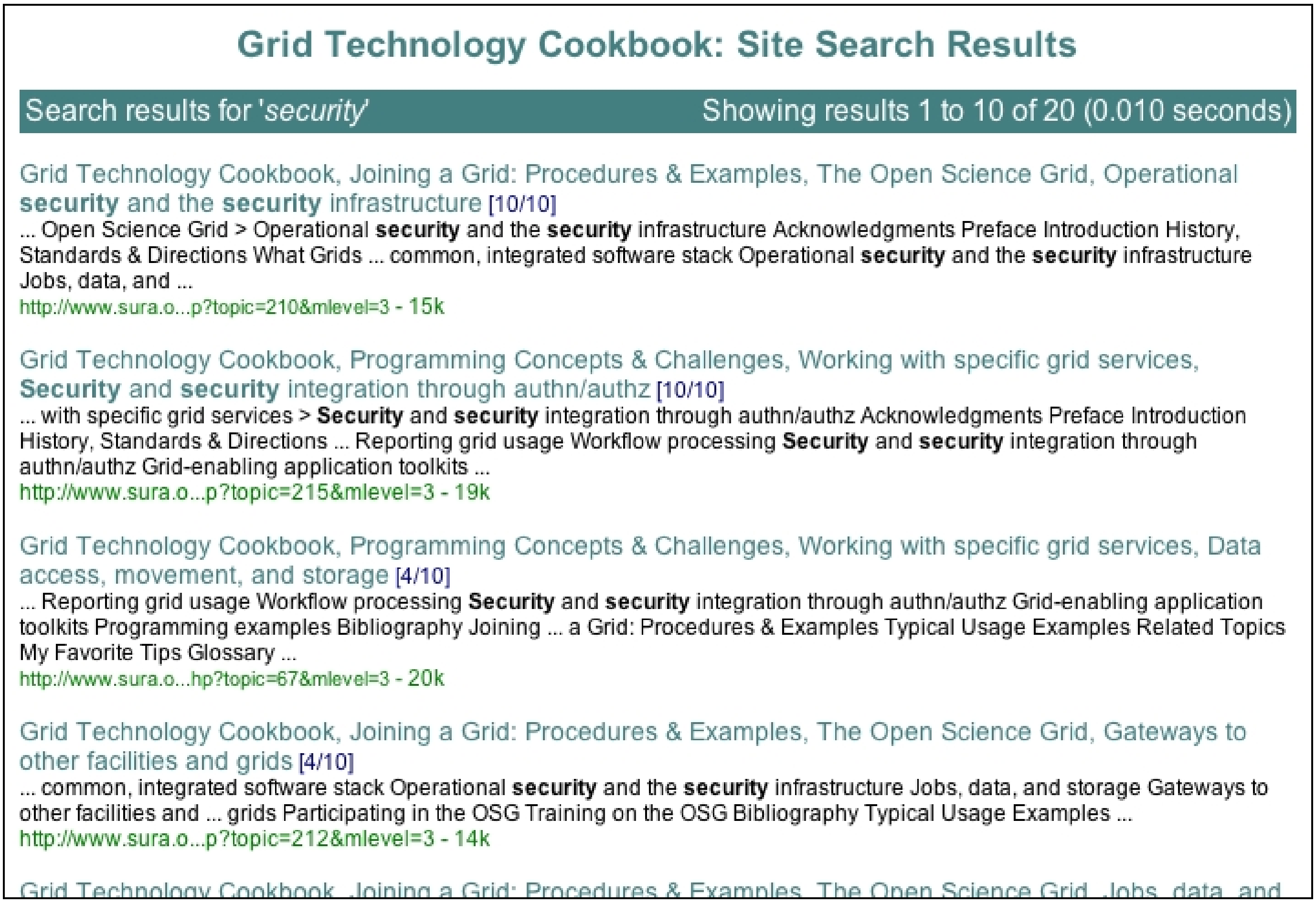
**Contact** Contact us or send feedback about the cookbook.

### Search

To use the Search tool (developed by iSearch), enter your search text into the box that appears in the right of the toolbar. Click on **Search**.

security

Upon entering your search criteria, you'll see a "Google-like" response:



Grid Technology Cookbook: Site Search Results

Search results for '*security*' Showing results 1 to 10 of 20 (0.010 seconds)

Grid Technology Cookbook, Joining a Grid: Procedures & Examples, The Open Science Grid, Operational **security** and the **security** infrastructure [10/10]
... Open Science Grid > Operational **security** and the **security** infrastructure Acknowledgments Preface Introduction History, Standards & Directions What Grids ... common, integrated software stack Operational **security** and the **security** infrastructure Jobs, data, and ...
http://www.sura.o...p?topic=210&mlevel=3 - 15k

Grid Technology Cookbook, Programming Concepts & Challenges, Working with specific grid services, **Security** and **security** integration through authn/authz [10/10]
... with specific grid services > **Security** and **security** integration through authn/authz Acknowledgments Preface Introduction History, Standards & Directions ... Reporting grid usage Workflow processing **Security** and **security** integration through authn/authz Grid-enabling application toolkits ...
http://www.sura.o...p?topic=215&mlevel=3 - 19k

Grid Technology Cookbook, Programming Concepts & Challenges, Working with specific grid services, Data access, movement, and storage [4/10]
... Reporting grid usage Workflow processing **Security** and **security** integration through authn/authz Grid-enabling application toolkits Programming examples Bibliography Joining ... a Grid: Procedures & Examples Typical Usage Examples Related Topics My Favorite Tips Glossary ...
http://www.sura.o...hp?topic=67&mlevel=3 - 20k

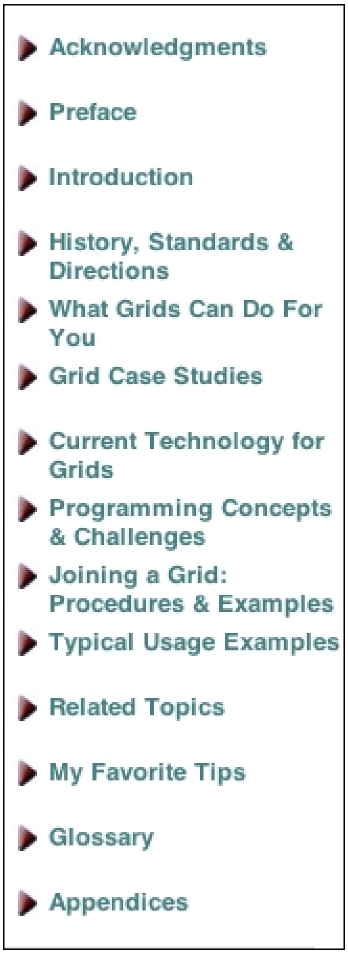
Grid Technology Cookbook, Joining a Grid: Procedures & Examples, The Open Science Grid, Gateways to other facilities and grids [4/10]
... common, integrated software stack Operational **security** and the **security** infrastructure Jobs, data, and storage Gateways to other facilities and ... grids Participating in the OSG Training on the OSG Bibliography Typical Usage Examples ...
http://www.sura.o...p?topic=212&mlevel=3 - 14k

Grid Technology Cookbook, Joining a Grid: Procedures & Examples, The Open Science Grid, Jobs, data, and

Notice that you have another search box at the bottom if you want to change or further your search.

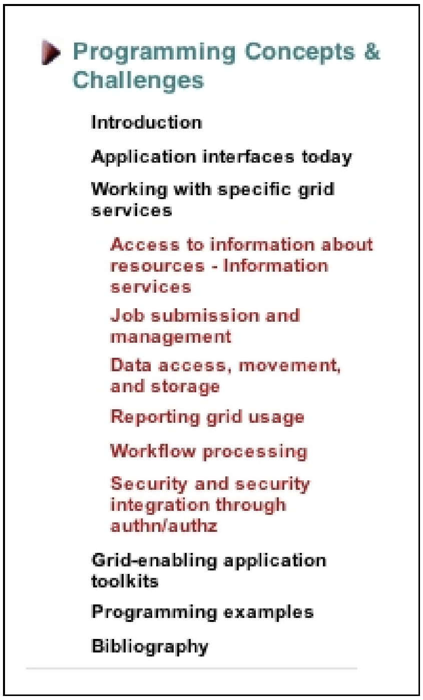
## Table of Contents

The left hand table of contents:



- Acknowledgments
- Preface
- Introduction
- History, Standards & Directions
- What Grids Can Do For You
- Grid Case Studies
- Current Technology for Grids
- Programming Concepts & Challenges
- Joining a Grid: Procedures & Examples
- Typical Usage Examples
- Related Topics
- My Favorite Tips
- Glossary
- Appendices

will expand up to two level of subtopics:


Programming Concepts & Challenges

Introduction

Application interfaces today

Working with specific grid services

Access to information about resources - Information services

Job submission and management

Data access, movement, and storage

Reporting grid usage

Workflow processing

Security and security integration through authn/authz

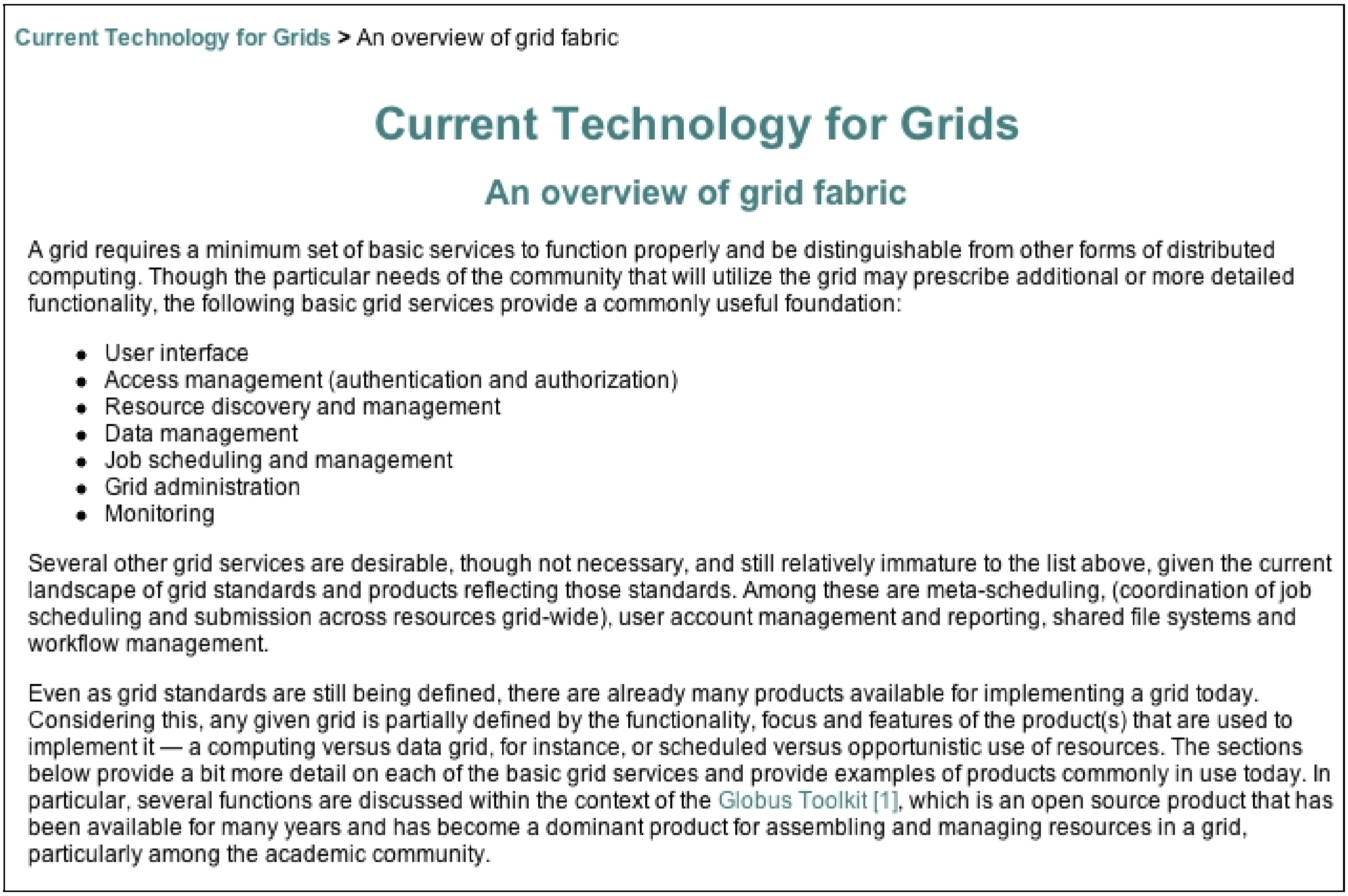
Grid-enabling application toolkits

Programming examples

Bibliography

### Menu Bar and Content

Lastly, the content area will include a menu bar and the actual section content. The top menu bar shows the navigation path taken to get to this spot. You can also traverse backwards by clicking on the bold topic items. For instance, in this example you can go back to see all topics under "Current Technology for Grids" by clicking on the bold text.


Current Technology for Grids > An overview of grid fabric

Current Technology for Grids

An overview of grid fabric

A grid requires a minimum set of basic services to function properly and be distinguishable from other forms of distributed computing. Though the particular needs of the community that will utilize the grid may prescribe additional or more detailed functionality, the following basic grid services provide a commonly useful foundation:

- User interface
- Access management (authentication and authorization)
- Resource discovery and management
- Data management
- Job scheduling and management
- Grid administration
- Monitoring

Several other grid services are desirable, though not necessary, and still relatively immature to the list above, given the current landscape of grid standards and products reflecting those standards. Among these are meta-scheduling, (coordination of job scheduling and submission across resources grid-wide), user account management and reporting, shared file systems and workflow management.

Even as grid standards are still being defined, there are already many products available for implementing a grid today. Considering this, any given grid is partially defined by the functionality, focus and features of the product(s) that are used to implement it — a computing versus data grid, for instance, or scheduled versus opportunistic use of resources. The sections below provide a bit more detail on each of the basic grid services and provide examples of products commonly in use today. In particular, several functions are discussed within the context of the Globus Toolkit [1], which is an open source product that has been available for many years and has become a dominant product for assembling and managing resources in a grid, particularly among the academic community.

We hope you find this easy to use. But please contact us if you have any questions, comments, or suggestions for the cookbook by using our feedback form at **Contact**.

# Introduction

## What is a grid?

Grid technologies represent a significant step forward in the effective use of network-connected resources, providing a framework for sharing distributed resources while respecting the distinct administrative priorities and autonomy of the resource owners. A grid can also help people discover and enable new ways of working together — providing a means for resource owners to trade unused cycles for access to significantly more compute power when needed for short periods, for example, or establishing a new organizational or cultural paradigm of focused investments in common infrastructure that is made available for broad benefit and impact.

Arriving at a common definition of "a grid" today can be very difficult. Perhaps the most generally useful definition is that **a grid consists of shared heterogeneous computing and data resources networked across administrative boundaries.** Given such a definition, a grid can be thought of as both an access method and a platform, with grid middleware being the critical software that enables grid operation and ease-of-use. For a grid to function effectively, it is assumed that

- hardware and software exists on each resource to support participation in a grid and,
- agreements and policies exist among grid participants to support and define resource sharing.

Standards to define common grid services and functionality are still under development. The promise of the transparent and ubiquitous resource sharing has excited and inspired a variety of views of a grid, often with considerable hype, from within multiple sectors (academe, industry, government) and flavored by numerous perspectives.

Many products are available for implementing "a grid", or grid-like capabilities. In some cases, the focus is on providing high performance capability, either through eased or increased access to existing high performance computing (HPC) resources, or a new level of performance realized through the orchestration of existing resources. In other cases, the focus is on using the network coupled with grid middleware to provide users or applications with seamless access to distributed resources of varying types, often in the service of solving a single problem or inquiry. With both standards and products under rapid development, product selection inevitably affects the definition of the resulting grid — that is, any given grid is at least partially defined by the functionality, focus and features of the product(s) that are used to implement it. Throughout this Cookbook, high level concepts and general examples will consider a variety of "grid types" but specific examples and case studies necessarily reflect particular products and approaches, with emphasis on those most commonly implemented today.

When grid technology is viewed as evolving into a generalized and globally shared infrastructure (a "grid of grids", comprised of campus grids, projects grids, regional grids, institutional or organizational grids, etc.), the vision is often referred to as "the Grid", still only a concept but similar in many ways to today's Internet, which evolved from distributed IP networks loosely united to provide a globally-used capability.

## Is it a grid or a cluster?

Clusters are often compared to, and confused with, grids. A cluster can be defined as a group of computers coupled together through a common operating system, security infrastructure and configuration that are used as a group to handle users' computing jobs. Clusters fall into a variety of categories, including the following.

- High performance computing (HPC) clusters provide a cost-effective capability that rivals or exceeds the performance of large shared-memory multiprocessors for many applications. Such clusters typically consist of thousands, tens of thousands, or hundreds of thousands of compute elements (i.e., processors or cores) and a high performance network (e.g., Myrinet, Infiniband, etc.) that is substantially more efficient than Ethernet.

- Beowulf clusters comprised of commodity-hardware compute nodes running Linux software and with dedicated interconnects (and similar architectures using other operating systems.)
- "Cycle-scavenging" services (aggregating and scheduling access to compute cycles that would otherwise go unused on individual systems, not necessarily running the same operating system (e.g., Condor pools).

For the purposes of this cookbook, a grid is assumed to consist of at least two such systems that connect across administrative domains.

A **computational grid** emphasizes aggregate compute power and performance through its collective nodes. A **data grid** emphasizes discovery, transfer, storage and management of data distributed across grid nodes.

# What instruments, resources and services might you find on a grid?

The predominant impression, or sometimes de facto definition, of a grid is that it is a collection of computational resources that can be combined to produce a greater HPC capability than each resource can provide on its own. In fact, many grids are focused on computation, at least initially, since the concepts and processes for combining computational elements are the most mature and compute-intensive applications are more obviously positioned to benefit from the multiplication of capability made possible by grid technology. A grid, however, can facilitate access to a wide variety of resources, and the type and timing of resources to be added to any given grid depends on the intended use community and application set. Resources other than compute resources may be more obvious or compelling for a particular community to share, such as visualization tools, high-capacity storage, data services, or access to unique or distributed instruments (e.g., telescopes, microscopes, sensors).

The actual process for adding a resource to a grid — or "grid-enabling" the resource — varies according to the type of resource being added as well as the grid technology in use. Compute resources are often the focus of examples within this Cookbook due to their prevalence and relatively straight-forward (or at least common!) inclusion in a grid. Processes to grid-enable other types of resources (e.g. data services, visualization, instruments) are less well known, are likely to be more variable from grid product to grid product, and may also be proprietary or highly dependent on the technical specifications of the particular device.

Some examples that illustrate the value and variety of making different resources available via a grid include:

- George E. Brown, Jr. Network for Earthquake Engineering Simulation [1] - From their Web site: "NEES is a shared national network of 15 experimental facilities, collaborative tools, a centralized data repository, and earthquake simulation software, all linked by the ultra-high-speed Internet2 connections of NEESgrid. Together, these resources provide the means for collaboration and discovery in the form of more advanced research based on experimentation and computational simulations of the ways buildings, bridges, utility systems, coastal regions, and geomaterials perform during seismic events ... NEES will revolutionize earthquake engineering research and education. NEES research will enable engineers to develop better and more cost-effective ways of mitigating earthquake damage through the innovative use of improved designs, materials, construction techniques, and monitoring tools." The NEES Central portal provides a single launching point for access to a variety of facilities (see NEEScentral web site [20]) including instruments such as geotechnical centrifuges, shake tables and tsunami wave basins.

- Laser Interferometer Gravitational-Wave Observatory (LIGO) [3] - From their Web site: "The Laser Interferometer Gravitational-Wave Observatory (LIGO) is a facility dedicated to the detection of cosmic gravitational waves and the harnessing of these waves for scientific research...the LIGO Data Grid is being developed with an initial focus on distributed data services — replication, movement, and management — versus high-powered computation. " The gravitational wave detectors produce large amounts of observational data that is analyzed alongside similar scale expected or predicated data by scientists working in this field.

- Earth System Grid [4] - From their Web site: "The primary goal of ESG is to address the formidable challenges associated with enabling analysis of and knowledge development from global Earth System models. Through a combination of Grid technologies and emerging community technology, distributed federations of supercomputers and large-scale data and analysis servers will provide a seamless and powerful environment that enables the next generation of climate research." Both data resources/services and high performance computational resources are necessary on this grid to meet a primary project objective: "High resolution, long-duration simulations performed with advanced DOE SciDAC/NCAR climate models will produce tens of petabytes of output. To be useful, this output must be made available to global change impacts researchers nationwide, both at national laboratories and at universities, other research laboratories, and other institutions."

- cancer Biomedical Informatics Grid (caBIG) [5] - From their Web site: "To expedite the cancer research communities, access to key bioinformatics tools, platforms and data, the NCI is working in partnership with the Cancer Center community to deploy an integrating biomedical informatics infrastructure: caBIG (cancer Biomedical Informatics Grid). caBIG is creating a common, extensible informatics platform that integrates diverse data types and supports interoperable analytic tools in areas including clinical trials management, tissue banks and pathology, integrative cancer research, architecture, and vocabularies and common data elements." The current suite of software development toolkits, applications, database technologies, and Web-based applications from caBIG are openly available from their Tools, Infrastructure, Datasets Web site [21], as tools for the target research community but also as models and reusable components for meeting similar service needs in other grid environments.

- Two notable initiatives are also addressing, at a more general level, the question of how to connect and control instruments in particular within a grid environment:

    - ♦ Grid-enabled Remote Instrumentation with Distributed Control and Computation [2] (GRIDCC) — From their Web site: "Recent developments in Grid technologies have concentrated on providing batch access to distributed computational and storage resources. GRIDCC will extend this to include access to and control of distributed instrumentation ... The goal of the GRIDCC project is to build a widely distributed system that is able to remotely control and monitor complex instrumentation.
    - ♦ Instrument Middleware Project [6] From their Web site: "The Common Instrument Middleware Architecture (CIMA) project, supported by the National Science Foundation Middleware Initiative, is aimed at "Grid enabling" instruments as real-time data sources to improve accessibility of instruments and to facilitate their integration into the Grid... The end product will be a consistent and reusable framework for including shared instrument resources in geographically distributed Grids."

  Both of the above initiatives are implementing their emerging products and services into actual and specific pilot applications to verify the efficacy and extensibility of their architecture and approach. Between the two initiatives, examples of grid-enabled instrumentation are being further developed in several diverse fields, including electrical and telecommunication grids (those "other grids"!), particle physics, earth observation and geohazard monitoring, meteorology, and x-ray crystallography.

## Who can access grid resources?

Authentication (authN) and authorization (authZ) are used together on grids to enforce conditions of use for resources as specified by the resource owner. This is recognized by Foster et al. in describing grid technology as a "resource-sharing technology with software and services that let people access computing power, databases, and other tools securely online across corporate, institutional, and geographic boundaries without sacrificing local autonomy" [11]. A researcher in the higher-education community, for example, may not only be a computer user on their campus's primary network, they may be a user of regional, national, or international resources within grid-based projects. Each grid determines what process and proof is acceptable to identify a user (authentication), and decides what that user is then authorized to access (authorization.)

Authentication (authN) is the act of identifying an individual user through the presentation of some credential. It does not include determining what resources the user can access, which is considered authorization. The process of authentication verifies that a real-world entity (e.g. person, compute node, remote instrument, application process) is who or what its identifier (e.g., username, certificate subject, etc.) claims it to be. In the process, the authentication credentials are evaluated and verified as being from a trusted source and at a particular level of assurance. Examples of credentials include a smartcard, response to a challenge question, password, public-key certificate, photo ID, fingerprint, or a biometric [12] [13] [14]. Authentication is also often referred to as identity management.

Authorization (authZ) refers to the process of determining the eligibility of a properly authenticated entity to perform the functions that it is requesting (access a grid-based application, service, or resource, for instance). The term "authorization" may be applied to the right or permission that is granted, the issuing of the token that proves a subject has that right, or to the token itself (e.g., a signed assertion). Signed assertions and other authorization characteristics are stored for reference in a variety of ways: within a local file system, on an external physical device (e.g. a smartcard), in a separate data system, or within system or enterprise-wide directories [12] [13] [14]. The characteristics that are assessed to determine status or levels of authorization for a given entity are often referred to as "attributes" of that entity.

Organizations contributing to a grid infrastructure develop policies for conditions of use of the grid resources and use authentication and authorization tools to implement those policies. Several types of authentication and authorization mechanisms have been developed or adopted for grids over time and are in active use today. There is not (yet?) consensus on which technologies are or will prove to be most effective, particularly for grids to scale to the level of global infrastructure, or for inter-departmental, inter-institutional, multi-project or multi-purpose grids, in which resources are not governed under the same administrative domain. However, a variety of sound, operational authN/Z approaches do exist. It is valuable to review several options when deciding on an approach to meet immediate as well as future needs of a given grid deployment, keeping in mind that choosing a particular toolkit may lock you into a particular authentication/authorization model.

## Bibliography

[1] George E. Brown, Jr. Network for Earthquake Engineering Simulation (http://www.nees.org)
[2] Grid Enabled Remote Instrumentation with Distributed Control and Computation (GRIDCC) (http://www.gridcc.org/)
[3] Laser Interferometer Gravitational-Wave Observatory (LIGO) (http://www.ligo.caltech.edu)
[4] Earth System Grid (http://www.earthsystemgrid.org/)
[5] cancer Biomedical Informatics Grid (caBIG) (http://cabig.cancer.gov/index.asp)

[6]  Instrument Middleware Project (http://www.instrumentmiddleware.org/metadot/index.pl)
[7]  Grid Café (http://gridcafe.web.cern.ch/gridcafe/gridatwork/gridatwork.html)
[11] Foster, The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, 2002
[12]  nmi-edit Glossary (http://www.nmi-edit.org/glossary/index.cfm)
[13]  GFD Authorization Glossary (http://www.gridforum.org/documents/GFD.42.pdf)
[14]  Internet2 Authentication WebISO (http://middleware.internet2.edu/core/authentication.html)
[17]  SURA's NMI Case Study Series (http://www.sura.org/programs/nmi_testbed.html#NMI)
[18]  Adiga, Henderson, Jokl, et al. "Building a Campus Grid: Concepts and Technologies" (September 2005) (http://www1.sura.org/3000/SURA-AuthNauthZ.pdf)
[19]  Adiga, Barzee, Bolet, et al. "Authentication & Authorization in SURAgrid: Concepts and Technologies", (May 2005) (http://www1.sura.org/3000/BldgCampusGrids.pdf)
[20]  NEEScentral website (https://central.nees.org/?action=DisplayFacilities)
[21]  caBIG Tools, Infrastructure, Datasets (https://cabig.nci.nih.gov/inventory/)

# History, Standards & Directions

## Introduction

Most software developers are aware of the role and importance of software standards, especially when attempting to create a distributed middleware infrastructure, or applications and services that can be reused or inter-operate with other systems or infrastructure. Standards percolate throughout all aspects of software development, from the formats of datatypes, on-wire protocols through to design patterns and the architecture of component frameworks. Without software standards, although development can be quicker, developers can easily create "islands" of software that work as isolated solutions but will need to be revised, sometimes significantly, if the runtime environment changes.

This chapter aims to give the reader an understanding and status of important current and near-future standards in the Grid arena. A short history of distributed computing, metacomputing, and the Grid is provided to To frame the discussion, This history will help put the development of grid standards in perspective and is followed by a review of several relevant current standards bodies, along with a summary of the standards associated with each. Additional detail is provided for keycurrent and emerging standards that will have the most impact on the future of the Grid, followed by some final conclusions.

## History

### Early Distributed Computing

The history of distributed computing can arguably by traced to 1960, when J.C.R. Licklider suggested "a network of such [computers], connected to one another by wide band communication lines" which provided "the functions of present-day libraries together with anticipated advances in information storage and retrieval and [other] symbiotic functions." [1]. A large amount of work on networking continued after that, leading to the initial development of the ARPANET, starting in 1969. The goal of these networks was sometimes simply moving data from one machine to another, but at other times, it consisted of the the more ambitious goal of enabling active processes on multiple machines to communicate with one another. For example, the 1976 RFC 707 [2] discussed network-based resource sharing, and proposed the remote procedure call as a mechanism to permit effective resource sharing across networks.

By the mid 1980s, distributed computing became an active, major field of research, particularly as local, national and international networks became more ubiquitous. In 1984, John Gage of Sun used the phrase "The Network is the Computer" to describe the idea that the connections between computer are really what enables systems to be effectively used. In 1985, the Remote-UNIX project [3] at the University of Wisconsin created software to capture and exploit idle cycles in computers (also known as "cycle scavenging") and provided these to the scientific community, who were looking for additional options to solve computationally-intense problems. This led to the development of Condor project [4] , which is widely used today as distributed middleware. In 1989, the first version of Parallel Virtual Machine (PVM [5]) was written at Oak Ridge National Laboratory. PVM enabled multiple, distributed computers to be used to run a single job. PVM initially was used to link together workstations that were located in the same general area.

### Metacomputing

In 1987, the Corporation for National Research Initiatives (CNRI) suggested a research program in Gigabit Testbeds [6] to the NSF. This led to five five-year projects which started in 1990. Some of these projects were focused on networking, and others on applications, including linking supercomputers together. The term metacomputing was coined to refer to this idea of multiple computers working together while physically separated. Larry Smarr, then at NCSA, is generally credited with popularizing this term.

In 1995, the I-Way project [7] began. This project worked to integrate previous tools and technologies, such as those aimed at locating and accessing distributed resources for computation and for storage, and a number of network technologies. I-Way was generally viewed as being successful, as it deployed a distributed platform containing components at seventeen sites for use by 60 research groups. One key part of the project was the recognition that having a common software stack (I-Soft) installed on a front-end machine (point-of-presence, or I-POP) at each site was an effective way of hiding some of the complexity about the individual resources and their locations.

## Grid Computing

### *Globus*

In 1996, researchers at Argonne National Laboratory and the University of Southern California started The Globus Project [8, 9, 10, 11]. The aim of the project was to build on earlier work undertaken in the I-Way project and focus on helping scientists develop distributed and collaborative applications that make use of the Internet's infrastructure for large-scale problems.

At the heart of the project is the Globus Toolkit, which is developed by the Globus Alliance. It provides a number of services, including those for resource monitoring and discovery, job submission, security, and data management.

The toolkit has evolved many times since its inception in 1996. Globus versions 1 and 2 had procedural interfaces and were more aimed towards distributed high-performance applications. The Open Grid Services Architecture (OGSA) that was first announced by the Global Grid Forum in February 2002, and later declared to be its flagship architecture for the Grid has a significant affect on the Globus Toolkit. OGSA defines a service-oriented grid architecture. The Globus Alliance produced an incarnation of this architecture in the form of Globus Toolkit version 3 (GT3), called the Open Grid Service Infrastructure (OGSI), which was first released in July 2003. Critics identified several problems with OGSI, and consequently in January 2004 Hewlett-Packard, IBM, Fujitsu, and the Globus Alliance announced the WS-Resource Framework (WS-RF). The Globus Toolkit was refactored again, and in April 2005 version 4 (GT4) of the software was released. In GT4 most of the services provided are implemented on top of WS-RF, although some are not. (The Globus Toolkit 3 Programmer's Tutorial provides some additional perspective on this from the Globus team [82], including some detail on which services fall into which categories [83].)

### *Legion*

Legion [12, 13], which emerged in late 1993, was an object-based meta-system developed at the University of Virginia. Legion aimed to provide a software infrastructure so that a system of heterogeneous, geographically distributed, high-performance machines could interact seamlessly. Legion attempted to provide a user, at their workstation, with a single, coherent, virtual machine. The Legion system itself was organized by classes and metaclasses and was originally based on Mentat [14]. In early 1996, Legion received its first national funding, and the initial prototype was rewritten by November 1997. The system was originally deployed between the University of Virginia, SDSC, NCSA and UC Berkeley. The system was first demonstrated at Supercomputing in 1997. Legion was subsequently deployed more widely, including sites in Japan and Europe in what was called NPACI-Net. As Legion was rolled out, various distributed applications were ported, including those from areas such as materials science, ocean modelling, sequence comparison, molecular modelling and astronomy.

In 1999, a company called Applied MetaComputing was founded and by 2001 it had raised sufficient venture capital to commercialize Legion. The company was renamed the AVAKI Corporation, and Legion became Avaki, which was first released as a commercial offering in September 2001. In 2005, Avaki was purchased by Sybase [15].

***UNICORE***

The Uniform Interface to Computing Resources (UNICORE) project [16] started in August 1997. The project aimed to seamlessly and securely join a number of German supercomputing centres together without changing their existing systems or procedures. The UNICORE consortium consisted of developers, supercomputing centres, users and vendors. The initial UNICORE system had a graphical interface based on Java Applets that was deployed via a Web browser. It also included a central job scheduler that used Codine from Genias (now Grid Engine [17], sponsored by Sun), and a security architecture based on X.509 certificates.

The UNICORE Plus project [18] started in January 2000, with two years of funding. The goal of this project was to continue the development of UNICORE with the aim of producing a grid infrastructure together with a Web portal. It also aimed to harden the software for production, integrate new services, and deploy the system to more participating sites. The Grid Interoperability Project (GRIP) [19] was an overlapping two-year project that started in 2001, and was funded by the European Union, that aimed to realize the interoperability of UNICORE with the Globus Toolkit, as well as working towards Grid interoperability standards.

Finally, the UniGrids project [20] that started in July 2004 is developing Grid services based on OGSA. The goal is to transform UNICORE into a system with interfaces that are compliant with WS-RF and that can interoperate with other compliant systems.

# Standards bodies

## The Global Grid Forum (GGF)

http://www.ggf.org/, 2000 — 2006

The Global Grid Forum grew out of a series of conversations, workshops, and Birds of a Feather (BoF) sessions that addressed issues related to grid computing. The first of these BoFs was held at SC98, the annual conference of the high-performance computing community. That meeting led to the creation of the Grid Forum, a group of grid developers and users in the U.S who were dedicated to defining and promoting grid standards and best practices. By the end of 2000, the Grid Forum had merged with the European Grid Forum (eGrid) and the Asia-Pacific Grid Forum to form the Global Grid Forum. The first Global Grid Forum meeting was held in March 2001. After that, the GGF produced numerous standards and specifications documents and held world-wide meetings. The GGF merged with the Enterprise Grid Alliance (EGA) to form the Open Grid Forum (OGF) in June 2006. GGF standards and products have been subsumed into OGF standards.

## The Enterprise Grid Alliance (EGA)

http://gridalliance.org/, 2004 — 2006

The EGA was formed in 2004 to focus exclusively on accelerating grid adoption in enterprise data centres. The EGA addressed obstacles that organizations face in using enterprise grids through open, interoperable solutions and best practices. The alliance published the EGA Reference Model and Use Cases [21], and documents that described Security Requirements [22] as well as Data and Storage Provisioning [23]. The EGA significantly raised awareness worldwide of enterprise grid requirements through effective marketing programs and regional operations in Europe and Asia. The EGA merged with the GGF to form the OGF. EGA members were primarily vendors and integrators.

## The Open Grid Forum (OGF)

http://www.ogf.org/, 2006 —

The OGF was formed by the merger of the GGF and EGA in June 2006. OGF members include vendors, integrators, academic and government laboratories and programs, and users. It has working groups in a number of areas, including applications, architecture, compute, data, management, and security.

- Applications work includes an API for submission and control of jobs (drmaa-wg), an API and related services for checkpointing (gridcpr-wg), an API for grid remote procedure calls (gridrpc-wg), and an API for grid applications (saga-core-wg).
- In architecture, there is general work on the OGSA specification (ogsa-wg) as well as work to create a name space for OGSA and to produce a WS-Naming naming specification (ogsa-naming-wg).
- Compute work includes discussion of resource management protocols (graap-wg) and grid scheduling (gsa-rg), defining a language for job submission (jsdl-wg), and work in OGSAm, which includes a specification for a minimal subset of services (ogsa-bes-wg), a core use case for high-performance computing (ogsa-hpcp-wg), and protocols for scheduling (ogsa-rss-wg).
- In data, work includes a language to describe data files and streams (dfdl-wg), standards for grid data services (dais-wg), interfaces and an architecture for grid file systems (gfs-wg), storage management functionality (gsm-wg), the gridFTP protocal(gridftp-wg), an interface for file-like functionality across grids (byteio-wg), interfaces for moving data across varying protocols (ogsa-dmi-wg), and an overall data architecture under OGSA (ogsa-d-wg).
- Management work includes defining application contents (acs-wg); describing service configuration, deployment, and lifecycle management (cddlm-wg); defining an accounting service (rus-wg); and defining a record for use in accounting (ur-wg).
- Finally, in security, there is work on defining specifications for interoperability of authorization components (ogsa-authz-wg).

## The Organization for the Advancement of Structured Information Standards (OASIS)

http://www.oasis-open.org/, 1993 —

The Organization for the Advancement of Structured Information Standards (OASIS) consortium is non-profit making voluntary international organization that promotes industry standards for e-business. OASIS was founded in 1993 as SGML Open and changed its name in 1998 to reflect its expanded technical scope. The consortium produces various Web services standards along with standards for security, e-business. OASIS has more than 5,000 participants representing over 600 organizations and individual members in 100 countries. The standards include those related to the Extensible Markup Language (XML) and the Universal Description, Discovery, and Integration (UDDI) service. The Web services standards produced by OASIS focus primarily on higher-level functionality such as security, authentication, registries, business process execution, and reliable messaging.

## The Liberty Alliance

http://www.projectliberty.org/, 2001 —

The Liberty Alliance project is an international coalition of companies, nonprofit groups, and government organizations formed in 2001 to develop an open standard for federated identity management, which addresses technical, business, and policy challenges surrounding identity and Web services. The project has the vision of enabling a networked world that is based on open standards where consumers, can easily conduct online transactions in a private and secure way. The Liberty Alliance has developed the Identity Federation

Framework, which enables identity federation and management and provides interface specifications for personal identity profiles, calendar services, wallet services, and other specific identity services.

## The World Wide Web Consortium (W3C)

http://www.w3.org/, 1994 —

The World Wide Web Consortium (W3C) is an international organization conceived by Tim Berners-Lee in 1994 with the aims of promoting common and interoperable protocols. The W3C created the first Web services specifications in 2003, which have evolved through several versions and also become the underlying building blocks for many grid services. The initial focus was on low-level, core functionality such as SOAP and the Web Services Description Language (WSDLbut the W3C has since spearheaded many other Web related standards. The W3C has now developed more than 80 technical specifications for the Web, ranging from XML and HTML to Semantic Web technologies such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL). W3C members are organizations that typically invest significant resources in Web technologies. OASIS is a member, and the W3C has partnered with the OGF in the Web services standards area.

## The Distributed Management Task Force (DTMF)

http://www.dmtf.org/, 1992 —

The Distributed Management Task Force (DMTF) is an industry-based organization founded in 1992 to develop management standards and integration technologies for enterprise and Internet environments. The DMTF focuses on developing and unifying management standards with the aim of enabling a more integrated and cost effective approach to management through interoperable management solutions. The DMTF has created the Common Information Model (CIM), and also developed communication/control protocols such Web-Based Enterprise Management (WBEM), the Systems Management Architecture for Server Hardware (SMASH) initiative, and core management services/utilities. The DMTF formed an alliance with the GGF in 2003 for the purpose of building a unified approach to the provisioning, sharing, and management of Grid resources and technologies.

## The Internet Engineering Task Force (IETF)

http://www.ietf.org/, 1986 —

The Internet Engineering Task Force (IETF) is an open international community of network designers, operators, vendors, and researchers concerned with the evolution and smooth operation of the Internet. The Globus Alliance has worked with the IETF to produce two RFCs: RFC4462 — Generic Security Service Application Program Interface (GSS-API) Authentication and Key Exchange for the Secure Shell (SSH) Protocol [24], and RFC3820 — Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile [25]. These are discussed further under the Grid Security Infrastructure (GSI).

## The Web Services Interoperability Organization, (WS-I)

http://www.ws-i.org/, 2002 —

The Web Services Interoperability Organization (WS-I) is an open industry body formed in 2002 to promote the adoption of Web services and interoperability among its different implementations. Its role is to integrate existing standards rather than create new specifications. WS-I creates, promotes and supports generic protocols for the interoperable exchange of messages between Web services. In order to do this WS-I publishes profiles that describe in detail which specifications a Web service should adhere to and offer guidance in their proper use. The overall goal is to provide a set of rules for integrating different service implementations with a minimum number of features that impede compatibility.

# Current standards

## Web Services Specifications and Standards

Web Services [26, 84] are loosely coupled platform-independent XML-based applications that operate and communicate within distributed systems. The core components of the Web Services architecture are SOAP for communications, Web Services Description Language (WSDL) for describing network services as a set of endpoints operating on messages containing either document- or procedure-oriented information, and Universal Description Discovery & Integration (UDDI) protocol that defines a set of services supporting the description and discovery of businesses, organizations, service providers, the services available, and the technical interfaces used to access these services.

### *SOAP*

SOAP Version 1.2 [27] is an XML-based protocol intended for exchanging structured information in a distributed environment. SOAP uses XML technologies to define an extensible messaging framework that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation specific semantics.

The SOAP Version 1.2 specification consists of three parts:

- Part 0 is a document intended to be a tutorial on the features of the SOAP Version 1.2,
- Part 1 is a specification document that defines the SOAP messaging framework,
- Part 2 describes a set of extensions that may be used with the SOAP messaging framework.

### *Web Services Description Language (WSDL)*

WSDL 1.1 [28] is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. These related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow the description of endpoints and their messages regardless of what message formats or network protocols are used to communicate. However, the only bindings described are in conjunction with SOAP 1.1, HTTP GET/POST, and MIME.

### *Universal Description Discovery and Integration (UDDI)*

The Universal Description Discovery & Integration (UDDI) [29] standard defines a set of services that support the description and discovery of:

- Businesses, organizations, and other Web services providers,
- The Web services they make available,
- The technical interfaces which may be used to access those services.

UDDI is based on a set of standards that include HTTP, XML, XML Schema, and SOAP, that provides an infrastructure for a Web services-based software to be published and searched for either publicly or effectively privately internally within an organization.

### *WS-RF*

WS-RF [30] is a set of Web services specifications being developed by the OASIS organization. Taken together and with the WS-Notification (WSN) specification, these specifications describe how to implement OGSA capabilities using Web services. The purpose of the Web Services Resource Framework (WS-RF) is to define a generic framework for modelling and accessing persistent resources using Web services so that the definition and implementation of a service and the integration and management of multiple services is made easier. WS-RF has a standard approach to extend Web Services. It is based on different standard/recommended WS-* specifications:

- WS-ResourceProperties (WS-RP) [31] are the properties of a WS-Resource, which are modeled as XML elements in the resource properties document. A WS-Resource has zero or more properties expressible in XML, representing a view on the WS-Resource's state.
- WS-ResourceLifetime (WS-RL) [32] standardizes the means by which a WS-Resource can be destroyed, monitored and manipulated.
- WS-ServiceGroup (WS-SG) [33] defines a means of representing and managing heterogeneous, by-reference, collections of Web services. This specification can be used to organize collections of WS-Resources, for example aggregate and build services that can perform collective operations on a set of WS-Resources.
- WS-BaseFaults (WSRF-BF) [34] defines an XML Schema for base faults, along with rules for how this base fault type is used and extended by Web services.
- WS-Addressing [35] provides a mechanism to place the target, source and other important address information directly within a Web services message. In short, WS-Addressing decouples address information from any specific transport protocol. WS-Addressing provides a mechanism called an endpoint reference for addressing entities managed by a service.
- WS-Notification [36, 37] is a family of documents including three specifications: WS-BaseNotification defines the Web services interfaces for NotificationProducers and NotificationConsumers; WS-BrokeredNotification defines the Web services interface for the NotificationBroker, which is an intermediary that among other things, allows the publication of messages from entities that are not themselves service providers.
- WS-Topics [38] defines a mechanism to organize and categorize items of interest for subscription known as "topics."

WS-RF is itself extendable through other WS-* specifications, such as WS-Policy, WS-Security, WS-Transaction, WS-Coordination.

At the 18th Global Grid Forum meeting (September 2006), discussions were held on the infrastructure to host grid applications that evolved WS-RF to Web Services Resource Transfer (WS-RT). This evolution is intended to better handle state information that is required for persistent services.

### *WS-RT*

The Web Services Resource Transfer (WS-RT) specification [39] was developed jointly by IBM, Hewlett-Packard, Intel, and Microsoft to provide a unified resource access protocol for Web Services.

WS-RT extends WS-Transfer operations, by adding the capability to operate on fragments of management resource representations. The WS-Transfer specification, which defines standard messages for controlling resources using the familiar paradigms of "get", "put", "create", and "delete". The extensions primarily deal with fragment-based access to resources to satisfy the common requirements of WS-RF and WS-Management. The WS-RT specification will form a core component of a unified resource access protocol for the Web services. The specification intends to meet the following:

- Define a standardized technique for accessing resources using semantics familiar to those in the system management domain, using get, put, create and delete.
- Define WSDL 1.1 portTypes, that are compliant with WS-I Basic Profile 1.1.
- Describe the minimal requirements for compliance without constraining richer implementations.
- How to compose with other Web service specifications for secure, reliable, transacted message delivery.
- Provide extensibility for more sophisticated and/or currently unanticipated scenarios.
- Support a variety of encoding formats including SOAP 1.1 and SOAP 1.2 envelopes, and others.

## Grid Specifications and Standards

### *Architecture*

The OGF describes OGSA [40, 41, 42, 43] as representing an evolution towards a Grid system architecture based on Web services concepts and technologies.

Building on both Grid and Web services technologies, OGSI defines mechanisms for creating, managing, and exchanging information among entities called Grid services. Succinctly, a Grid service is a Web service that conforms to a set of conventions (interfaces and behaviors) that define how a client interacts with a Grid service. These conventions, and other OGSI mechanisms associated with Grid service creation and discovery, *provide for the controlled, fault-resilient, and secure management of the distributed and often long-lived state that is commonly required in advanced distributed applications,* [44, 45], and focus on technical details, providing a full specification of the behaviors and WSDL interfaces that define a Grid service. However, some aspects of OGSI (e.g., specification very dense, stateful versus stateless services) create problems for the convergence of Web services and grid services, and thus have led the community to try again with WS-RF.

The OGSA WS-RF Basic Profile 1.0 [46] is an *OGSA Recommended Profile as Proposed Recommendation* as defined in the OGSA Profile Definition [47] . The OGSA WS-RF Basic Profile 1.0 describes uses of widely accepted specifications that have been found to enable interoperability. The specifications considered in this profile are specifically those associated with the addressing, modeling, and management of state: WS-Addressing [35], WS-ResourceProperties [31] , WS-ResourceLifetime [32] , WS-BaseNotification [36] , and WS-BaseFaults [34] .

### *Scheduling*

The interaction between the large variety of complex Grid services expected to exist will require resource management and scheduling solutions that allow the coordinated use of the services, something that is currently not readily available. Access to resources is typically subject to individual access, accounting, priority, and security policies that are imposed by the resource owners. In addition the consideration of different policies is also important for the implementation of various services, for example accounting or billing services. Generally those policies are enforced by local management systems. Therefore, an architecture that supports the interaction of independent local management systems with higher-level scheduling services is an important component for the Grid. Further, a user of a Grid may also establish individual scheduling objectives. Future Grid scheduling and resource management systems must consider those constraints in the scheduling process.

A scheduling architecture must support the cooperation between different scheduling instances managing arbitrary Grid resources, including network, software, data, storage, and processing units. Co-allocation and the reservation of resources will be key aspects of the new scheduling architecture, which will also integrate user- or provider-defined scheduling policies. The GSA-RG intends to determine the components needed for a generic and modular scheduling architecture and its interactions. The group has started by creating a dictionary of terms and keywords [48], and identifying a set of relevant use cases based on experiences obtained by existing Grid projects [49].

### *Resource Management*

The Grid, as any computing environment, requires some degree of system management, such as the management of jobs, security, storage and networks. The management of the Grid is a potentially complex task given that resources are often heterogeneous, distributed, and cross multiple management domains.

The OGSA Resource Management document [50] contains a discussion of the issues of management that are specific to a Grid and especially to OGSA. It first defines the terms and describes the management requirement as they relate to a Grid, and then discusses the individual interfaces, services, and activities that are involved in Grid management, including both management within the Grid and the management of its infrastructure. It concludes with a gap analysis of the state of manageability in OGSA, primarily identifying Grid-specific management functionality that is not provided for by emerging distributed management standards. The gap analysis is intended to serve as a foundation for future work.

### *System Configuration*

Successful realization of the Grid vision of a broadly applicable and adopted framework for distributed systems integration, virtualization, and management requires the support for configuring Grid services, their deployment, and managing their lifecycle [51]. A major part of this framework is a language used to describe the necessary components and systems. The Configuration Description, Deployment, and Lifecycle Management document [52] provides a definition of the CDDLM language that is based on the SmartFrog (Smart Framework for Object Groups) and its requirements. The CDDLM component model document [53] provides a definition of the model and process whereby a Grid resource is configured, instantiated, and destroyed. The CDDLM API document [54] provides the WS-RF-based SOAP API for deploying applications to one or more target computers. The code that calls the API can upload files to the service implementing the API, then submit a deployment descriptor for deployment of the application contained in the file.

### *Data*

Three recommendations regarding data access and integration services made by the DIAS-WG (Database Access and Integration Services Working Group) are currently being considered by the OGF: WS-DAI (core), WS-DAIR (relational data), and WS-DAIX (XML data).

- WS-DAI [55] is a specification for a collection of generic data interfaces that can be extended to support specific kinds of data resources, such as relational databases, XML repositories, object databases, or files. Related specifications (currently, WS-DAIS and WS-DAIX) define how specific data resources and systems can be described and manipulated through such extensions. The specifications can be applied in regular web services environments or as part of a grid fabric.
- WS-DAIR [56] is a specification for a collection of data access interfaces for relational data resources, which extends interfaces defined in the "Web Services Data Access and Integration" document (WS-DAI). The specification can be applied in regular web services environments or as part of a grid fabric.
- WS-DAIX [57] is a specification for a collection of data access interfaces for XML data resources, which extends interfaces defined in the Web Services Data Access and Integration document (WS-DAI). The specification can be applied in regular web services environments or as part of a grid fabric.

### *Data Movement*

The GridFTP protocol has become a popular data movement tool used to build distributed grid-oriented applications. The GridFTP protocol extends the FTP protocol by adding certain features designed to improve the performance of data movement over a wide area network, to allow the application to take advantage of "long fat" communication channels, and to help build distributed data handling applications.

Several groups have developed independent implementations of the GridFTP v1 protocol [58] for different types of applications. The experience gained by these groups uncovered several drawbacks of the GridFTP v1 protocol. Mandrichenko et al [59] propose modifications of the protocol to address the majority of the issues found.

### *Security*

The OGSA Security Roadmap [60] defines an authorization service that allows services to make queries and receive responses in regards to access control on grid services. OGSI authorization services are Grid Services providing authorization functionality over an exposed Grid Service portType. A client sends a request for an authorization decision to the authorization service and in return receives an authorization assertion or a decision. A client may be the resource itself, an agent of the resource, or an initiator or a proxy for an initiator who passes the assertion on to the resource.

Welch et al [61] define a number of use cases for authorization in OGSI covering the possible set of actions that may be attempted against a Grid Service, as well as how the different existing authorization services listed previously may be used. From these use cases it derives a set of requirements for authorization in OGSI.

### *Grid Security Infrastructure*

The goal of the Grid Security Infrastructure (GSI) [62, 63] is to allow secure authentication and communication over an open network. The GSI is based on public key encryption and X.509 certificates, and adheres to the Generic Security Service API (GSS-API) [24], which is a standard API for security systems promoted by the Internet Engineering Task Force (IETF). Extensions to these standards have been added for single sign-on and delegation [25]. GSI provides:

- A public-key system;
- Mutual authentication through digital certificates;
- Credential delegation and single sign-on through proxy certificates.

## Emerging standards and specifications

In this section we briefly detail and discuss what we believe to be the most important of the emerging or more established grid-based standards. It should be borne in mind that the standards that have been included in this section are closely tied to the dominant grid-based applications being routinely used today.

An unscientific review of grid applications that have been described in recent research papers and publicized on the Web reveals that current grid usage is dominated by "high throughput" applications, which are mostly "parameter sweep" or "workflow" applications. The former is many instances of the same application, each with different input data, where the resulting output data is then analyzed. The latter is a possibly-sophisticated pipeline of processes, "plugged" together to form a chain that can undertake a series of computational tasks on the original input data set, where a transformed data set is produced. Typically, in both types of applications, some pre-processing is undertaken to create the parameter sweep or workflow, and then the application is sent off to a software component that schedules and runs the individual tasks on the back-end grid resources. These applications rely on the ability to both schedule and reserve back-end resources. A third type of application that is increasingly becoming common is the integration of distributed and heterogeneous databases. Obviously, each database instance is potentially quite different; each could hold census, medical, geographical, historical, or other records. Queries across these databases can potentially reveal interesting patterns that provide unique insights. This type of application, where a user can send off distributed queries to back-end databases, is becoming ever more popular. This application type relies on the standardization of data access and integration technologies. While, there are many other grid applications, we believe that these three broad types will be dominant for the immediate future, and therefore they will determine the most important emerging standards and specifications.

## OGSA

Using the OGSA model, which proposes a Service-Oriented Architecture (SOA), currently seems to be the best way for the Grid to become more accepted. A SOA provides an opportunity for almost any provider to supply user services. Moreover it should enable a grid user to bind together a range of diverse services in a workflow that can undertake the tasks needed by their application. The high-level architectural view, inherent in OGSA is conceptually important, however, the actual implementation details of OGSA are crucial, because any SOA cannot be globally successful without well-defined standards.

## From WS-RF To WS-RT

Many in the grid community believe that stateful services are an important architectural facet, but this has been perhaps the most contentious and debated area over the last few years. With the adoption of OGSA, two instantiations of this architecture have appeared: first, the Open Grid Services Infrastructure (OGSI), and more recently, the Web Services Resource Framework (WS-RF). The former was dropped for numerous reasons, but mainly because it diverged from normal Web Services tooling, and because it contained too much in one standard. WS-RF materialized soon after, and seemed to be a better solution, but it appears that this too has now been dropped in favor of Web Services Resource Transfer (WS-RT). It is unclear, at this moment, why this has occurred, but it is possible that this move may be more politically motivated than technically motivated. Existing grid middleware, such as Globus will be once again refactored to use WS-RT, but the effect of yet another change for the community is unclear. One effect of similar changes in the past has been for the community to either continue to use procedural middleware, such as Globus 2.4, or to resort to using basic Web Services, and standards SOAP and WSDL.

## Registries

In a SOA, a registry is a vital component if clients and services are going to find each other and bind together. Globus originally used LDAP, but has now moved to an in-memory XML-based registry that supports XPath and XQuery. gLite, the EGEE middleware, has R-GMA as its registry. This is based on relational database concepts, is non-standard, and has its own data schema. The Grid community is also using UDDI-based registries. The UDDI standard has changed a lot over the last few years, and OASIS is currently working on version 4 of the UDDI standard, while common UDDI implementations are based on version 2 of the standard, which does not meet the needs of the Grid community for a variety of reasons. The only currently successful use of UDDI for grid purposes is via efforts such as Grimoires [64], which has extended UDDI to suit the needs of the Grid community. Other registry standards that may be applicable are starting to emerge. One example of this is ebXML [65], a registry that is capable of storing any type of electronic content such as XML or text documents, images, sound and video. The ebXMLsoft Registry and Repository [66] supports a number of clients, including web browsers, SOAP, Java, and REST [67] (Representational State Transfer).

## JSDL

There are now many languages for submitting jobs to the Grid; hence interoperability has been difficult if not impossible. A common language for this purpose is therefore essential. The Job Submission Description Language (JSDL) [68] is a declarative language for describing the requirements of job submission. A JSDL document describes the job requirements, identification information, the application, e.g., executable, arguments, the required resources, e.g., CPUs, memory, and the input/output files. JSDL does not define a submission interface, what the results of a submission will look like, or how resources are selected. JSDL 1.0 was published by the OGF as GFD-R-P.56 in November 2005 [69] and includes a description of JSDL elements and XML Schema. JDSL works with a number of scheduling systems, including Condor, LSF, Sun's Grid Engine and UNICORE, as well as with UNIX fork.

## DRMAA

A key component of the grid is a distributed resource management system: software that queues, dispatches, and controls jobs. The Distributed Resource Management Application API (DRMAA) working group [70] has released the DRMAA specification, which offers a standardized API for application integration with C, Java, and Perl bindings. DRMAA can be used to interact with batch/job managements systems, local schedulers, queuing systems, and workload management systems. DRMAA has been implemented in a number of DRM systems, including Sun's Grid Engine, Condor, PBS/Torque, Gridway, gLite, and UNICORE.

## SAGA

The Simple API for Grid Applications (SAGA) [71], has the potential to become an important specification, due to the current problems for application developers, which revolves around the rapid rate of change in middleware *de facto* standards and APIs, its complexity, and the fact that different middleware exists on different grid systems. SAGA aims to be to the grid application developer what MPI has been to the developer of parallel application. If SAGA is successful, there will be a surge in development of new grid applications, the rewriting of some current grid applications to have significantly less code, and the emergence of libraries written on top of SAGA. SAGA started when a number of projects contemplating similar issues came together in 2004, including GAT, ReG Steering, and CoG. In October 2006, a draft SAGA-API was released, specified in SIDL (Scientific Interface Definition Language), which is object-oriented and language-neutral. If the promise of SAGA can be delivered, a stable period for application development will follow, similar to that delivered by MPI in the parallel computing arena over the last 15 to 20 years.

## GridFTP

An important feature of a distributed environment is the movement of various types of data between remote components. Data movement can include data staging, copying an executable to a remote platform, inter-application communications, or copying output data back to the user of an application. As noted earlier in the section on data movement in Grid Specifications and Standards, the GridFTP protocol has become popular for moving data in distributed grid-oriented applications. GridFTP extends FTP, as defined by RFC959 and other IETF documents, by adding features such as multi-streamed transfer, auto-tuning and grid-based security.

## Workflow

Workflow-based technologies can be found almost everywhere; they can be found embedded in a range of development tools, network applications and Web services. There are many grid-based system too, ranging from those that support SOAs, such as Kepler [72] and Taverna [73], to those that support applications specific middleware such as Globus, Condor, GridAnt [74], and Pegasus [75]. Even though workflow standards seem to be everywhere, they have not bridged the gap to broad adoption.

## Data Access and Integration

There is a need for middleware to assist with the access to and integration of data from separate sources that are distributed over the Grid. The OGF Database Access and Integration Services Work Group (DAIS-WG) [76] is working toward standards in this area. Two important standards are emerging, OGSA-DAI [77], middleware that allows data resources such as relational or XML databases to be accessed via Web Services, and the Distributed Query Processing (DQP) system, known as OGSA-DSP [78], that allows efficient queries across these distributed data resources.

## Summary and conclusions

Standards of all types are crucial if the vision of the Grid is to be fully realized. There are a large number of both standards bodies and standards that impact and define today's Grid. Some standards are built on one another, and some standards oppose each other. (The recent roadmap on WS [79] from HP, IBM, Intel, and Microsoft may be a sign that there will be fewer competing specifications in the future.) There are a number of generalizations that can be made about standards processes, and almost all of them, both positive and negative, apply to the standards on which the Grid is based. Some of the problems in the current Grid standards are:

- Vested interest and potential intransigence on the part of some major players who are defining standards,
- Lack of involvement from other key players,
- Changing road maps of related standards,
- General politics.

The effect of all these problems is the delay of the overall standards process, which in turn distresses developers who then have to make design choices based on those standards that are currently available. This causes developers to use multiple alternatives, which reduces the acceptance of the later-released standards. There are many precedents where well-developed standards have not been taken-up, at least partially due to their late emergence, such as OSI [80] and HPF [81].

## Bibliography

[1] J. C. R. Licklider, "Man-Computer Symbiosis," IRE Trans. on Human Factors in Electronics, v. HFE-1, pp. 4--11, Mar. 1960
[2] http://tools.ietf.org/html/rfc707 (http://tools.ietf.org/html/rfc707)
[3] M. J. Litzkow, "Remote UNIX — Turning Idle Workstations into Cycle Servers," Proc. of USENIX, pp. 381--384, Sum. 1987
[4] M. Litzkow, M. Livny, M. Mutka, "Condor — A Hunter of Idle Workstations," Proc. of 8th Int. Conf. of Dist. Comp. Sys., pp. 104--111, Jun. 1988
[5] V. S. Sunderam, "PVM: A Framework for Parallel Distributed Computing," Concurrency: Prac. and Exp., v. 2(4), pp. 315--339, Dec. 1990
[6] Gigabit Testbed Initiative Final Report, 1996. (http://www.cnri.reston.va.us/gigafr/)
[7] I. Foster, J. Geisler, W. Nickless, W. Smith, S. Tuecke, "Software Infrastructure for the I-WAY High Performance Distributed Computing Experiment," Proc. 5th IEEE Symposium on High Performance Distributed Computing, pp. 562--571, 1997.
[8] The Globus Project (http://www.globus.org/)
[9] The Globus Alliance (http://www.globus.org/alliance/)
[10] I. Foster, C. Kesselman, S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," Lecture Notes in Computer Science, v. 2150, 2001.
[11] I. Foster, C. Kesselman, J. Nick, S. Tuecke "The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration, 2002.
[12] Legion (http://legion.virginia.edu/)
[13] Grimshaw, A. S., Wulf, W. A., "The Legion Vision of a Worldwide Virtual Computer," Comm. of the ACM, v. 40(1), January 1997.
[14] The Mentat project (http://www.cs.virginia.edu/~mentat/)
[15] Sybase Avaki EII (http://www.sybase.com/products/developmentintegration/avakieii)
[16] UNICORE (http://www.unicore.org/)
[17] Grid Engine open source project website (http://www.sun.com/software/gridware/)

[18] UNICORE Plus (http://www.fz-juelich.de/unicoreplus/)
[19] GRIP (http://www.fz-juelich.de/zam/cooperations/grip)
[20] UniGrids (http://www.unigrids.org/)
[21] Enterprise Grid Alliance, "EGA Reference Model and Use Cases v1.5" (http://www.gridalliance.org/en/WorkGroups/ReferenceModel.asp)
[22] Enterprise Grid Alliance,"EGA Grid Security Requirements v1.0" (http://www.gridalliance.org/en/WorkGroups/GridSecurity.asp)
[23] Enterprise Grid Alliance, "Enterprise Data and Storage Provisioning Problem Statement and Approach," (http://www.gridalliance.org/en/WorkGroups/DataandStorageProvisioningRequirements.asp)
[24] Jeffrey Hutzelman, Jospeh Salowey, Joseph Galbraith, and Von Welch, "RFC4462: Generic Security Service Application Program Interface (GSS-API) Authentication and Key Exchange for the Secure Shell (SSH) Protocol," In RFC4462, Internet Engineering Task Force, 2006.
[25] S. Tuecke, V. Welch, D. Engert, L. Perlman, M. Thompson, "RFC3820: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile," In RFC3820, Internet Engineering Task Force, 2004.
[26] Web Services (http://www.w3.org/2002/ws/)
[27] SOAP version 1.2 (http://www.w3.org/TR/2002/WD-soap12-part0-20020626/)
[28] WSDL (http://www.w3.org/TR/wsdl)
[29] UDDI (http://uddi.org/pubs/uddi_v3.htm#_Toc85907967)
[30] WS-RF Primer, (http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf)
[31] WS-ResourceProperties (WS-RP) (http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-06.pdf)
[32] WS-ResourceLifetime (WS-RL) (http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-03.pdf)
[33] WS-ServiceGroup (WS-SG) (http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ServiceGroup-1.2-draft-02.pdf)
[34] WS-BaseFaults (WS-BF) (http://docs.oasis-open.org/wsrf/wsrf-ws_base_faults-1.2-spec-pr-01.pdf)
[35] WS-Addressing (http://www.w3.org/Submission/ws-addressing/)
[36] WS-BaseNotification, March 2004 (ftp://www6.software.ibm.com/software/developer/library/ws-notification/WS-BaseN.pdf)
[37] WS-BrokeredNotification, March 2004 (ftp://www6.software.ibm.com/software/developer/library/ws-notification/WS-BrokeredN.pdf)
[38] WS-Topics, March 2004 (ftp://www6.software.ibm.com/software/developer/library/ws-notification/WS-Topics.pdf)
[39] Web Services Resource Transfer (WS-RT) (http://devresource.hp.com/drc/specifications/wsrt/WS-ResourceTransfer-v1.pdf)
[40] I. Foster, C. Kesselman, J. M. Nick, S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration" (http://www.globus.org/alliance/publications/papers/ogsa.pdf)
[41] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich, "The Open Grid Services Architecture, Version 1.0" (http://www.gridforum.org/documents/GWD-I-E/GFD-I.030.pdf)
[42] I. Foster, D. Gannon, H. Kishimoto, J. J. Von Reich, "Open Grid Services Architecture Use Cases" (ttp://www.gridforum.org/documents/GWD-I-E/GFD-I.029v2.pdf)
[43] H.Kishimoto, J. Treadwell,"Defining the Grid: A Roadmap for OGSA\texttrademark\ Standards: Version 1.0" (http://www.ogf.org/documents/GFD.53.pdf)
[44] S. Tuecke, K. Czajkoski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, D. Snelling, P. Vanderbilt, "Open Grid Services Infrastructure (OGSI): Version 1.0" (http://www.ogf.org/documents/GFD.15.pdf)
[45] Open Grid Service Infrastructure Primer (http://tinyurl.com/yss7tp)
[46] I. Foster, T. Maguire, D. Snelling, "OGSA WS-RF Basic Profile 1.0" (http://www.ogf.org/documents/GFD.72.pdf)
[47] T. Maguire, D. Snelling, "OGSA Profile Definition Version 1.0" (http://www.ogf.org/documents/GFD.59.pdf)

[48] M. Roehrig, M. Ziegler, "Grid Scheduling Dictionary of Terms and Keywords" (http://www.ogf.org/documents/GFD.11.pdf)
[49] R. Yahyapour, P. Wieder,"Grid Scheduling Use Cases" (http://www.ogf.org/documents/GFD.64.pdf)
[50] F. B. Maciel, "Resource Management in OGSA" (http://www.ogf.org/documents/GFD.45.pdf)
[51] D. Bell, T. Kojo, P. Goldsack, S. Loughran, D. Milojicic, S. Schaefer, J. Tatemura, P. Toft, "Configuration Description, Deployment, and Lifecycle Management (CDDLM) Foundation Document" (http://www.ogf.org/documents/GFD.50.pdf)
[52] P. Goldsack, "Configuration Description, Deployment, and Lifecycle Management: SmartFrog-Based Language Specification" (http://www.ogf.org/documents/GFD.51.pdf)
[53] S. Schaefer, "Configuration Description, Deployment, and Lifecycle Management: Component Model: Version 1.0" (http://www.ogf.org/documents/GFD.65.pdf)
[54] S. Loughran, "Configuration Description, Deployment, and Lifecycle Management: CDDLM Deployment API" (http://www.ogf.org/documents/GFD.69.pdf)
[55] M. Antonioletti, M. Atkinson, A. Krause, S. Laws, S. Malaika, N. W. Paton, D. Pearson, G. Riccardi, "Web Services Data Access and Integration — The Core (WS-DAI) Specification, Version 1.0" (http://www.ogf.org/documents/GFD.74.pdf)
[56] M. Antonioletti, B. Collins, A. Krause, S. Laws, J. Magowan, S. Malaika, N. W. Paton, "Web Services Data Access and Integration â – The Relational Realisation (WS-DAIR) Specification, Version 1.0" (http://www.ogf.org/documents/GFD.76.pdf)
[57] M. Antonioletti, S. Hastings, A. Krause, S. Langella, S. Lynden, S. Laws, S. Malaika, N. W. Paton, "Web Services Data Access and Integration â – The XML Realization (WS-DAIX) Specification, Version 1.0" (http://www.ogf.org/documents/GFD.75.pdf)
[58] W. Allcock, J. Bester, J. Bresnahan, S. Meder, P. Plaszczak, S. Tuecke,"GridFTP: Protocol Extensions to FTP for the Grid" (http://www.ogf.org/documents/GFD.20.pdf)
[59] I. Mandrichenko, W. Allcock, T. Perelmutov,"GridFTP v2 Protocol Description" (http://www.ogf.org/documents/GFD.47.pdf)
[60] R. Siebenlist, V. Welch, S. Tuecke, I. Foster N. Nagaratnam, P. Janson, J. Dayka, A. Nadalin, "OGSA Security Roadmap (Draft)" (http://www.cs.virginia.edu/~humphrey/ogsa-sec-wg/ogsa-sec-roadmap-v13.pdf)
[61] V. Welch, F. Siebenlist, D. Chadwick, S. Meder, L. Pearlman, "OGSA Authorization Requirement" (http://www.ogf.org/documents/GFD.67.pdf)
[62] GSI Working Group (https://forge.gridforum.org/projects/gsi-wg)
[63] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke, "A Security Architecture for Computational Grids," Proc. 5th ACM Conference on Computer and Communications Security Conference, pp. 83--92, 1998.
[64] Grimoires (http://www.ecs.soton.ac.uk/research/projects/grimoires)
[65] ebXML Registry Services Specification v2.5 (http://www.oasis-open.org/committees/regrep/documents/2.5/specs/ebrs-2.5.pdf)
[66] ebXMLsoft Registry and Repository (http://www.ebxmlsoft.com/)
[67] REST (http://en.wikipedia.org/wiki/REST)
[68] JDSL (https://forge.gridforum.org/projects/jsdl-wg/)
[69] JSDL-doc (http://www.gridforum.org/documents/GFD.56.pdf)
[70] DRMAA (http://www.drmaa.org)
[71] SAGA (http://www.ogf.org/gf/group_info/view.php?group=saga-rg)
[72] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludaescher, S. Mock,"Kepler: An Extensible System for Design and Execution of Scientific Workflows," Proc. of 16th Int. Conf. on Sci. and Statistical Database Management (SSDBMÃ 04), pp. 423--424, 2004
[73] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, P. Li, "Taverna: A Tool for the Composition and Enactment of Bioinformatics Workflows," Bioinformatics J., v. 20(17), pp. 3045--3054, 2004
[74] K. Amin, G. vonLaszewski, "GridAnt: A Grid Workflow System,"Argonne National Laboratory, Feb 2003
[75] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, S.Patil, M. Su, K. Vahi, M. Livny, "Pegasus: Mapping Scientific Workflows onto the Grid," Across Grids Conference 2004
[76] DAIS-WG (https://forge.gridforum.org/projects/dais-wg)

[77] OGSA-DAI (http://www.ogsadai.org.uk)
[78] OGSA-DQP (http://www.ogsadai.org.uk/about/ogsa-dqp/)
[79] K. Cline, J. Cohen, D. Davis, D. F. Ferguson, H. Kreger, R. McCollum, B. Murray, I. Robinson, J. Schlimmer, J. Shewchuk, V. Tewari, W. Vambenepe, "Toward Converging Web Service Standards for Resources, Events, and Management" (http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/Harmonization_Roadmap.pdf)
[80] ISO standard 7498-1, 1994 (http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip)
[81] High Performance Fortran standards (http://hpff.rice.edu/versions/)
[82] Globus Toolkit 3 Programmer's Tutorial, Key Concepts: WSRF & GT4 (http://gdp.globus.org/gt3-tutorial/multiplehtml/ch01s05.html)
[83] Globus Toolkit 3 Programmer's Tutorial, Key Concepts: OGSA, WSRF, and GT4 (http://gdp.globus.org/gt4-tutorial/multiplehtml/ch01s01.html)
[84] Web Services Standards as of Q1 2007 (http://www.innoq.com/resources/ws-standards-poster/)

# What Grids Can Do For You

## Payoffs and tradeoffs

The goal of grids is to enable and to simplify access to distributed resources. Based on the electric power grid as a model, a strong concept behind the development of grid technology is to provide a basic computational infrastructure that users could draw on for computation, visualization and data services. A person plugs in a toaster, radio or other appliance, without worrying about where the power is coming from or how it gets to them. In an ideal world, grid infrastructure would enable computational resources, data services, and even specialized instrumentation or sensors to be "plugged into" the grid, with user interfaces similarly "plugged in" to provide access without users needing to worry about many of the details as to where the devices, services or data reside. The challenge of grids is that the resources involved are distributed across a wide area, are administered and controlled by a variety of individuals and organizations, and adhere to a variety of usage policies and procedures. In addition, the performance characteristics and benefits will vary in that some grids are used to facilitate access to HPC resources (supercomputers), some bring together commodity computing capability, and all are dependent on the performance and reliability of the system-level, local, and also wide area network interconnects that tie them together.

In this chapter, we consider the cost-benefit analysis in terms of the effort required to coordinate the use of a heterogeneous set of resources that exist across administrative domains. That is, what makes such an extensive effort of coordination and software development (i.e., middleware) worth while? What are the tradeoffs that must be considered for an organization in the process of deciding whether or not to deploy or use resources on a grid? In this chapter, we will discuss some of the issues in general terms, with more detail further on in the cookbook.

**Access to resources beyond those locally available**

If a researcher were offered access to compute clusters, visualization engines, and a multitude of databases beyond what was locally available, most would be at least cautiously interested. Commonly anticipated advantages from an end-user perspective include:

- Improved model resolution resulting from access to greater compute power
- Increased size or number of calculations or applications that can be executed simultaneously
- Access to specialized visualization resources, allowing the rendering of complex scientific results in forms more easily interpreted by researchers
- Access to large amounts of preprocessed and well organized data across high speed networks and the ability to participate in and contribute to large, geographically dispersed research collaborations

Some difficulty arises, however, from the fact that resources on a grid are not often owned or controlled by a single administrative domain. This can affect the "cost" of computing — in terms of ready access, ease-of-use or even actual financial cost — beyond what may be initially obvious. Even so, grid computing arguably provides its greatest benefit when aggregating resources across project or organizations, enabling individuals within participating organizations to share resources and knowledge at unprecedented levels. There are a variety of regional, national and international-scale grid initiatives that provide shared access to specialized and general grid computing capabilities in support of the research and education mission. Later in this section we will provide several examples of existing grid initiatives providing a variety of services.

An alternate perspective on the inter-organizational sharing of resources comes from organizational management, who may ask "Why should I provide others with access to machines that came at my institution's cost and in response to specific needs and requests from my institution's users?" This question comes up time and again as institutions — or even departments within an institution — contemplate adding significant resources to a grid that is beyond their local domain. Accumulating resources locally may initially seem to be the most effective approach to meeting local needs, however, the drive for increased capability and diversity within a growing community can rapidly outpace local budget and resources for system acquisition and maintenance. Sharing resources through an inter-organizational grid can be a more cost-effective way to meet ranging and evolving local needs while increasing the capabilities available to the community at large. In addition, sharing resources with other organizations can provide users with access to a multiplicity of compute architectures and other types of resources not locally available, and, as importantly, to a larger community of potential collaborators and relationships for both technological and scientific advancement.

A notable challenge in the sharing of resources across institutions is determining the identity of users from different organizations so that local as well as grid-wide access and authorization policies can be applied. The successful coordination of authentication and authorization mechanisms with identity management technologies is key. For instance, Globus leverages Public Key Infrastructure (PKI) [1] as a basis for its management of access to grid resources. PKI offers a framework for organizations to share and trust assertions of identity through the exchange of digital certificates supported by public and private digital keys. If one's organization already utilizes PKI for identity management and is joining a grid that is Globus-based, integration at this level is fairly straightforward. If not, processes and technologies for mapping or converting organizational identities into appropriate PKI-based credentials need to be established. While this may not be complex in all situations, an organization must have sufficient IT resources and expertise to evaluate possible solutions, and, ideally, integration and cooperation with those who manage and administer the organization's existing identity management system(s).

### Performance and speedup

Computational resources, and specifically high performance systems or clusters, are often the first type of resource one thinks of at the mention of a grid. High performance, high-end, "super" computing has been around for a long time. It can be difficult for an organization to engage its diverse audience in an effort to construct HPC infrastructure at a campus. It is easier to engage in these discussions in the context of establishing a grid, especially since the grid offers the potential of making compute resources available to a larger community as well as augmenting the resources available at its member institutions.

The tradeoff here is that the grid doesn't always provide a complete solution. Cross platform schedulers, accounting, message passing paradigms, and so forth are required. Ongoing work in both standards and product development is attempting to bridge these gaps and much of the detail can now be hidden from the user through th euse of web services and interfaces. Joining a grid and accessing it through web services will be covered in significant detail later.

### Collaboration

As noted earlier, groups within an individual institution may be too small to justify or fund the type of resources they need and, in fact, they may only need those resources from time to time. As sponsoring agencies began to fund broader collaborations, the idea of "communities" evolved. Communities generally come in a number of categories such as "interest", "practice", "purpose" and so forth. (See Wikipedia "Community of interest" [2] for more explanation.) In our case, the people in these communities share interest, practice, purpose [and so forth] in a particular field of science or engineering. Grids help these communities build and share resources as well. The payoffs are in sharing knowledge, building expertise together (in both their shared area as well as in grid use), and enabling the community to build better cases together for more resources. The tradeoff is the increased complexity and management that grid use brings in order to use those resources. In this cookbook we will attempt to bridge the gaps and smooth out some of the complexity in the most simple terms possible.

**Alignment with National Vision for 21st Century Discovery**

In the National Science Foundation's recent report, "Cyberinfrastructure Vision for 21st Century Discovery", the term cyberinfrastructure is defined as, "... computing systems, data, information resources, networking, digitally enabled-sensors, instruments, virtual organizations, and observatories." From Arden Bement's introduction to this report:

> "At the heart of the cyberinfrastructure vision is the development of a cultural community that supports peer-to-peer collaboration and new modes of education based upon broad and open access to leadership computing; data and information resources; online instruments and observatories; and visualization and collaboration services. Cyberinfrastructure enables distributed knowledge communities that collaborate and communicate across disciplines, distances and cultures. These research and education communities extend beyond traditional brick-and-mortar facilities, becoming virtual organizations that transcend geographic and institutional boundaries."

Clearly grid computing will have a central role in the development of the cyberinfrastructure capabilities envisioned by the NSF. Understanding the basics of grid computing and working with collaborative teams of scientists and computing professionals to use and help develop grid computing tools and techniques will be an increasingly important component of a successful agency funding strategy.

# Examples of Evolving Grid-based Services and Environments

## Aggregating computational resources

A grid layer can make otherwise separate, distributed and different computational hardware appear as a single, common resource to which the user can submit jobs in a standard way. For instance, users may submit a genome alignment application via a grid portal and the job will run on any of several clusters, whether those clusters are at one university or another, or whether the operating systems are different versions.

Several examples of projects that are developing frameworks and toolkits for aggregating resources include:

- TeraGrid — From the TeraGrid website [57]: "TeraGrid is an open scientific discovery infrastructure combining leadership class resources at nine partner sites to create an integrated, persistent computational resource. Using high-performance network connections, the TeraGrid integrates high-performance computers, data resources and tools, and high-end experimental facilities around the country. Currently, TeraGrid resources include more than 250 teraflops of computing capability and more than 30 petabytes of online and archival data storage, with rapid access and retrieval over high-performance networks. Researchers can also access more than 100 discipline-specific databases. With this combination of resources, the TeraGrid is the world's largest, most comprehensive distributed cyberinfrastructure for open scientific research."

  TeraGrid is coordinated through the Grid Infrastructure Group (GIG) at the University of Chicago, working in partnership with the Resource Provider sites: Indiana University, Oak Ridge National Laboratory, National Center for Supercomputing Applications, Pittsburgh Supercomputing Center, Purdue University, San Diego Supercomputer Center, Texas Advanced Computing Center, University of Chicago/Argonne National Laboratory, and the National Center for Atmospheric Research.
- SURAgrid — From the SURAgrid website [8], "SURAgrid is a consortium of organizations collaborating and combining resources to help bring grid technology to the level of seamless, shared infrastructure. The vision for SURAgrid is to orchestrate access to a rich set of distributed capabilities in order to meet diverse users' needs. Capabilities to be cultivated include locally contributed resources, project-specific tools and environments, highly specialized or HPC access, and gateways to

national and international cyberinfrastructure. SURAgrid resources currently include over 10 teraflops of pooled computing resources, accessed through a common SURAgrid portal using a common authentication and authorization mechanism, the SURAgrid Bridge Certificate Authority."

- Geodise — The Geodise project [3], aimed initially at Computational Fluid Dynamics (CFD) applications, has the mission "To bring together and further the technologies of Design Optimisation, CFD, GRID computation, Knowledge Management & Ontology in a demonstration of solutions to a challenging industrial problem". Funded by the Engineering and Physical Sciences Research Council (EPSRC) [4] in the United Kingdom (UK), Geodise involves multidisciplinary teams working on a state of the art design tool demonstrator. Intelligent design tools will steer the user through set up, execution, post-processing, and optimization activities. These tools are physically distributed, under the control of multiple elements, to improve design processes that can require assimilation of terabytes of distributed data.

- Elastic Compute Cloud — Brush up that Amazon account! They aren't just about books and CDs anymore.

  Amazon Web Services [9] now provides application and service developers with direct access to Amazon's technology platform. From their website, "Build on Amazon's suite of web services to enable and enhance your applications. We innovate for you, so that you can innovate for your customers." Their Solutions catalog [10] shows services such as E-Commerce, Simple Storage, and so forth. Their Elastic Compute Cloud [11] (Amazon EC2) service is "a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers." Known also as *utility computing* by other service providers, Amazon EC2 presents a virtual computing environment that allows you to use web service interfaces to requisition machines for use, load them with your custom application environment, manage your network's access permissions, and run your image using as many or few systems as you desire. Pricing is per instance-hour consumed, per GB of storage transferred to/from Amazon, and per GB-month of Amazon S3 (Simple Storage Solution) used.

  InfoWorld's [12] article Amazon.com's rent-a-grid [13] provides an interesting and compact summary of the service. To quote them, "As the service's name suggests, though, if you need an elastic capability that can nimbly grow or shrink, EC2 is the only game in town." The author quickly points out that 3Tera [14] is coming out with their AppLogic grid system [15] soon though.

## Improved access for data-intensive applications

In an ideal world, a grid user may start up a data-intensive application and the grid will assemble the data streams combining data from multiple, distributed sources, so that the user experiences fast responses and sees the data as a logical whole. Several service components are needed to realize that vision, including data discovery, storage, possibly replication and version control, and reliable data transfer.While still developing towards the ideal, current data grids can manage access to data that may have been collected and stored at different locations, and provide controlled, secure access for communities as well as individuals. A grid workflow can be developed to manage data integration transparently for the user, or handle data access such that an application can process the data with improved throughput.

Applications in fields such as high energy physics (HEP), life sciences, and climate and weather modeling not only use but also generate massive amounts of data. These compute intensive applications can realize great benefit from access to an expanded pool of computational and data storage and management resources brought together using grid technology. In this section we will concentrate on the data side of that puzzle.

The International Virtual Data Grid Laboratory (iVDgL) [16] was a global data grid that served forefront experiments in physics and astrophysics. Its resources were comprised of heterogeneous computing and storage. Networking resources spanned the U.S., Europe, Asia and South America, thus providing a unique laboratory that tested and validated Grid technologies at international and global scales. The iVDgL was operated as a single system for the purposes of interdisciplinary experimentation in Grid-enabled, data-intensive scientific computing. Its goal was to drive the development, and transition to every day production use, of Petabyte-scale virtual data applications.

Applications that made use of the iVDgL include:

- Compact Muon Solenoid (CMS) [17] — an experiment at the Large Hadron Collider (LHC) [18] at CERN [19] in Geneva Switzerland. U.S. CMS [20] is a collaboration of U.S. scientists participating in CMS. This collaboration includes scientists at universities and Fermi National Accelerator Laboratory (FNAL) [21]. As their website states "The CMS experiment is designed to study the collisions of protons at a center of mass energy of 14 TeV. The physics program includes the study of electroweak symmetry breaking, investigating the properties of the top quark, a search for new heavy gauge bosons, probing quark and lepton substructure, looking for supersymmetry and exploring other new phenomena." [U.S. CMS Overview [22]]
- A Toroidal LHC ApparatuS (ATLAS) [23] — another experiment at the LHC, ATLAS is also designed to detect particles created by the proton-proton collisions, " the main goal for ATLAS is to look for a particle dubbed Higgs, which may be the source of mass for all matter. Findings may also offer insight into new physics theories as well as a better understanding of the origin of the universe." [U. S. ATLAS] [24]. U.S. Atlas includes scientists at universities and Brookhaven National Laboratory (BNL) [25].
- The Sloan Digital Sky Survey (SDSS) [26] — when completed, SDSS will provide detailed optical images covering more than a quarter of the sky, and a 3-dimensional map of about a million galaxies and quasars. The SDSS is managed by the Astrophysical Research Consortium for its participating institutions, including universities, museums, and laboratories. The SDSS data server, SkyServer [27], holds two primary databases: BESTDR1 and TARGDR1. An identical schema is used for both, but BESTDR1 has been processed with the "best available software" for handling noise and is therefore somewhat bigger. Combined the databases take over 800 GB of storage which is over 3.4 billion rows (records) [28]. SDSS is now up to Data Release 5 [29].

iVDgL sites in Europe and the U.S. were linked by a multi-gigabit per second transatlantic link funded by the European DataTAG project [30].

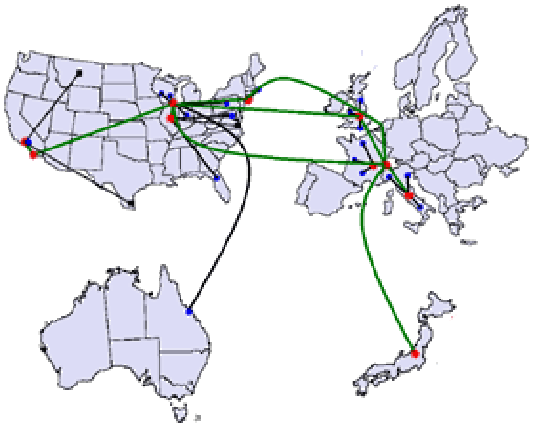


Figure WGD-3. iVDgL Project map.

(Interesting fact discovered while drafting this summary: "A TeV is a unit of energy used in particle physics. 1 TeV is about the energy of motion of a flying mosquito. What makes the LHC so extraordinary is that it squeezes energy into a space about a million million times smaller than a

- mosquito." [31])

- The EU-DataGrid Project [32], funded by the European Union, had as its purpose " to build the next generation computing infrastructure providing intensive computation and analysis of shared large-scale databases, from hundreds of TeraBytes to PetaBytes, across widely distributed scientific communities." A collaboration of about twenty European research institutes, DataGrid fulfilled its objectives in March of 2004 and moved on to become the EGEE (Enabling Grids for E-sciencE) [33].

  The DataGrid project focused on three application areas:

  - ♦ High Energy Physics — As has iVDgL, DataGrid set the stage for handling the huge amounts of data produced by the LHC. A multi-tiered, hierarchical computing model has been adopted to share data and computing efforts among multiple institutions. The Tier-0 center is located at CERN and is linked by high speed networks to approximately ten major Tier-1 data processing centers. These fan out the data to a large number of smaller centers known as Tier-2s.
  - ♦ Biology and Medical Image Processing — The DataGrid project's biology testbed provided the platform for new algorithms on data mining, databases, code management, graphical interface tools and facilitated sharing of genomic and medical imaging databases for the benefit of international cooperation and health care.
  - ♦ Earth Observations — The European Space Agency missions involve the download, from space to ground, of about 100 Gigabytes of raw images per day. Dedicated ground infrastructures have been set up to handle the data produced by instruments onboard the satellites.
    DataGrid demonstrated an improved way to access and process large volumes of data stored in distributed European-wide archives.

  See the DataGrid Project Description [34] for more information.

- Looking at it from another perspective, projects like OGSA-DAI [35] develop middleware to assist with access and integration of data from separate sources via the grid. Directly from their website, "OGSA-DAI is motivated by the need to:
  - ♦ Allow different types of data resources — including relational, XML and files — to be exposed onto Grids.
  - ♦ Provide a way of querying, updating, transforming and delivering data via web services.
  - ♦ Provide access to data in a consistent, data resource-independent way.
  - ♦ Allow metadata about data, and the data resources in which this data is stored, to be accessed.
  - ♦ Support the integration of data from various data resources.
  - ♦ Provide web services that can be combined to provide higher-level web services that support data federation and distributed query processing.
  - ♦ To contribute to a future in which scientists move away from technical issues such as handling data location, data structure, data transfer and integration and instead focus on application-specific data analysis and processing."

  Many grid projects are using OGSA-DAI including

  - ♦ LEAD [36] — Linked Environments for Atmospheric Discovery
  - ♦ caGrid [37] — the Cancer Biomedical Informatics Grid
  - ♦ AstroGrid [38] — a project to build an infrastructure for the Virtual Observatory (VObs)
  - ♦ BRIDGES [39] — Biomedical Research Informatics Delivered by Grid Enabled Services
  - ♦ eDiaMoND [40] — a Grid for X-Ray Mammography
  - ♦ GeneGrid [41] — exploiting existing micro array and sequencing technologies and the large volumes of data generated through screening services. to develop specialist tissue specific

datasets relevant to the particular type of disease being studied
- and more [42].

## Federation of shared resources toward global services

A particularly important aspect of the grid is that of support for "virtual organizations," or VOs. When the high-energy physics community began collaborating on large-scale physics problems, researchers from many different and widely separated organizations needed to work together. The problem domain was so vast that researchers at any one site needed the expertise from researchers at other sites in order to make progress. A project might represent dozens, hundreds or thousands of scientists collaborating together. The concept of the "virtual organization" recognized that such project groups would convene from various organizations and need to work together as if they were, in fact, from a single organization. In fact, VOs may be very dynamic and ad hoc, coming together for very specific purposes, working together for fixed time periods, adding and losing members over time.

Grid middleware can support sharing of resources using a federated approach, where participating organizations retain control over their local resources and services but also share these resources in a way that becomes globally scalable. For example, an institution would authenticate users locally for access to institutionally-controlled resources but leverage grid security infrastructure to enable those same users to access external grid resources. Additionally, users that are identified as members of a particular project, or VO, could be authorized to use resources in a way that has been pre-approved for members of that group.

- Funded by the National Science Foundation, the Computational Chemistry Grid [43], (CCG) has developed a java client to facilitate access to a controlled set of applications, HPC and storage resources for use by the computational chemistry community. Project partners include the Center for Computational Sciences at the University of Kentucky, the Center for Computation & Technology at Louisiana State University, the National Center for Supercomputing Applications (NCSA), Texas Advanced Computing Center (UT Austin) and the Ohio Supercomputer Center. From their Web site: "The 'Computational Chemistry Grid' (CCG) is a virtual organization that provides access to high performance computing resources for computational chemistry with distributed support and services, intuitive interfaces and measurable quality of service." Access is granted through an approval process, with allocations "available to US academic and government research staff and to non-US academic researchers." Three types of project allocations are available: research, community research and instructional. Research allocations are intended to support large, often multi-year scientific research projects. Community allocations are shorter term and intended to be used towards development of a larger research effort. Instructional allocations can be used to support academic instruction in the field.

- The cancer Biomedical Informatics Grid [44], (caBig) is a virtual organization of "over 800 people from approximately 50 NCI-designated Cancer Centers and other organizations" in a "voluntary network or grid...to enable the sharing of data and tools, creating a World Wide Web of cancer research." Development of the project is taking place under the leadership of the National Center Institute's Center for Bioinformatics and has the primary goal of "[speeding] the delivery of innovative approaches for the prevention and treatment of cancer". However, the concepts and technologies involved are also being developed with an eye towards reuse and adaptability outside of the cancer research community. Releases of software and components are publicly available on the project's community web site. A separate informational web site is available for those who are not intending to use services or tools but who are interested in knowing more about the initiative: http:cabig.cancer.gov [45].

- The Open Science Grid (OSG) [46], is an outgrowth of three notable physics projects — the DOE-funded Particle Physics Data Grid (www.ppdg.net), and the NSF-funded Grid Physics Network (GriPhyN, www.griphyn.org) and the International Virtual Data Grid Laboratory (iVDGL, www.ivdgl.org). Collaborators leading and within these projects became interested in the benefits of grid technology for disciplines beyond physics and began to develop their grid middleware and related services with an eye towards broader use. Today, the concept of a "virtual organization" is central to the conceptual as well as operational functioning of OSG, and there are well over two-dozen VOs participating, representing a variety of scientific fields. Organizations that contribute resources to OSG retain control of those resources but enable use by project groups through access management tools that have been designed around the VO concept. From their Web site: "A Virtual Organization (VO) is a collection of people (VO members), computing/storage resources (sites) and services (e.g., databases). In OSG, we typically use the term VO to refer to the collection of people, and the terms Site, Computing Element (CE), and/or Storage Element (SE) to refer to the resources owned and operated by a VO." As an organization itself, OSG is also focused on establishing interoperability with other grids, such as Teragrid, international, regional and campus grids.

## Harnessing unused cycles

Grids can enable an organization to capture the incredible amount of computing that exists in idle PCs and workstations. Users can use grid services to submit applications as if to a single resource — the grid manages submission to various computers, monitoring of status, and collection of the results.

Various tools, both open source and proprietary, exist to help an organization with this sort of grid-enabled service.

- Probably the most famous application is the cycle sharing application SETI@home [46]. SETI@home was proposed in 1995 and launched in 1999. As their website states "SETI (Search for Extraterrestrial Intelligence) is a scientific area whose goal is to detect intelligent life outside Earth. One approach, known as radio SETI, uses radio telescopes to listen for narrow-bandwidth radio signals from space. Such signals are not known to occur naturally, so a detection would provide evidence of extraterrestrial technology." SETI@home has developed a large community around their project and they include various statistics about their participants on their website.

  Today SETI@home uses software called BOINC [47]. BOINC has the expanded mission to use the idle time on your computer (Windows, Mac, or Linux) to cure diseases, study global warming, discover pulsars, and do many other types of scientific research. You can use the BOINC software to create your own project. Worldwide projects, such as the World Community Grid [48], use BOINC. As their mission states "World Community Grid's mission is to create the world's largest public computing grid to tackle projects that benefit humanity. Our work has developed the technical infrastructure that serves as the grid's foundation for scientific research. Our success depends upon individuals collectively contributing their unused computer time to change the world for the better. World Community Grid is making technology available only to public and not-for-profit organizations to use in humanitarian research that might otherwise not be completed due to the high cost of the computer infrastructure required in the absence of a public grid. As part of our commitment to advancing human welfare, all results will be in the public domain and made public to the global research community."
- Another well-known project is University of Wisconsin-Madison's Condor [49]. Condor is often used to manage clusters of dedicated processors, but it also has unique mechanisms that enable effective harnessing of wasted CPU power from otherwise idle desktop workstations.

  BOINC and Condor take very different approaching to the access and management of unused cycles. BOINC functions by enabling thousands or even millions of users to trust a small set of programs to run on their computer, typically leveraging the aggregate compute capacity towards the resolution of an overarching problem or inquiry. Condor harnesses unused cycles to run unspecified applications. This requires a deeper level of trust and so is likely to involve a smaller set of trusted computers. The benefit is the potential to run a much greater variety of applications, which significantly increases the utility of Condor as a high-throughput computing system.

Condor can

  - ♦ be configured to identify idle machines under various criteria
  - ♦ checkpoint and migrate jobs when those machines are no longer available
  - ♦ work in shared or non-shared file environment (that is, it can migrate files or retrieve from source as needed)

Condor also provides the job queueing mechanism, scheduling policy, priority scheme, resource monitoring, and resource management. So Condor provides seamless access to a combination of distributed computers.

- United Devices [50] offers a number of commercial HPC products. Relevant to the discussion is Grid-MP ™ [51] which is an infrastructure solution for implementing and managing complex enterprise grids. GRID MP deployments can be single cluster management implementations to large-scale multi-resource grids. Per United Devices, the GRID-MP system has scaled to hundreds of thousands of CPUs and hundreds of thousands of jobs and can scale to over thousands of users.

  Grid MP was built from the ground up to have a comprehensive security architecture that includes transparent data encryption, secure authentication, digital signatures and tamper detection. A framework for rapid application integration is also included, based on open web services and standards. The interface provides controlled access to all aspects of the grid system. The system is designed for self-management via a web-based console, allowing administrative access from anywhere. Grid MP devices and users can be grouped with maximum flexibility. An administrator can set up priority allocation and provisioning policies.

## High-speed optical networking, network-aware applications

As noted in the "Networks, switches and interconnects for grids" section of this Cookbook, "...networks are the virtual bus for the virtual grid computer and are central to the efficient, effective operation of grids." As grids evolve, they are beginning to use high bandwidth optical networks to interconnect grid nodes, increasing the speed and efficiency possible between input/output, CPUs, storage and other elements of the computational process. We are also seeing the advent of "smart" applications — those that are able to actively (or even proactively!) evaluate network conditions and react with dynamic adjustments to insure successful operation. Both of these trends can improve performance and thru-put as perceived by the users of grid applications today, however, they also hold great promise for the future. Some people feel that, to truly realize the potential of grid technology, applications, middleware and network services must interact much more frequently, intelligently and seamlessly than they do today, to produce an adaptive capability much more akin to using a single computer than distributing a problem across multiple systems. Several concepts mentioned in the "Networks, switches and interconnects for grid" section (virtual and dynamic circuits, advanced monitoring, end-to-end performance, QoS) form a foundation for further development in this area. In addition to the several project examples provided in the "Networks..." section, the following projects are exploring innovations relevant to the advancement of grid technology:

- The focus of the Enlightened Computing [52] (Highly-dynamic Applications Driving Adaptive Grid Resources) project is "...on developing dynamic, adaptive, coordinated and optimized use of networks connecting geographically distributed high-end computing resources and scientific instrumentation. A critical feedback-loop consists of resource monitoring for discovery, performance, and SLA compliance, and feed back to co-schedulers for coordinated adaptive resource allocation and coscheduling... For this project we have assembled a global alliance of partners to develop, test, and disseminate advanced software and underlying technologies which provide generic applications with the ability to be aware of their network, Grid environment and capabilities, and to make dynamic, adaptive and optimized use of networks connecting various high end resources. We will develop advanced software and Grid middleware to provide the vertical integration starting from the application down to the optical control plane."

- From the Optiputer [53] website: "The OptIPuter, so named for its use of Optical networking, Internet Protocol, computer storage, processing and visualization technologies, is an envisioned infrastructure that will tightly couple computational resources over parallel optical networks using the IP communication mechanism. The OptIPuter exploits a new world in which the central architectural element is optical networking, not computers — creating "supernetworks". This paradigm shift requires large-scale applications-driven, system experiments and a broad multidisciplinary team to understand and develop innovative solutions for a "LambdaGrid" world. The goal of this new architecture is to enable scientists who are generating terabytes and petabytes of data to interactively visualize, analyze, and correlate their data from multiple storage sites connected to optical networks."

- From the CANARIE*4 [54] website (and the concept of customer-empowered networks [55]), "CA*net 4 will, as did its predecessor CA*net 3, interconnect the provincial research networks [of Canada], and through them universities, research centers, government research laboratories, schools, and other eligible sites, both with each other and with international peer networks. Through a series of point-to-point optical wavelengths, most of which are provisioned at OC-192 (10 Gbps) speeds, CA*net 4 will yield a total initial network capacity of between four and eight times that of CA*net 3...CA*net 4 will embody the concept of a "customer-empowered network" which will place dynamic allocation of network resources in the hands of end users and permit a much greater ability for users to innovate in the development of network-based applications. These applications, based upon the increasing use of computers and networks as the platform for research in many fields, are essential for the national and international collaboration, data access and analysis, distributed computing, and remote control of instrumentation required by researchers."

# A Future View of "the Grid"

In an article in Scientific American [56], Ian Foster describes just how ubiquitous and transparent grids might be in the future. "By linking digital processors, storage systems and software on a global scale, grid technology is poised to transform computing from an individual and corporate activity into a general utility" — a utility similar to water distribution and electrical power systems in both its value and the invisibility of the system itself to the consumer. Today's researchers, information technology staff and commercial vendors are transforming grid technology in such a way that what are presently exclusive high performance computing and data services, may one day be widely available via a pervasive, daily (and perhaps somewhat mundane) utility.

It was barely a 100 years ago that the average citizen could only fantasize about fully wired houses (what did "fully wired" mean a century ago?) with ubiquitous, "always on" electric power. It is perhaps not too fanciful to imagine how academia, industry or even individuals might have utilitarian access in the future to what are today expensive, complex high performance computing resources. Such a grid of computing and data services could have widespread and socially valuable effects on the world. Given the rapidity with which grid technology is maturing and being deployed, it is possible to imagine scenarios in which entire communities benefit from grid activities in both ordinary and extraordinary circumstances.

The following scenario, set in 2012 in the southeastern United States, imagines how a ubiquitous "grid of grids" (or "the Grid") would serve as part of the technical infrastructure supporting community health science and services. In this scenario, entire user application communities are able to realize the benefits of the Grid infrastructure. The Grid is envisioned as supporting multiple, general grid functions that include computation, data management, collaboration services and knowledge discovery. In this scenario, these functions specifically support:

- Pre-hospital data analysis
- Bioinformatics
- Medical records data mining and
- Bio-medical simulations

**News Release**
**September 12, 2012**
**Houston, Texas**
**Regional Grid Helps Heal Houston.**

*The aftermath of last week's category-4 tropical storm Hale has disrupted local services and displaced several hundred thousand citizens this week. While not reaching the devastation of 2005's category 5 storm Katrina, the city and surrounding area are severely impacted by wind, rain and flooding from the storm. Luckily the Katrina aftermath is not being replayed, in part because core Grid infrastructure allows vital services to continue seamlessly operating using other compute and data nodes on the broad grid-based cyberinfrastructure that now spans the southeastern United States.*

*The regional Grid cyberinfrastructure has a significant impact on the health care delivery systems in this city today. Though power outages from Hale have shut down many local computing facilities, the city's major hospitals are only minimally affected since they can use the Grid to access computing capabilities from sites across the southeast. Emergency first responders remain highly effective, receiving significant support from physicians in other states. Using grid-based telemedicine technologies for remote assessment of critical vital signs, local emergency medical teams work directly with remote physicians in determining medical triage decisions for the best medical care. Meanwhile, the scheduling and coordination of our city's patient care, involving the complex coordination of providers, equipment and facilities to match individual treatment requirements, uses a dynamic priority-based scheduler over the Grid. Using artificial intelligence, the scheduler helps manage and prioritize patient access to health care, expedites their treatment, and optimizes allocation of critical health care system resources. The complex algorithms to determine patient care decisions automatically find and run on the best available computing resources distributed across the southeast's regional grid, ensuring that patient wait times are kept to a minimum.*

*Patient outcomes from Hale-related injuries are being vastly improved, benefiting from early patient evaluations (pre-hospital data analysis) that medical first responders are able to upload directly to the grid from accident scenes. These evaluations are providing immediate, expansive physiologic readings on large numbers of trauma patients and helping ground-based medical first responders arrange air transports for the most critical patients. At trauma centers, the predictive ability of patient data is much more clinically relevant through the use of grid enabled data mining, neural networks, and decision tree analysis during the first 24 hours of admission. These grid-based systems feed physiologic databases with more useful, and patient specific, outcome data than the mere survival data typically used only a few years ago. Medical personnel are able to select the best treatment option.*

> *Improved clinical outcomes, based on identifying predictive input markers, are derived by running sophisticated algorithms against the extensive medical health records data grid. Now a key part of the health care system, medical records data mining is conducted on a rich set of records redundantly stored across the secure grid infrastructure — so Houston's records remain available even though the local systems are temporarily off-line. Using optical, point-to-point networks, these distributed medical records are accessible from highly secure databases that have been deployed across the regional grid. Moreover, medical records data is the foundation of an extensive and readily accessible knowledge base. For example, a large collection of radiological data is available along with relevant patient history, clinical and histological information, for retrieval and comparative interpretation using computer assisted diagnostic (CADx) systems and other visualization tools. Further, Houston's medical records (with all person-specific information removed) are included with other valuable health status demographics that are used by Problem Knowledge Coupler (PKC) systems. Such systems, valuable as an alternative teaching tool for diagnostic skill development, also are providing improved diagnostics for patients during the Hale aftermath. The PKC systems use grid-accessible medical data from thousands of prior medical cases to suggest recommended procedures and to extrapolate best practices*
>
> *Advanced bioinformatics and bio-medical simulation components of the southeastern Grid are also providing further benefits for Hale storm victims. In the first week after Hale, a rash began afflicting many of our city's residents. While initially confined to the Houston area, the illness soon spread to the neighboring Gulf Coast. Rumors about the 11th anniversary of 9/11 attacks and possible release of toxins by terrorists started to spread and threatened to complicate the area's storm relief efforts. Fortunately, a local medical research facility with a bioinformatics program worked with a team of biologists from other universities in the region and the Centers for Disease Control and Prevention in Atlanta. The team used dozens of the Grid's distributed computational resources to search many genomic and proteomic databases in parallel to identify the specific agent causing the rash.*
>
> *With the identification of a probable agent, the teams are applying biomedical simulation techniques across many Grid resources to analyze models of how the disease vectors propagate the agent involved. The simulations are using a cognitive reasoning system with an advanced conceptual modeling approach for nuclear, biological and chemical (NBC) threat assessment, predictive analysis, and decision-making. These models are showing medical teams how to stem the agent's spread and, indeed, these same models are enabling additional health care system personnel to receive preventative training.*
>
> *While the storm's impact on Houston and the surrounding area is definitely being felt, the overall experience has been significantly less difficult and traumatic due to the presence of a sophisticated grid across the southeast. The grid brings the southeast's extensive computation, data, simulation and collaboration resources together under a shared infrastructure that is serving emergency responders, medical teams and distributed health care systems to provide effective, patient-specific care that is so vital to minimizing long-term consequences to people and the region.*

Of course, this is a hypothetical scenario, yet the future reality may quite likely be more surprising than even as imagined above. Grid infrastructure is maturing and represents a significant sea change in how computation, simulation, bioinformatics, collaboration and knowledge are supported. The ability to access resources anywhere at anytime, with the ability to survive interruptions from local conditions, is an important benefit offered by grids as part of a global cyberinfrastructure. Building that imagined infrastructure will certainly depend on the contributions being made now in grid implementations and deployments.

# Bibliography

[1] Public Key Infrastructure (http://tinyurl.com/39kx4a)
[2] Community of interest (http://en.wikipedia.org/wiki/Community_of_interest)
[3] Geodise project (http://www.geodise.org/)
[4] Engineering and Physical Sciences Research Council (http://www.epsrc.ac.uk/default.htm)
[5] The Geodise Toolboxes, A User's Guide (http://www.geodise.org/documentation/html/index.htm)
[6] The Geodise Project: Making the Grid Usable Through Matlab (http://www.gridtoday.com/grid/343938.html)
[7] Grid Today (http://www.gridtoday.com/gridtoday.html)
[8] SURAgrid (http://www.sura.org/programs/sura_grid.html)
[9] Amazon Web Services (http://tinyurl.com/2sbgmv)
[10] [Amazon's] Solutions catalog (http://solutions.amazonwebservices.com/connect/index.jspa)
[11] [Amazon's] Elastic Compute Cloud (http://www.amazon.com/gp/browse.html?node=201590011)
[12] Infoworld (http://www.infoworld.com/)
[13] Amazon.com's rent-a-grid (http://www.infoworld.com/article/06/08/30/36OPstrategic_1.html)
[14] 3Tera (http://www.3tera.com/index.html)
[15] AppLogic grid system (http://www.infoworld.com/4449)
[16] International Virtual Data Grid Laboratory (http://www.ivdgl.org/)
[17] Compact Muon Solenoid (CMS) (http://cms.cern.ch/)
[18] Large Hadron Collider (LHS) (http://public.web.cern.ch/Public/Content/Chapters/AboutCERN/CERNFuture/WhatLHC/WhatLHC-en.html)
[19] CERN (http://public.web.cern.ch/Public/Welcome.html)
[20] U. S. CMS (http://www.uscms.org/)
[21] Fermi National Accelerator Laboratory (http://www.fnal.gov/)
[22] U. S. CMS Overview (http://www.uscms.org/Public/overview.html)
[23] A Toroidal LHC ApparatuS (ATLAS) (http://atlas.web.cern.ch/Atlas/index.html)
[24] U. S. ATLAS (http://www.usatlas.bnl.gov/)
[25] Brookhaven National Laboratory (BNL) (http://www.bnl.gov/world/)
[26] Sloan Digital Sky Survey (SDSS) (http://www.sdss.org/)
[27] SkyServer (http://cas.sdss.org/dr5/en/)
[28] SDSS Databases (http://cas.sdss.org/dr5/en/sdss/data/data.asp#databases)
[29] SDSS Data Release 5 (http://cas.sdss.org/dr5/en/sdss/release/)
[30] DataTAG (http://datatag.web.cern.ch/datatag/)
[31] TeV in layman's terms (http://public.web.cern.ch/Public/Content/Chapters/AboutCERN/CERNFuture/WhatLHC/WhatLHC-en.html)
[32] EU-DataGrid Project (http://web.datagrid.cnr.it/servlet/page?_pageid=1407&_dad=portal30&_schema=PORTAL30&_mode=3)
[33] Enabling Grids for E-sciencE (EGEE) (http://www.eu-egee.org/)
[34] DataGrid Project Description (http://web.datagrid.cnr.it/servlet/page?_pageid=873,879&_dad=portal30&_schema=PORTAL30&_mode=3)
[35] OGSA-DAI (http://www.ogsadai.org.uk/index.php)
[36] LEAD (http://www.lead.ou.edu/)
[37] caGrid (http://cabig.nci.nih.gov/)
[38] AstroGrid (http://www.astrogrid.org/)
[39] BRIDGES (http://www.brc.dcs.gla.ac.uk/projects/bridges/)
[40] eDiaMoND (http://www.ediamond.ox.ac.uk/)
[41] GeneGrid (http://www.qub.ac.uk/escience/projects/genegrid)
[42] more OGSA-DAI grid projects (http://www.ogsadai.org.uk/about/projects.php)
[43] Computational Chemistry Grid (https://www.gridchem.org)
[44] cancer Biomedical Informatics Grid (https://cabig.nci.nih.gov)
[45] caBIG (http:cabig.cancer.gov)

[46] SETI@home (http://setiathome.berkeley.edu/)
[47] BOINC (http://boinc.berkeley.edu/)
[48] World Community Grid (http://www.worldcommunitygrid.org/)
[49] Condor (http://www.cs.wisc.edu/condor/)
[50] United Devices (http://www.ud.com/)
[51] Grid-MP ™ (http://www.ud.com/products/gridmp.php)
[52] Enlightened Computing (http://enlightenedcomputing.org)
[53] Optiputer (http://www.optiputer.net)
[54] CANARIE*4 (http://www.canarie.ca/advnet)
[55] CANARIE*4 customer-empowered networks (http://www.canarie.ca/advnet/cen.html)
[56] Foster, Ian, "The Grid: Computing without Bounds", Scientific American, April 2003.
[57] Teragrid (http://www.teragrid.org)

# Grid Case Studies

## Grid Applications

### SCOOP Storm Surge Model

**Collaborators**

Lavanya Ramakrishnan, Renaissance Computing Institute
Brian O. Blanton, Renaissance Computing Institute
Howard M. Lander, Renaissance Computing Institute
Richard A. Luettich, Jr, UNC Chapel Hill Institute of Marine Sciences
Daniel A. Reed, Renaissance Computing Institute
Steven R. Thorpe, MCNC

**Summary**

Recently, large-scale ocean and meteorological modeling has resulted in the use of Grid resources and high performance environments for running these models. There is a need for an integrated system that can handle real-time data feeds, schedule and execute a set of model runs, manage the model input and output data, make results and status available to the larger audience. Here, we describe the distributed software infrastructure that we have built to run a storm surge model in a Grid environment. Our solution builds on existing standard grid and portal technologies including the Globus toolkit [2], Open Grid Computing Environment [4] (OGCE) and lessons learned from grid computing efforts in other science domains. Specifically, we implement specific techniques for resource management and increased fault tolerance due to the sensitivity of the application.

This framework was developed as a component of Southeastern Universities Research Association's (SURA) Southeastern Coastal Ocean Observing and Prediction [15] (SCOOP) program The SCOOP program is a distributed project that includes Gulf of Maine Ocean Observing System, Bedford Institute of Oceanography, Louisiana State University, Texas A&M, University of Miami, University of Alabama in Huntsville, University of North Carolina, University of Florida and Virginia Institute of Marine Science. SCOOP is creating an open-access grid environment for the southeastern coastal zone to help integrate regional coastal observing and modeling systems.

For full model details and more complete grid component descriptions, see SCOOP Storm Surge Model.

**Technology Components**

The front-end to the system is through a portal that provides the interface for users to interact with the ocean observing and modeling system. The real-time data for the ensemble forecast arrives through Unidata's Local Data Manager [10] (LDM), an event-driven data distribution system that selects, captures, manages and distributes meteorological data products. Once all the data for a given ensemble member has been received, available grid resources are discovered using a simple resource selection algorithm. After the files are staged, the model run is executed and the output data is staged back to the originating site. The final result of the surge computations is inserted back into the SCOOP LDM stream for subsequent analysis and visualization by other SCOOP partners [15a]. Thus specifically our architecture has the following Grid components:

- An Application Coordinator that acts as a central component that orchestrates the data and job management actions and interacts with the Globus services.
- A resource monitoring and notification framework that is used to collect monitoring data and track data flow status in the system.

- A resource selection API that queries grid resource to determine the best resources available to run each of the jobs.
- An application preparation component that prepares the application bundle that needs to be used on a remote resource.
- A front-end portal that allows users to conduct retrospective analysis, access historical data from previous model runs and observe the status of daily forecast runs from the portal

### Data and Control Flow of the NC SCOOP System

Before we describe in detail each of the components used in the framework, we briefly describe the control flow of our framework. The ADCIRC storm surge model can be run in two modes. The “forecast” mode is triggered by real-time data arrival of wind data from different sites through the Local Data Manager [10]. In the “hindcast” mode, the modeler can either use a portal or a shell interface to launch the jobs to investigate prior data sets (post-hurricane). The figure shows the architectural components and the control flow for the NC SCOOP system:

1. In the forecast mode the wind data arrives at the LDM node (Step 1.F. in figure). In our current setup, the system receives wind files from University of Florida and Texas A&M. Alternatively, a scientist might log into the portal and choose the corresponding data to re-run a model (Step 1.H. in figure).
2. In the hindcast run, the application coordinator locates relevant files using the SCOOP catalog at UAH[17] and retrieves them from the SCOOP archives located at TAMU and LSU[12]. In the forecast runs, once the wind data arrives, the application coordinator checks to see if the hotstart files are available locally or are available at the remote archive. If they are not available and not being generated currently (through a model run), a run is launched to generate the corresponding hotstart files to initialize the model for the current forecast cycle.
3. Once the model is ready to run (i.e. all the data is available), the application coordinator will use the resource selection component to select the best resource for this model run.
4. The resource selection component queries the status at each site and ranks the resources, accounting for queue delays and network connectivity between the resources.
5. The application coordinator then calls an application specific component that prepares an application package that can be shipped to remote resources. The application package is customized with specific properties for the application on a particular resource and includes the binary, the input files and other initialization files required for the model run.
6. The self-extracting application package is transferred to the remote resource and the job is launched using standard grid mechanisms.
7. Once the application coordinator receives the “job finished” status message, it retrieves the output files from the remote sites.
8. The results are then available through the portal (Step 8.H in figure). Additionally, in case of forecast mode, we push the data back through LDM (Step 8.F in figure) which is archived and visualized by other SCOOP partners downstream.
9. The application coordinator publishes status messages at each of the above steps to a centralized messaging broker. Interested components such as the portal can subscribe to relevant messages to receive real-time status notification of the job run.
10. In addition the resource status information is also collected across all the sites that can be observed through the portal as well as used for more sophisticated resource selection algorithms.

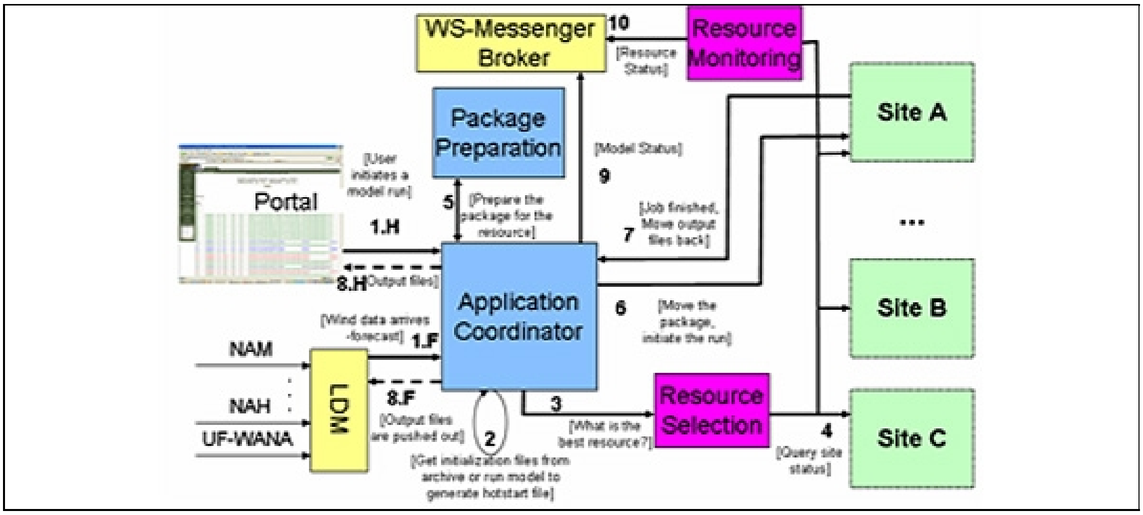

WS-Messenger Broker
10
[Resource Status]
Resource Monitoring
Package Preparation
[Model Status]
9
Site A
[User initiates a model run]
Portal
1.H
5
[Prepare the package for the resource]
7
[Job finished, Move output files back]
...
8.H
[Output files]
Application Coordinator
6
[Move the package, initiate the run]
Site B
[Wind data arrives -forecast]
1.F
NAM
NAH
UF-WANA
LDM
8.F
[Output files are pushed out]
2
[Get initialization files from archive or run model to generate hotstart file]
3
[What is the best resource?]
Resource Selection
4
[Query site status]
Site C

Figure CS-1. Architectural components and the control flow for the NC SCOOP system.

### Contact

scoop-support@renci.org, Renaissance Computing Institute.

### Acknowledgements

This framework was developed as a component of Southeastern Universities Research Association's (SURA) Southeastern Coastal Ocean Observing and Prediction (SCOOP) program [15]. The SCOOP program is a distributed project that includes numerous research partners [15a]. Funding for SCOOP has been provided by the Office of Naval Research, Award N00014-04-1-0721 and by the National Oceanic and Atmospheric Administration's NOAA Ocean Service, Award NA04NOS4730254. Full acknowledgements are provided in the detailed version of this paper, available in the Related Links section of this Cookbook.

## Open Science Grid

### Collaborators

The Open Science Grid consortium consists of around 23 member organizations and several partners. An up to date list can be found under the OSG Council [21] web page. The participants are called Virtual Organizations [22], or VOs, where a VO is a collection of people (VO members), computing/storage resources (sites) and services (e.g., databases.) Technical Activity [23] groups round out the organization through liaison, service and development activities.

### Introduction and Overview

Scientists from many different fields use the Open Science Grid to advance their research. The OSG Consortium includes members from particle and nuclear physics, astrophysics, bioinformatics, gravitational-wave science and computer science collaborations. Consortium members contribute to the development of the OSG and benefit from advances in grid technology. Applications in other areas of science, such as mathematics, medical imaging and nanotechnology, benefit from the OSG through its partnership with local and regional grids or their communities' use of the Virtual Data Toolkit software stack.

The following chart shows running applications as well as the current load on the OSG over a one week period. The subsequent sections in this case study will look a little further into several of these applications.

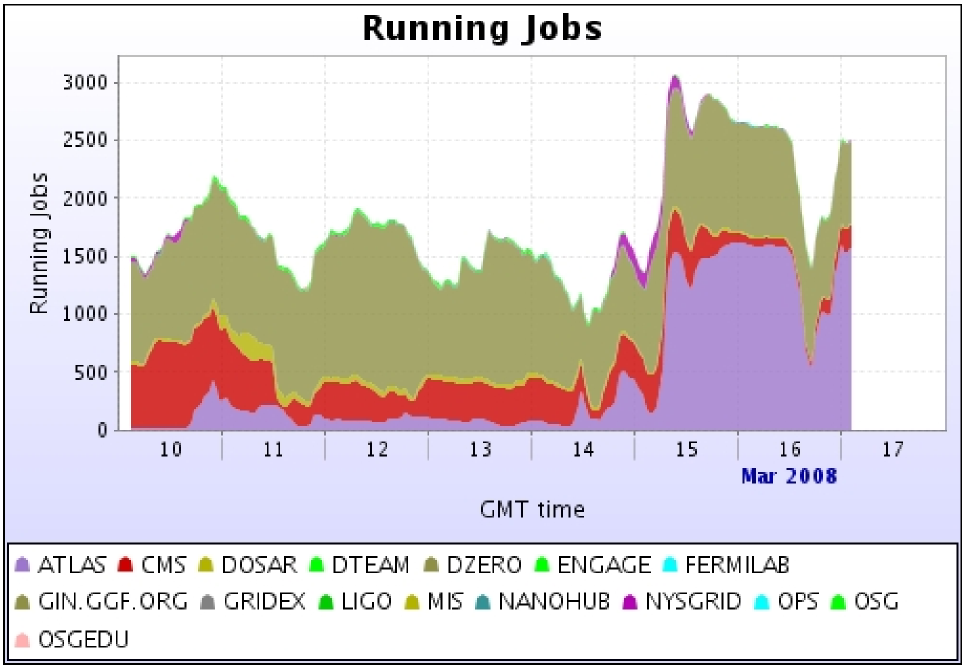


Figure CS-2. Current running applications and load on the Open Science Grid. Plot provided by MonALISA [24].

## CMS: The Compact Muon Solenoid

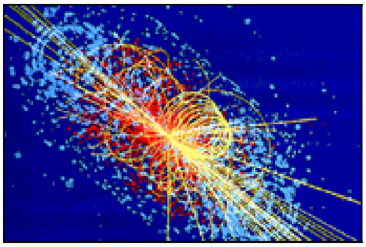


Figure CS-3. Simulated decay of Higgs boson in the future CMS experiment at CERN. (Credit: CERN)

### Collaborators, Organizations

The USCMS Collaboration consists of various US universities and Fermi National Accelerator Laboratory (FNAL). The Collaboration works closely with the CMS Collaboration at CERN to accomplish the missions of the experiment. Major funding of this program is provided by The US Department of Energy (DOE) and the National Science Foundation (NSF).

See US CMS Institutions and Members [25] for details.

### Summary/Description

From the U.S. CMS website [26]:

> "The CMS experiment is designed to study the collisions of protons at a center of mass energy of 14 TeV. The physics program includes the study of electroweak symmetry breaking, investigating the properties of the top quark, a search for new heavy gauge bosons, probing quark and lepton substructure, looking for supersymmetry and exploring other new

phenomena."

The USCMS Software and Computing [27] project provides the computing and software resources needed to enable US scientists to participate in CMS activities.

According to the CERN Architectural Blueprint RTAG [28] (October, 2002) the configuration and control of Grid-based operation should be encapsulated in components and services intended for these purposes. Apart from these components and services, grid-based operation should be largely transparent to other components and services, application software, and users. Grid middleware constitutes optional libraries at the foundation level of the software structure. Services at the basic framework level encapsulate and employ middleware to offer distributed capability to service users while insulating them from the underlying middleware. For the USCMS, the OSG provides the necessary Grid middleware components (that are also made to be interoperable with the LCG/EGEE components.)

**Data and Control Flow**

The CMS experiment employs a tiered computing model. Tier0 is at CERN in Switzerland. FNAL is one of seven Tier1's and universities in the US and Brazil are the Tier2's. Experimental data is produced at the Tier0 and replicated at Tier1's. Tier2's have the responsibility of hosting data that is interesting for regional users and will be used for data analysis by users through OSG gatekeepers at those Tier2's. Monte Carlo simulated events (MC events) are produced at Tier2's and Tier1's. These MC events are transferred to region Tier1's (FNAL in case of USCMS) or the Tier0. Thus, the model for the CMS experiment calls for data to be passed by the CMS detector at CERN in Switzerland, to a series of large computing sites around the world (and MC events the opposite direction.)

The CMS Tier-2 centers in the United States and around the world have more work yet to do on their network infrastructure before they're ready to accept the large data rates expected when the experiment starts running — up to 100 megabytes per second. The eventual goal for the computing sites during 2007 is to sustain the use of more than 50% of their network capacity for an entire day. For example, for the Purdue-UCSD network link that would mean sustaining transfers at approximately four gigabits per second for one day [29].

Data storage responsibilities are shared between OSG, the VO, and the site. For example, OSG defines storage types and the API's and the information schema for finding storage. The VO manages the data transfers and the catalogues. The site chooses the storage type and amount, and implements publication of storage information according to the OSG rules (more specifically the Glue schema.) The following image is an example of CMS data transfer across several days in early 2007.

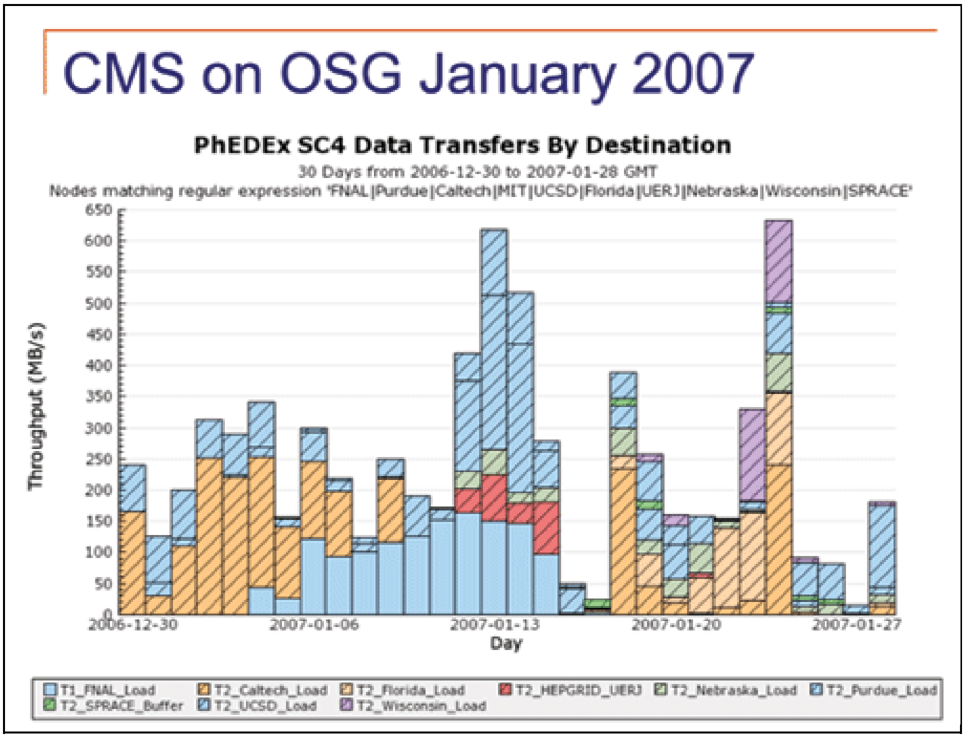


Figure CS-4. CMS data transfer at OSG sites. [30]

Likewise, job submission responsibilities are shared by OSG, the VO, and site. OSG defines the interface to the batch system and information schema and provides the middleware that implements them. The VO manages the job submissions and workflows. (This is through either the Condor-G job submission tools or the workload management systems developed by grid projects such as EGEE/LCC.) The site chooses which batch system to use but configures that system interface in accordance with OSG rules.

The workflow can be described as:

- The VO administrators, called the software deployment team, install the application software. Users have read-only access from batch slots.
- Data is produced at CERN. MC events are produced by the MC production teams at OSG or EGEE/LCG sites.
- Data movement is carried out by a system called the PhEDEx. CERN controls the rate of data movement and sites or authorized personnel subscribe to necessary data through the PhEDEx system. The VO administrator moves MC events produced at the site to the upper Tiers via gftp. Users have read-only access from batch slots.
- Users submit their jobs via condor-g. The jobs run in batch slots, writing output to local disks. The jobs copy their output from the local disks to the data area via gftp.
- Users collect their output from the site(s) via gftp for follow-up analysis.

**Contact**

US CMS Organization, Institution, and Member Contacts [31]

## SDSS: Sloan Digital Sky Survey

Figure CS-5. SDSS Image of the Week
(click for this week's image.)

### Collaborators, Organizations

The SDSS collaboration includes 150 scientists at 25 institutions [32]. An advisory council [33] represents the institutions and advises the ARC Board of Governors on matters relating to the projects.

### Summary/Description

The Sloan Digital Sky Survey (SDSS) is focused on producing a detailed optical image and 3-dimensional map covering a significant portion of the sky. With the amount of data that must be stored and managed, and the compute power required to produce the rich, integrated visual results, the project is a clear example of a scientific milestone that is dependent on advancements in distributed, collaborative high performance computing.

From the SDSS website: [34]

> The SDSS uses a dedicated, 2.5-meter telescope on Apache Point, NM, equipped with two powerful special-purpose instruments. The 120-megapixel camera can image 1.5 square degrees of sky at a time, about eight times the area of the full moon. A pair of spectrographs fed by optical fibers can measure spectra of (and hence distances to) more than 600 galaxies and quasars in a single observation. A custom-designed set of software pipelines keeps pace with the enormous data flow from the telescope.
>
> The SDSS completed its first phase of operations "SDSS-I" in June, 2005. Over the course of five years, SDSS-I imaged more than 8,000 square degrees of the sky in five bandpasses, detecting nearly 200 million celestial objects, and it measured spectra of more than 675,000 galaxies, 90,000 quasars, and 185,000 stars. These data have supported studies ranging from asteroids and nearby stars to the large scale structure of the Universe.
>
> The SDSS has entered a new phase, SDSS-II, continuing through June, 2008. With a consortium that now includes 25 institutions around the globe, SDSS-II will carry out three distinct surveys the Sloan Legacy Survey, SEGUE, and the Sloan Supernova Survey to address fundamental questions about the nature of the Universe, the origin of galaxies and quasars, and the formation and evolution of our own Galaxy, the Milky Way."

For more background information on mapping universe and new discoveries, see About US [35] at the SDSS web site.

### Contact

The SDSS business manager and institutional representatives are listed on the SDSS Contact US [36] web page.

### Acknowledgements

Funding for the SDSS and SDSS-II has been provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Science Foundation, the U.S. Department of Energy, the National Aeronautics and Space Administration, the Japanese Monbukagakusho, the Max Planck Society, and the Higher Education Funding Council for England.

## ATLAS

Figure CS-6. The ATLAS Detector
(click for more images.)

### Collaborators, Organizations

The ATLAS collaboration consists of various boards, institutions, committees, and working groups. Over 1,850 individuals at roughly 175 institutions across 37 countries work together. See The ATLAS Organization [36] for more details. A very interesting discussion on how the collaboration works can be found at How ATLAS Collaborates [37].

### Summary/Description

One of the discoveries eagerly anticipated by particle physicists working on the world's next particle collider is that of supersymmetry, a predicted lost symmetry of nature. Physicists from the University of Wisconsin-Madison are using Open Science Grid resources to show that there is a good possibility of discovering supersymmetry with data collected during the first few months of the collider's operation, if the new symmetry exists in nature.

Supersymmetry, often called SUSY, predicts the existence of superpartner particles for every known particle, or sparticles, for every known fundamental particle.. Recent experiments have suggested that most of the matter in our universe is not made of familiar atoms, but of some new sort of dark matter. Discovering a hidden world of sparticles may shed light on the nature of this dark matter, connecting observations performed at earth-based accelerators with those performed by astrophysicists and cosmologists.

### Data and Control Flow

To accurately simulate the search for supersymmetry required physicists to create a gateway to three different grid environments from their desks at CERN. They used the Virtual Data Toolkit, an ensemble of middleware tools distributed and maintained with the collaboration of OSG members, to create an access point to resources from the Open Science Grid, the LHC Computing Grid and the University of Wisconsin-Madison's Condor pool.

"The most difficult part was to make a grid which is interoperable, such that the requirements of all existing grid flavors could be included," they explained. "This was done by modifying the current VDT, and consuming more than 215 CPU years in less than two months using resources from the OSG and Madison's Condor Pool."

With so many computing resources at their disposal, they simulated for the first time an accurate background for SUSY searches. Comparing the simulated signals for several types of SUSY against the simulated background shows that physicists might be able to discover the long-sought sparticles with the first ATLAS experimental data.

See Simulating Supersymmetry with ATLAS [38] for the complete article.

### Contact

See the ATLAS Experiment home page [39].

### Acknowledgements

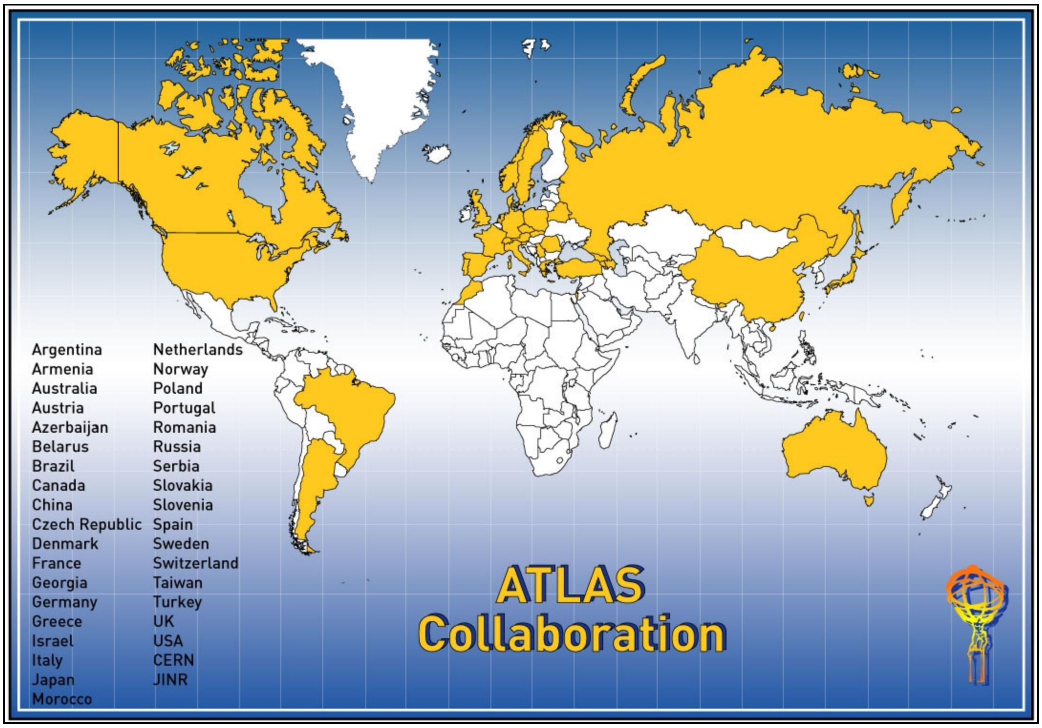


Figure CS-7. ATLAS Collaboration Map.

## SURAgrid Applications

### Simulation-Optimization for Threat Management in Urban Water Systems

### Collaborators

Sarat Sreepathi and Mahinthakumr, NCSU
Von Laszewski and Haetgen, University of Chicago
Uber and Feng, University of Cincinnati
Harrison, University of South Carolina

### Summary/Description

Contamination threat management is a very real and practical concern for any population utilizing a shared drinking water distribution system. Several components are involved including real-time characterization of the source and extent of the contamination, identification of control strategies, and design of incremental data sampling schedules. This requires dynamic integration of time-varying measurements of flow, pressure and contaminant concentration with analytical modules including models to simulate the state of the system, statistical methods for adaptive sampling, and optimization methods to search for efficient control strategies. The goal of this multi-disciplinary research project (NSF-funded from Jan 2006 to Dec 2008) is to develop a cyberinfrastructure system that will both adapt to and control changing needs in data, models, computer resources and management choices facilitated by a dynamic workflow design.

The application specifically incorporates dynamic water-usage data, in real-time, into a simulation-optimization process to inform decision making in threat management situations. The nature of this work is highly compute-intensive and requires multi-level parallel processing via computer clusters and high-performance computing architectures such as SURAgrid. The optimization component uses evolutionary computation based algorithms and the simulation component uses EPANET, a water distribution simulation code originally released by USEPA. Simulation-Optimization with EPANET is part of a multidisciplinary, three-year NSF-funded DDDAS (Dynamic Data-Driven Application Systems) research project to develop a cyberinfrastructure system that will both adapt to and control changing needs in data, models, computer resources and management choices facilitated by a dynamic workflow design. Project Partners: North Carolina State University; University of Chicago; University of Cincinnati University of South Carolina

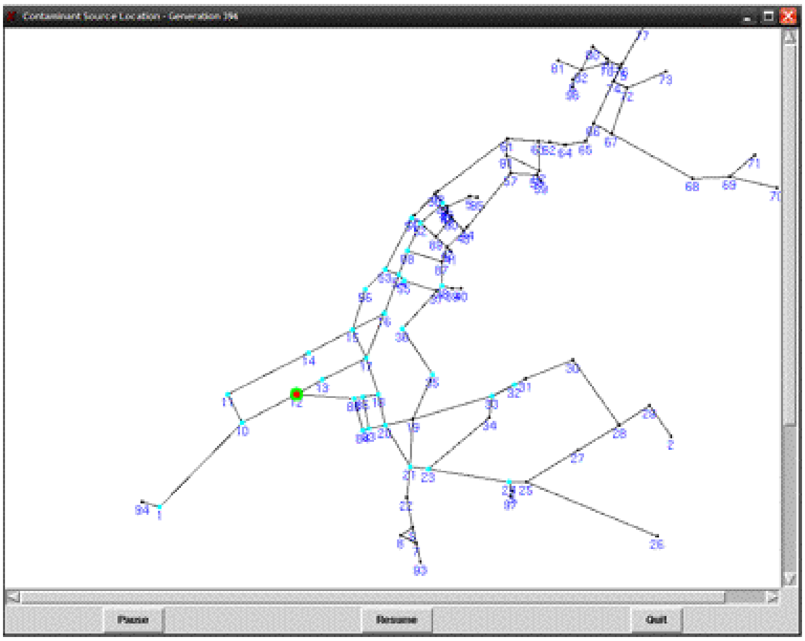


Figure CS-8. Graphical Monitoring Interface

The analytical modules (composed of thousands to millions of simulation instances that are driven by optimization search algorithms) used to simulate realistic water distribution systems are highly compute-intensive and require multi-level parallel processing via computer clusters. While data often drive the analytical modules, data needs for improving the accuracy and certainty of the solutions generated by these modules dynamically change when a contamination event unfolds. Since such time-sensitive threat events require real-time responses, the computational needs must also be adaptively matched with available resources. Grid environments composed of independent or loosely coupled computer clusters (e.g., the TeraGrid, SURAgrid) are ideal for this application as the simulation instances can be easily clustered (or bundled) into semi-independent sets, often requiring synchronization at various stages, that can be effectively executed in these environments through an intelligent allocation and monitoring mechanism which is currently being implemented as a middleware feature.

**SURAgrid Deployment**

The integrated simulation-optimization system developed through this project is intended to be used by the project team members during the two-year development phase of this project. Team members include application engineers at North Carolina Statue University (NCSU) and the University of Cincinnati, optimization methodology developers (NCSU and the University of South Carolina), and computer scientists (NCSU and the University of Chicago). The application engineers will test and analyze various water distribution contamination problem scenarios using realistic networks. The methodology developers will investigate various optimization search algorithms for source characterization, demand uncertainty and sensor sampling design.

The computer scientists will undertake the grid implementation, integration of various components, and performance testing in different grid environments and computer clusters, including SURAgrid. The team is using SURAgrid as an “on-ramp” to the TeraGrid. Citing specific SURAgrid benefits such as compute resource heterogeneity and low overhead to participate, the team plans to ready the application for porting to the TeraGrid by uncovering and addressing potential programming and workflow issues on SURAgrid.

## Grid Workflow

To be able to run jobs on SURAgrid, the NCSU user applies for an affiliate user certificate issued by SURAgrid site Georgia State University (GSU), who has a Certificate Authority (CA) that has been cross-certified with the SURAgrid Bridge CA (BCA). Cross-certification enables SURAgrid resource sites to trust the user certificate being presented by the NCSU user and, when the SURAgrid User Administrator at GSU also creates a SURAgrid account for the NCSU user, the user essentially has single-sign-on access to SURAgrid resources at cross-certified SURAgrid sites1. After they’ve authenticated to the SURAgrid resource, the user invokes the optimization method on the client workstation that initiates the middleware that directly communicates with the specific SURAgrid resource (authenticated through ssh keys) for job submission and intermediate file movement. Currently the application needs to be pre-staged by the user, but this functionality will be integrated into the middleware. The middleware, which uses public key cryptography, will provide a seamless, python-based application interface for staging initial data and executables, data movement, job submission, and real-time visualizations of application progress. The interface uses passwordless ssh commands to create the directory structure necessary to run the jobs and handles all data movement required by the application. It launches the jobs at each site in a seamless manner, through their respective batch commands. The middleware is able to minimize resource queue time by querying the resource at a given site to determine the size of resource to request. Most of the middleware functionality has been implemented at least at a rudimentary level and efforts are now focused on better integration and sophistication.

In addition to the middleware interface described above, the application consists of two major components: one for optimization, one for simulation. The optimization component presently used on the SURAgrid is called JEC (Java Evolutionary Computation toolkit), This is the client side that drives the simulation component by calling the middleware interface. Evolutionary algorithms call multiple instances of simulations (typically hundreds) at each generation (or iteration) and require synchronization at each generation as the simulation results have to be processed before beginning the next generation. Everything on the server side (middleware, simulation component, and the grid resources) is transparent to the client.

The simulation component is an MPI C wrapper written around EPANET that does a number of things. It bundles multiple simulations (typically hundreds) and performs simultaneous execution of these on a single cluster via a coarse-grained MPI-based parallelism feature. The wrapper saves a considerable amount of processing time by not duplicating I/O and parts of simulations that are common to all simulation instances. It also has a persistent capability such that, once an EPANET job is launched, it does not need to exit until all simulation instances have been completed across all generations of an evolutionary algorithm (i.e., once the simulation outputs are written for a given generation, it can maintain a wait state until the next set of evaluations arrives from the middleware). The output files are moved back to the client workstation as the simulation progresses on the resource side. A python/TK real-time visualization tool developed by NCSU then enables visualization of the progress of the algorithm on the water distribution network. The visualization tool also creates PNG files of various stages of the output.

## Acknowledgements

Simulation-Optimization with EPANET is part of a multidisciplinary, three-year NSF-funded DDDAS (Dynamic Data-Driven Application Systems) research project to develop a cyberinfrastructure system that will both adapt to and control changing needs in data, models, computer resources and management choices facilitated by a dynamic workflow design.

## Multiple Genome Alignment on the Grid

### Collaborators

Georgia State University
SURA

### Summary/Description

This application takes a number of genome sequences as input and gives an aligned sequence based on their structure by using a pairwise alignment algorithm. When run on grids like SURAgrid, carefully designed and grid-enabled algorithms like this, which implement a memory efficient method for computation and are also parallelized efficiently so that the workload is well distributed on grids, afford bioinformatics users a performance comparable to cluster environments while giving them added flexibility and scalability.

Biological sequence alignment is used to determine the nature of the biological relationship among organisms, for example, in finding evolutionary information, determining the causes and cures of diseases, and for gathering information about a new protein. Multiple genome sequence alignment (where several genome sequences are aligned rather than only two) is very important for analysis of genome and protein structures — particularly for showing relationships among structures being aligned. A significant challenge to researchers is the computational requirements to align multiple (more than three) sequences of very large size. With Georgia State University's (GSU) core research initiatives in life sciences, and particularly protein structure analysis, Dr. Yi Pan, currently GSU Chair Computer Science, and Nova Ahmed, as his graduate student, provided a significant contribution in this area by deploying a parallelized multiple sequence alignment algorithm application in a grid environment, thus improving computer processing of the large sequence lengths typical of genomic and proteomic science.

### SURAgrid Deployment

Although the parallel algorithm requires inter-processor communication to compute multiple aligned sequences, it actually reduces overall computation by independently solving and then merging a set of tasks. The new algorithm, which was initially designed for a shared memory architecture where it is helpful to reduce the memory requirement, did indeed improve performance during its initial runs. However, the resulting algorithm and its parallelization is also suited to grid environments such as SURAgrid that benefit this type of distributed, computationally intensive work. Ahmed's tests of grid-enabled clusters showed comparable performance to that of non-grid-enabled clusters (there was negligible overhead from the grid layer services) and a significant improvement over older shared memory-type systems. Pan and Ahmed's algorithm can provide very scalable, cost-effective computational performance for grid environments, where job submission and scheduling can be easier since users don't need account on every node and can submit multiple jobs at one time.

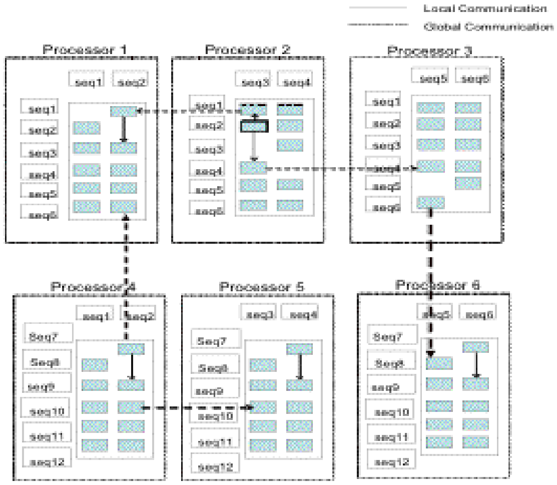


Figure CS-9. Parallel load distribution among processors for multiple sequence alignment

There were several iterations of testing for both the code and Georgia State and SURAgrid's access management infrastructure components. The end result of the collaboration is that Georgia State users run the multiple genome alignment application through the integration of their personal identity verification into Georgia State's campus identity management environment, which is then leveraged to provide external access to all SURAgrid resources.

To create a local grid certificate, the user sends a request from their official campus email and is issued a grid certificate based on their unique CampusID. The ACS Certificate Authority (CA) that ACS created and cross-certified with the SURAgrid Bridge CA (BCA), provides the local user's passport to SURAgrid resources. The cross-certification process enables a SURAgrid resource to trust the Georgia State local certificate being presented by the user. The user experience is further simplified by Georgia State's use of the SURAgrid user account system that essentially provides single-sign-on access to SURAgrid resources at cross-certified SURAgrid sites. The account management system overlays the cross-certification process and empowers the SURAgrid User Administrator from Georgia State to easily issue SURAgrid user accounts. The user's Georgia State issued certificate invokes the Globus Toolkit that allows Globus, on behalf of the algorithm application, to manage the grid services necessary to submit the application's jobs to various SURAgrid resources.

### Conclusion

As Georgia State continues to deploy grid technology, policies and processes of their campus grid, they expect the multiple genome algorithm alignment code will continue to be used to test and perfect the grid. Considering that it also provides a memory efficient, pair-wise alignment for large biological sequences in an optimal way, the application is an invaluable asset to Georgia State and to others interested in improved sequence alignment using SURAgrid resources.

### Acknowledgements

Nova Ahmed, Ph.D. student CS, Georgia Tech
Victor Bolet, Analyst Programmer Intermediate, Advanced Campus Services Georgia State
Dharam Damani, MS student CS, Georgia State University
Nicole Geiger, Analyst Programmer Associate, Advanced Campus Services Georgia State
Yi Pan, Professor, Chair Computer Science, Georgia State

# Grid Deployments

## Texas Tech TechGrid

### Collaborators

Texas Tech University

### Summary/Description

The Texas Tech grid project, TechGrid, mission is to integrate the numerous and diverse computational, visualization, storage, data, and spare lab desktop resources of Texas Tech University into a comprehensive campus cyber infrastructure for research and education. The integration of these vast resources into TechGrid will enable resource access and sharing on an unprecedented scale, while new Web-based and command-line interfaces will facilitate new models for utilization and coordination. The goals of rapid deployment, adoption, and evolution of TechGrid will enable it to serve as a research and teaching computing infrastructure, while also providing a platform for grid computing R&D. TechGrid will thus present a unique campus environment for knowledge discovery and education.

### About TechGrid

Texas Tech University grid, TechGrid, developed and deployed in 2002, is a comprehensive cyber infrastructure project to bring a distributed-knowledge environment to Texas Tech research and education. TechGrid consists of 600 Windows and Linux PC's donated from various parts of campus to share spare computational cycles while the donated resources are not being used. The grid software used to integrate these compute resources together is called Condor. Condor is a grid middleware package developed by the University of Wisconsin. During the past five years, TechGrid has helped facilitate the massive computing needs of research projects involving computational chemistry, bioinformatics, biology, physics, mathematics, engineering, and business statistical analysis. Additionally, TechGrid has been instrumental in teaching distributed and grid computing in the Texas Tech Advanced Technology Learning Center, Texas Tech Teaching Learning and Technology Center, Texas Tech Jerry Rawls School of Business, Texas Tech Computer Science department as well as the Texas Tech Mathematics and Statistics department.

The goal of the TechGrid project is to enable significant advances in scientific discovery and to foster innovative educational programs. TechGrid will integrate and simplify the usage of the diverse computational, storage, visualization, and some data resources of Texas Tech to facilitate new, powerful paradigms for research and education. The project will serve as a model for other campuses wishing to develop an integrated cyber infrastructure for research and education.

### Middleware

The grid distributes a compute job among compute nodes within the grid using grid middleware as the means to facilitate distributed computing.  The name of the grid middleware is Condor.

*What is Condor?*

From the University of Wisconsin Condor site [68]:

> Condor is a specialized workload management system for compute-intensive jobs. Like other full-featured batch systems, Condor provides a job queuing mechanism, scheduling policy, priority scheme, resource monitoring, and resource management. Users submit their serial or parallel jobs to Condor, Condor places them into a queue, chooses when and where to run the

jobs based upon a policy, carefully monitors their progress, and ultimately informs the user upon completion.

While providing functionality similar to that of a more traditional batch queuing system, Condor's novel architecture allows it to succeed in areas where traditional scheduling systems fail. Condor can be used to manage a cluster of dedicated compute nodes (such as a "Beowulf" cluster). In addition, unique mechanisms enable Condor to effectively harness wasted CPU power from otherwise idle desktop workstations. For instance, Condor can be configured to only use desktop machines where the keyboard and mouse are idle. Should Condor detect that a machine is no longer available (such as a key press detected), in many circumstances Condor is able to transparently produce a checkpoint and migrate a job to a different machine which would otherwise be idle. Condor does not require a shared file system across machines — if no shared file system is available, Condor can transfer the job's data files on behalf of the user, or Condor may be able to transparently redirect all the job's I/O requests back to the submit machine. As a result, Condor can be used to seamlessly combine all of an organization's computational power into one resource.

**Definitions, Components, and Software tools**

*Definitions*

1. Grid Zone: is a department or lab associated with a campus department that has volunteered resources to be used by the grid.

2. Grid Zone Administrator: a person who is responsible for the grid zone in their individual departments.

3.Campus Grid Administrator: a person who is responsible for the maintenance, upkeep, and operation of the grid, HPCC grid research, grid training, and interfacing with the general computing user base to supply grid based and High Performance Computing support and services to the Texas Tech campus community.

4. Grid Node: is an individual computer within a Grid Zone that contributes compute cycles to the grid.

5. Grid Attribute: individual settings such as permissions, performance, or scheduling mechanism that can be controlled by the Grid Administrator.

6. Bootstrap Server: is the central grid server responsible for controlling grid functions and job management.

*Components*

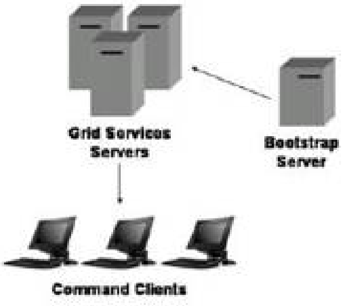


Figure CS-9. Job distribution on TechGrid.

**Applications**

Applications on the TechGrid include:

The Proth [40] code was provided by Dr. Chris Monico and grid-enabled to run on TechGrid. The code used several thousand CPU hours to look for prime numbers from sieved candidates.

The Partial Differential Equation [41] grid project of Dr. Sandro Manservisi was grid-enabled and used 1200 CPU hours.

The grid-enabled *Multivariate Minimization project* was completed and published at Global Grid Forum 8 . Title: Multivariate Minimization Using Grid Computing by K. Kulish, J. Perez, P. Smith. [42]

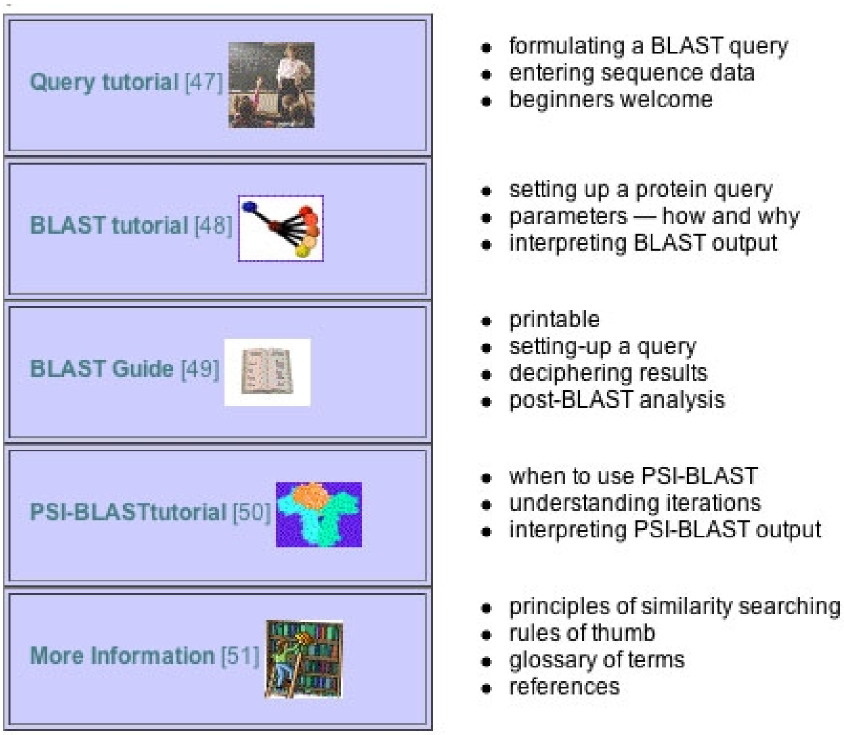
A Matlab executable was grid-enabled to simulate the lifespan of catfish for a PhD Thesis by Dr. Eric Albers [43].

Installation of and experimention with SRB (Storage Resource Broker) data grid [44] was completed.

The San Diego Supercomputing Center's supercomputing library of space movies were accessed.

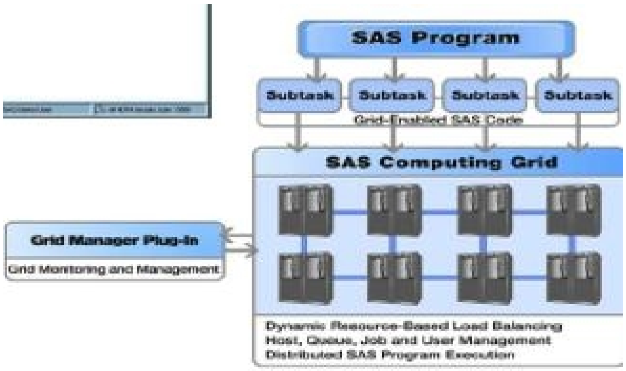
In ccooperated with the Architecture department, a 3-D Studio Max graphics rendering grid was created [45]. Denny Mingus and Glenn Hill were the main contacts.

In collaborattion with the Biology department, a grid-based BLAST [46] was explored.  Basic grid BLAST jobs were possible; however a means to move data was still required to handle large BLAST datasets.  Dr. Natalya Klueva and Dr. Randy Allen were the contacts for this project.



| | |
|---|---|
| Query tutorial [47] | formulating a BLAST query<br>entering sequence data<br>beginners welcome |
| BLAST tutorial [48] | setting up a protein query<br>parameters — how and why<br>interpreting BLAST output |
| BLAST Guide [49] | printable<br>setting-up a query<br>deciphering results<br>post-BLAST analysis |
| PSI-BLASTtutorial [50] | when to use PSI-BLAST<br>understanding iterations<br>interpreting PSI-BLAST output |
| More Information [51] | principles of similarity searching<br>rules of thumb<br>glossary of terms<br>references |

In collaboration with the Rawls College of Business a SAS-based compute grid [52] was created. The grid was designed and deployed in a 3 week period. Dr. Peter Westfall is the major contact for this project.

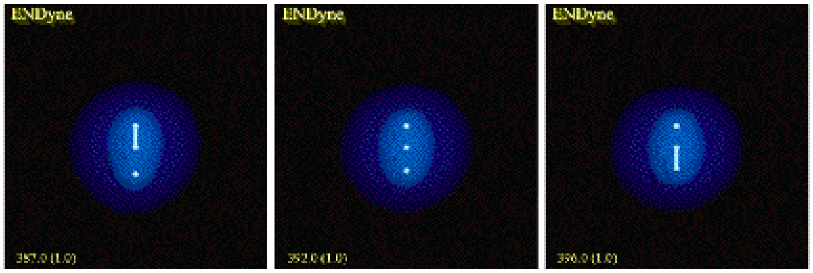

SAS Program
Subtask
Subtask
Subtask
Subtask
Grid-Enabled SAS Code
SAS Computing Grid
Grid Manager Plug-In
Grid Monitoring and Management
Dynamic Resource-Based Load Balancing
Host, Queue, Job and User Management
Distributed SAS Program Execution

A physics space simulation "Neighbors" for a physics graduate thesis [53] was grid-enabled. The purpose was to simulate the effects of tumbling debris on a spacecraft upon reentry into the Earth's atmosphere. Several thousand simulations were processed.

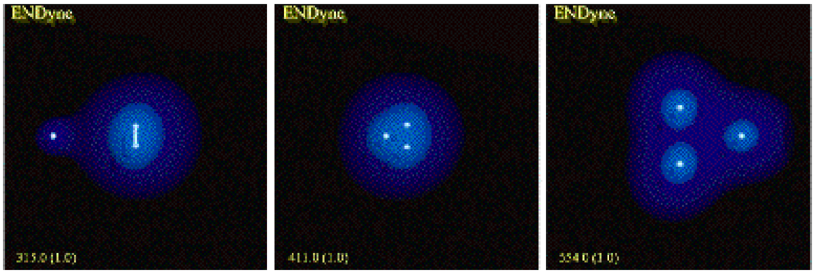
Texas Tech HPCC and the University of Virginia joined Data Grid to test the Internet2 connectivity between universities. Results were published in the ACM Journal of Computing.

In the USDA Grid Bioinformatics Project [54] TechGrid helped Dr. Scot Dowd with the Administration of Blast jobs to analyze the pig genome using TechGrid and Rocks clustering. This was a collaborative effort between Texas Tech and the USDA.

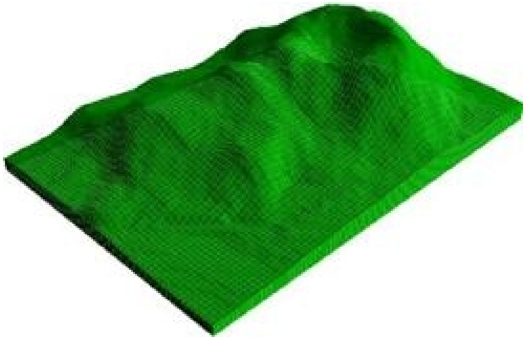
ENDYNE is a grid implementation of the electron nuclear dynamics theory: a coherent-states chemistry. ENDYNE is a TTU grid project that involves TTU computational chemists and TTU HPCC staff developing a grid-based method of calculating a coherent-states simulation that uses classical theoretical models and quantum mechanics to simulate the relationships between chemical atomic interactions.



Snapshots of a head-on collision of a proton and a hydrogen molecule at three different times.



Snapshots of a collision of a proton splitting the bond of hydrogen molecule at three different points of the trajectory.



3-D plot from R analysis

Researchers use the "R" programming language/framework [55] to process R macros on the grid to calculate mathematical models as well as genomic bioinformatics data.

**TechGrid Status**

TechGrid's compute nodes are located in the Advanced Technology Learning Center (ATLC), the High Performance Computing Center (HPCC) at Reese Center, the Computer Science department, the Business Building, the North Computing Center, and the Math Building.  Currently, TechGrid is made up of 600+ compute nodes spanning several domains and three operating systems.

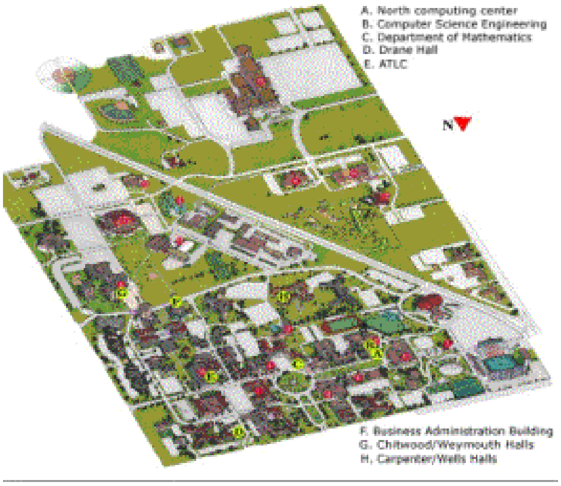

A. North computing center
B. Computer Science Engineering
C. Department of Mathematics
D. Drane Hall
E. ATLC
N
F. Business Administration Building
G. Chitwood/Weymouth Halls
H. Carpenter/Wells Halls

Figure CS-10. The campus-wide grid is distributed across the TTU campus.

### Contact

Jerry Perez, Texas Tech University.

URL: http://www.hpcc.ttu.edu/techgrid.html [56]

## White Rose Grid

### Collaborators, Organizations

The White Rose Consortium in Yorkshire, England: The universities of Leeds, Sheffield, and York

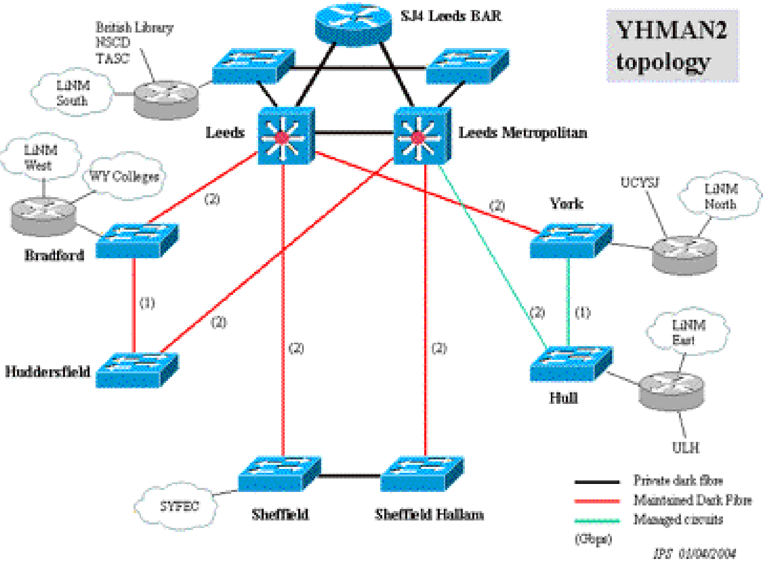


Figure CS-11. The White Rose Grid.

### Summary/Description

The White Rose Grid (WRG) e-Science Centre brings together those researchers from the Yorkshire region who are engaged in e-Science activities and through these in the development of Grid technology. The initiative focuses on building, expanding and exploiting the emerging IT infrastructure, the Grid, which employs many components to create a collaborative environment for research computing in the region.

The White Rose Grid (WRG) at Leeds also hosts one of the four core nodes of the National Grid Service (NGS), which offers a production quality grid service for use by UK academia. (The other nodes are at CCLRC-RAL, Oxford, and Manchester.)

**Components and Software/Toolkits**

The White Rose Grid comprises five large compute nodes of which three are located at the University of Leeds, one at the University of Sheffield and one at the University of York. It offers a heterogeneous computing environment based on Sun Microsystems [57] multiprocessor computers, and Intel Xeon and AMD Opteron based systems built by Streamline Computing [58]. These nodes are interconnected by the network managed by YHMAN.

- The Leeds Grid Node 1 is a constellation of shared-memory systems based on Sun Fire 6800 and V880 systems configured with UltraSPARC III Cu 900MHz processors and large physical memory (32GB).
- The Leeds Grid Node 2 comprises two Linux clusters based on 2.2 & 2.4 GHz Intel Xeon processors interconnected with Myrinet 2000 networks, and in total delivering 292 CPUs.
- The Leeds Grid Node 3 comprises Sun Microsystems? Sun Fire V40z and V20z servers with dual-core AMD Opteron processors supplied by Esteem Systems and integrated by Streamline Computing. Seven of these (V40z) comprise four 2.2 GHz dual-core processors configured with 192 GB memory. Eighty seven V20z servers are interconnected with a Myrinet network; each of these comprises two 2.0 GHz dual-core processors sharing in total 0.7 TB of distributed memory across 348 processor cores. The system runs the Linux (64-bit SuSE) operating system.
- The Leeds Nodes are connected to 12 TB SAN storage and two EMC Centera disk-based archiving systems set up to provide 12TB of archive space to users. Sun HPC ClusterTools, Sun Forte Developer software and Sun Grid Engine Enterprise Edition are installed on all systems.
- The 160 processor WRG Sheffield node has been supplied by Sun Microsystems and integrated by Streamline Computing. Eighty of these 2.4GHz AMD Opteron processors are 4-way nodes with 16GB main memory coupled by a Myrinet network; the remaining eighty nodes are 2-way nodes with 4GB main memory.
- At Sheffieled there is also a Tier-2 GridPP node supporting the particle physics grid. This system is configured with 160 processors in 2-way nodes, and it runs 64-bit Scientific Linux, which is Redhat based.
- The York Node includes two Beowulf type clusters, one (24 machine cluster; each providing two 2.4GHz dual core processors and 8 GB memory) in total offering 96 processor cores, 192 GB memory and 4.8 TB local scratch space; and the other which comprises 3 large memory nodes, each consisting of four 2.4 GHz dual core processors (8 cores per machine) and 8GB memory, in total delivering 24 processor cores configured with 96GB memory and 0.9 local scratch space. All these nodes are connected into a 10GB/s infinipath network for fast file access. In addition the cluster nodes are able to use this network for very low latency <2m MPI applications. Over 9TB of backed up storage is provided for users on SATA drive arrays and a 1 TB networked scratch space on f/c arrays.

WRG systems support applications written in FORTRAN, C, and C++, implementing parallelism through MPI or OpenMP. A couple of the Sun Fire V880s serve the open source Grid Portal, which interoperates with Globus middleware and Sun Grid Engine Enterprise Edition.

Furthermore, at the University of Leeds there is also the Virtual Environments Laboratory which comprises a T.A.N. 3D Holobench, SGI Onyx2 with interactive devices and projectors. Also a recently acquired visualisation node is available at Leeds for WRG researchers.

See the White Rose Grid Compute Node [59] description for more information.

## Applications

The following applications include current and past projects. See the White Rose Activities [60] page for more projects and more information on each project.

CARMEN is a 4-year EPSRC funded e-Science Pilot Project involving 11 Universities and 19 Investigators. It aims to use grid technologies to enable experimenters in neurophysiology to archive their datasets in a structure, making them widely accessible for computational modelers and algorithm developers to exploit. The project will provide integrated and coordinated services for the neuroscience data, enabling neuronal signal detection, sorting and analysis, as well as visualisation and modeling. Furthermore it will enable direct near real-time analysis of streamed experimental data, providing information to distributed teams of specialists that will allow difficult experiments to be optimised.

COLAB is a joint research project of the Universities of Leeds (UK) and Beihang in Beijing (China) co-led by Profs J Xu (Leeds) and J Huai (Beihang), and managed by the EPSRC White Rose Grid e  Science Centre established between Universities of Leeds, York and Sheffield. The project relates to the CROWN (China Research environment Over Wide-area Network) grid middleware system originally developed at Beihang University. Two sub-groups research the areas of Fault and Attack Tolerance, and Fault Injection-based Evaluation. Amongst other topics they investigate the provision of topologically aware fault and intrusion tolerance in grid systems as well as the provision of revised fault models for grid applications.

Grid-FIT (Grid-Fault Injection Technology) is a fault injector that utilizes network level fault injection to assess grid systems. Grid-FIT has been implemented specifically to test SOAP based web services systems and Globus systems.

Integrative Biology addresses two key problems in medicine today: the causes of cardiac failure and cancer tumours. Scientists are developing multi-scale models (from cells to whole organs) to help understand these problems. The size and complexity of the models demands significant compute power, and so this project brings together scientists and Grid computing experts. The project is being led by the University of Oxford and involves partners across the world, including the USA and New Zealand. Our contribution is in the area of computational steering and visualization, and is led by Professor Ken Brodlie and Dr James Handley.

The MoSeS (Modeling and Simulation for e-social Science) project is undertaken by the National Centre for e-Social Science node at the University of Leeds. The objective of this project is to develop representation of the entire UK population as individuals and households, together with a package of modeling tools which allows specific research and policy questions to be addressed.

The Scientific e-Communities Architecture (SeCA) project focuses on the design and evaluation of a novel Collaborative e-Science Architecture and its application, in the first instance to combustion chemistry. The project exploits Peer-to-Peer (P2P) technologies for supporting this scientific community model and a grid-based workgroup architecture for providing access to large computation and data resources. There are a number of challenges in realising the vision, for example, effective P2P resource discovery.

DAME (Distributed Aircraft Maintenance Environment), led by Prof Austin of York, was a major (£3.5m) e-Science project, which has developed a generic test-bed for distributed diagnostics. The application demonstrator built within the project offers a distributed maintenance environment motivated by the needs of Rolls Royce and its information system partner, Data Systems and Solutions.

The e-Demand project was supported by the Leeds and Durham Grid consortium, which includes experts from both academia and industry. The project has developed a demand-led and service-centric architecture for building complex but dependable and secure Grid applications based on the notion of ultra-late binding, dynamically bound service components, combined with atomic actions as a powerful control abstraction.

GEMSS (Grid-enabled Medical Simulation Services) is funded by the EU FP5 programme and is concerned with creating an environment in which computationally demanding tools native to the Health-Care sector can be made available to a wide spectrum of users. The goal is to provide a transparently accessible health computing resource suited to solving problems of large magnitude, with the end user having no awareness of the Grid computing platform(s). The project will evaluate the viability of this approach through several sample applications, including maxillo-facial surgery planning, neuro-surgery support, medical image reconstruction, radiosurgery planning and lung/cardiovascular simulations — the latter two have their base in Sheffield (Medical Physics)

GOSPEL, led by Professor M Berzins of Leeds University, and carried out in collaboration with Shell Research, has brought together advanced visualization, problem-solving environments, and computational techniques to create a Grid based workbench for the computational modeling of lubricants.

This ESRC demonstrator and the follow-on HYDRA2 project**,** both led by Dr M Birkin and Prof P M Dew from the University of Leeds, have demonstrated the use of grid technologies in support of the decision-making process in health care planning. A disparate set of data sources as well as a decision support module and visualization have been integrated to present the results.

myGrid will design, develop and demonstrate higher level functionalities over an existing Grid infrastructure that support scientists in making use of complex distributed resources. The project will develop a virtual laboratory workbench that will serve the life sciences community.

### Future Plans

Their future plans include determining ways to continue to fund grid computing across the universities, including the challenge that each school uses a different funding model. They are also looking at more relationship opportunities.

### Contact

See Contact Details [61] for more information.

### Acknowledgements

The White Rose Grid project operates under the auspices of the White Rose University Consortium, which is an affiliation of the three Yorkshire Universities of Leeds, York and Sheffield. This is a collaborative venture between the White Rose Universities and our IT partners: Esteem Systems, Sun Microsystems, and Streamline Computing.

The Yorkshire and Humber Development Agency, Yorkshire Forward, is enabling us to expand our activities into the region and engage research universities and companies in e-Science.

The project has also received funding from the UK e-Science Core Programme, Esteem Systems, and the White Rose Universities.

## Grid in New York State

### Collaborators, Organizations

This grid is led by Dr. Miller's Cyberinfrastructure Laboratory. Current collaborating institutions include Columbia University, the Hauptman-Woodward Medical Research Institute, Marist College, Niagara University, SUNY-Buffalo, SUNY-Geneseo, University of Rochester, and Syracuse University.

## Summary/Description

The Cyberinfrastructure Laboratory [xx] designed and deployed a Buffalo-based grid (ACDC-Grid) and a Western New York Grid (WNY Grid) before branching out to create a Grid involving institutions throughout New York State. This statewide Grid [xx] includes resources from a variety of institutions and is available in a simple and seamless fashion to users worldwide. This statewide Grid contains a heterogeneous set of resources and utilizes general-purpose IP networks [62, 63, 64, 65]. A major feature of this grid is that it integrates a computational grid (compute clusters that have the ability to cooperate in serving the user) with a data grid (storage devices that are similarly available to the user) so that the user may deploy computationally intensive applications that read or write large volumes of data files in a very simple fashion. In particular, this statewide Grid was designed so that the user does not need to know where data files are physically stored or where the application is physically deployed, while providing the user with easy access to their files in terms of uploading, downloading, editing, viewing, and so on.

The core infrastructure for this Grid encompassing institutions throughout New York State includes the installation of standard grid middleware and the use of an active Web portal for deploying applications. Several key packages were used in the implementation of NYS Grid and other packages have been identified in order to allow for the anticipated expansion of the system. The Globus Toolkit provides APIs and tools using the Java SDK to simplify the development of OGSI-compliant services and clients. It supplies database services and Monitoring & Discovery System index services implemented in Java, GRAM service implemented in C with a Java wrapper, GridFTP services implemented in C, and a full set of Globus Toolkit components. The recently proposed Web Service-Resource Framework provides the concepts and interfaces developed by the OGSI specification exploiting the Web services architecture.

This statewide Grid represents the next Grid in an evolution from an experimental Buffalo-based grid that involved a variety of independently run organizations at SUNY-Buffalo, as well as other local institutions, including Buffalo State College, the Hauptman-Woodward Medical Research Institute, and Canisius College to a persistent and hardened heterogeneous Western New York Grid that includes Niagara University, Geneseo State College, the Hauptman-Woodward Medical Research Institute, and SUNY-Buffalo. This Grid that includes institutions throughout New York State provides a variety of applications in order to support the users at the affiliated institutions, other users in New York State, as well as users from Open Science Grid.

## Middleware Efforts

The New York State Portal [46, 47, 48, 49]. which was derived from the ACDC-Grid Portal, provides access to a dozen or so compute-intensive software packages, large data storage devices, and the ability to submit applications to a variety of grids containing tens of thousands of processors. Our Grid Portal integrates several software packages and toolkits in order to produce a robust system that can be used to host a wide variety of scientific and engineering applications. Specifically, our portal is constructed using the Apache HTTP server, HTML, Java and PHP scripting, PHPMyAdmin, MDS/GRIS/GIIS from the Globus Toolkit, OpenLDAP, WSDL, and related open source software that interfaces with a MySQL database.

Our Grid Portal provides a single-point of access to our statewide Grid for those users who want to concentrate on their disciplinary research and scholarship and do not want to be burdened with low-level details of utilizing a Grid. Applications are typically ported to the Grid Portal through our Grid-Enabling Application Templates, which provide developers with a template for porting a fairly traditional science or engineering application to our Grid-based Web Portal. This approach provides the developer with access to various databases, APIs, PHP scripts, HTML files, shell scripts, and so on, in order to provide a common platform to port applications and for users to efficiently utilize such applications. The generic template for developing an application provides a well-defined standard scientific application workflow for a Grid application. This workflow includes a variety of functions that include data grid interactions, intermediate processing, job specification, job submission, collection of results, run-time status, and so forth. The template provides a flexible methodology that promotes efficient porting and utilization of scientific routines. It also provides a systematic approach for allowing users to take advantage of sophisticated applications by storing critical application and user information in a MySQL database. Most applications have been ported to our Grid Portal within 1-2 weeks.

Our lightweight Grid Monitoring software [66] is used to monitor resources from a variety of Grids, including the statewide Grid, Western New York Grid, Open Science Grid, Open Science Grid Testbed, and TeraGrid, to name a few. With production Grids still in their infancy, the ability to efficiently and effectively monitor a grid is important for users and administrators. Our Grid Monitoring System runs a variety of scripts continually, stores information in a MySQL database, and displays the information in an easy to digest and navigate Grid Dashboard. The Dashboard is served by an Apache Server and is written in Java and PHP scripts. It provides a display that consists of a radial plot in the center of the main page that presents an overview of an available Grid, surrounded by histograms and other visual cues that present critical statistics. By clicking on any of these individual components, the user can drill down for more details on the information in question. These drilldown presentations include dynamic and interactive representations of current and historical information. For example, a user or administrator can easily determine the number of jobs running or queued on every system of any available Grid, the amount of data being added or removed from nodes on a grid, as well as a wealth of current and historical information pertaining to the individual nodes, Grids, or virtual organizations on an available Grid. Our work contributes to the widespread monitoring initiative in the distributed computing community that includes NetLogger, GridRM, Ganglia, and Network Weather Service, to name a few.

Our Grid Operations Dashboard [67] was designed to provide discovery, diagnosis, and the opportunity for rapid publication and repair of critical issues to grid administrators. The operational status of a given resource is determined by its ability to support a wide variety of Grid services, which Prescott typically refers to as site functional tests. Tests are performed regularly and sequentially in order to verify an every more complex set of services on a node. These results are reported in our Operations Dashboard in an easy to read chart.

The development of data storage solutions for the Grid and the integration of such solutions into Grid Portals is critical to the success of heterogeneous production-level Grids that incorporate high-end computing, storage, visualization, sensors, and instruments. Data grids typically house and serve data to grid users by providing virtualization services to effectively manage data in the storage network. The Storage Resource Broker is an example of such a system. Our Intelligent Migrator, currently being integrated into our Grid Portal, represents an effort to provide a scalable and robust data service to the users of this statewide Grid. The Intelligent Migrator examines and models user utilization patterns in an effort to make efficient use of limited storage so that the performance of our physical data grid and the services provided to our computational grid are significantly enhanced. Our integrated Data Grid provides users with seamless access to their files, which may be distributed across multiple storage devices. Our system implements data virtualization and a simple storage element installation procedure that provides a scalable and robust system to the users. In addition, our system provides a set of on-line tools for the users so that they may maintain and utilize their data while not having to be burdened with details of physical storage location.

### Applications

The Cyberinfrastructure Laboratory has enabled the successful porting and implementation of numerous applications to this statewide Grid.

- Shake-and-Bake(SnB) — Molecular Structure Determination Application
- Buffalo-and-Pittsburgh (BnP) — SnB and PHASES Complete Protein Phasing
- Ostrich — Optimization and Parameter Estimation Tool for Groundwater Modeling
- Aseismic Design & Retrofit (EADR) — Passive Energy Dissipation System for Designing Earthquake Resilient Structures
- Princeton Ocean Model Great Lakes (POMGL) — Great Lakes Hydrodynamic Circulation Model
- Titan — Computational Modeling of Hazardous Geophysical Mass Flows

- Chem — Commercial Quantum Chemistry Software Package
- NWChem — Computational Chemistry Software Package developed and maintained by DOE
- Split — Modeling Groundwater Flow with the Analytic Element Method

**Future Plans**

The goal of this Grid is to bring a mixture of organizations, both public and private, onto a shared grid within New York State. The nodes on the grid will include compute systems, storage devices, visualization systems, sensors, imaging systems, and a wide variety of Internet-ready devices. To date, the Cyberinfrastructure Laboratory has reached more than a dozen organizations throughout the state.

An on-going project with very positive, yet preliminary, results is our intelligent scheduling system. This system uses optimization algorithms and profiles of users, their data, their applications, as well as network bandwidth and latency, to improve a grid meta-scheduling system.

**Acknowledgements**

Funding for these grid initiatives was provided by the National Science Foundation under a series of grants including an ITR, MRI, and CRI. Additional support was provided by the Center for Computational Research at SUNY-Buffalo. Critical personnel responsible for establishing and maintaining the grid include students, staff, and post-docs from SUNY-Buffalo, namely, Jon Bednasz, Steve Gallo, Mark Green, Cathy Ruby, and Naimesh Shah. In addition, grid administrators at the participating institutions have been extremely responsive and worked in an extraordinarily collaborative fashion. We would also like to thank the members of Open Science Grid for all of their technical support, input, and advice.

# Bibliography

[1] I. Foster, C. Kesselman and S. Tuecke, “The Anatomy of the Grid: Enabling Scalable Virtual Organizations,” International Journal of Supercomputer Applications, 15(3), 2001.
[2] I. Foster and C. Kesselman, “Globus: A Metacomputing Infrastructure Toolkit,” International Journal of Supercomputer Applications, 11(2):115-128, 1997.
[3] J. Novotny, S. Tuecke and V. Welch, “An Online Credential Repository for the Grid: MyProxy,” Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), August 2001.
[4] Open Grid Computing Environment. (http://www.collab-ogce.org/nmi/index.jsp)
[5] W. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal and S. Tuecke, “Data Management and Transfer in High Performance Computational Grid Environments,” Parallel Computing, 28 (5), pp. 749-771, May 2002.
[6] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith and S. Tuecke, “A Resource Management Architecture for Metacomputing Systems,” Workshop on Job Scheduling Strategies for Parallel Processing, pg. 62-82, 1998.
[7] I. Foster, C. Kesselman, G. Tsudik and S. Tuecke, “A Security Architecture for Computational Grids,” Fifth ACM Conference on Computer and Communications Security, pp. 83-92, 1998.
[8] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, S. Tuecke. “A Resource Management Architecture for Metacomputing Systems.” Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, pg. 62-82, 1998.
[9] R.A. Luettich, J. J. Westerink, and N. W. Scheffner, ADCIRC: An advanced three-dimensional circulation model for shelves, coasts and estuaries; Report 1: theory and methodology of ADCIRC- 2DDI and ADCIRC-3DL, Technical Report DRP-92-6, Coastal Engineering Research Center, U.S. Army Engineer Waterways Experiment Station, Vicksburg, MS, 1992.
[10] Unidata Local Data Manager, 2006. (http://www.unidata.ucar.edu/software/ldm/)
[11] P. Bogden, G. Allen, G. Stone, J. Bintz, H. Graber, S. Graves, R. Luettich, D. Reed, P. Sheng, H.

Wang,W. Zhao, The Southeastern University Research Association Coastal Ocean Observing and Prediction Program: Integrating Marine Science and Information Technology," Proceedings of the OCEANS 2005 MTS/IEEE Conference. Sept 18-23, 2005.
[12] D. Huang, G. Allen, C. Dekate, H. Kaiser, Z. Lei and J. MacLaren "getdata: A Grid Enabled Data Client for Coastal Modeling," HPC2006.
[13] P. Bogden, "The SURA Coastal Ocean Observing and Prediction Program (SCOOP) Service-Oriented Architecture," Proceedings of MTS/IEEE 06 Conference in Boston, Session 3.4 on Ocean Observing Systems, September 18-21, 2006.
[14] J. Bintz et al. "SCOOP: Enabling a Network of Ocean Observations for Mitigating Coastal Hazards," Proceedings of the Coastal Society 20th International Conference, 2006.
[15] SCOOP Website, 2006. (http://scoop.sura.org/)
[15a] SCOOP Partners (http://scoop.sura.org/partners.html)
[16] North Carolina Forecasting System. (http://www.renci.org/projects/indexdr.php)
[17] S. Graves, K. Keiser, H. Conver, M. Smith. “Enabling Coastal Research and Management with Advanced Information Technology,” 17th Federation Assembly Virtual Poster Session, July 2006.
[18] G. von Laszewski, I. Foster, J. Gawor, and P. Lane, "A Java Commodity Grid Kit," Concurrency and Computation: Practice and Experience, vol. 13, no. 8-9, pp. 643-662, 2001. (http:/www.cogkit.org/)
[19] K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, “Grid Information Services for Distributed Resource Sharing.” Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.
[20] R. Wolski, N. Spring, C. Peterson, “Implementing a Performance Forecasting System for Metacomputing: The Network Weather Service,” in Proceedings of SC97, November, 1997.
[21] OSG Council (http://www.opensciencegrid.org/About/Who_is_the_Open_Science_Grid%3F/OSG_Council_Members)
[22] OSG Virtual Organizations (http://www.opensciencegrid.org/About/OSG_Organization/Virtual_Organizations)
[23] OSG Technical Activity Groups (http://www.opensciencegrid.org/About/OSG_Organization/Technical_Activities)
[24] MonALISA Graph of OSG Activity (http://monalisa.grid.iu.edu:8080/show?page=index.html)
[25] US CMS Institutions and Members (http://uscms.fnal.gov/uscms/organization/uscms_institutes_t_members.html)
[26] U.S. CMS website (http://www.uscms.org/Public/overview.html)
[27] USCMS Software and Computing (http://www.uscms.org/SoftwareComputing/index.html)
[28] CERN Archtectural Blueprint RTAG (http://lcgapp.cern.ch/project/blueprint/BlueprintReport-final.doc)
[29] Feature: Meeting the Data Transfer Challenge, ISGTW, Jan 17, 2007 (http://www.isgtw.org/?pid=1000226)
[30] 2007 Open Science Grid Consortium Meeting, UCSD, San Diego, CA, March 5-8, 2007, Frank Wurthwein, OSG Application Coordinator, OSG Extension Lead, Experimental Elementary Particle Physics, UCSD
[31] US CMS Organization, Institution, and Member Contacts (http://www.uscms.org/Public/contact.html)
[32] SDSS Institutions (http://www.sdss.org/members/index.html)
[33] SDSS Advisory Council (http://www.sdss.org/directorate/adco.html)
[34] SDSS Website (http://www.sdss.org/)
[35] SDSS — About US (http://www.sdss.org/background/)
[36] SDSS — Contact US (http://www.sdss.org/contacts.html)
[37] How ATLAS Collaborates (http://atlasexperiment.org/hac.html)
[38] Simulating Supersymmetry with ATLAS (http://tinyurl.com/2q79p9)
[39] ATLAS Experiment Home Page (http://atlasexperiment.org/)
[40] Proth (http://primes.utm.edu/programs/gallot/)
[41] Partial Differential Equation (http://www.math.ttu.edu/~smanserv/)
[42] Title: Multivariate Minimization Using Grid Computing by K. Kulish, J. Perez, P. Smith. (http://www.cs.vu.nl/ggf/apps-rg/meetings/ggf8/kulish.pdf)
[43] PhD Thesis by Dr. Eric Albers

(http://www.iemss.org/iemss2002/proceedings/pdf/volume%20uno/298_albers.pdf)
[44] SRB (Storage Resource Broker) data grid (http://www.sdsc.edu/srb/index.php/Main_Page)
[45] 3-D Studio Max graphics rendering grid
(http://www.arch.ttu.edu/resources/FAQ/3D/net_render_max_animation.asp)
[46] BLAST (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/information3.html)
[47] Query tutorial (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/query_tutorial.html)
[48] BLAST tutorial (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/tut1.html)
[49] BLAST Guide (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/guide.html)
[50] PSI-BLASTtutorial (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/psi1.html)
[51] More Information on BLAST (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/auxiliary.html)
[52] SAS-based compute grid (http://www.sas.com/technologies/architecture/grid/index.html)
[53] "Neighbors" space simulation (http://dspace.lib.ttu.edu/bitstream/2346/1219/1/thesis.pdf)
[54] Bioinformatics Project (http://www.animalgenome.org/pigs/)
[55] "R" programming language/framework (http://www.r-project.org/)
[56] Texas Tech TechGrid (http://www.hpcc.ttu.edu/techgrid.html)
[57] Sun Microsystems (http://www.sun.com/)
[58] Streamline Computing (http://www.streamline-computing.com/)
[59] White Rose Grid Compute Node (http://www.wrgrid.org.uk/ComputeNodes.html)
[60] White Rose Grid Activities (http://www.wrgrid.org.uk/Activities.html)
[61] White Rose Grid Contact Details (http://www.wrgrid.org.uk/Contactus.html)
[62] M.L. Green and R. Miller, Grid computing in Buffalo, New York, Annals of the European Academy of Sciences, 2003, pp. 191-218.
[63] M.L. Green and R. Miller, Molecular structure determination on a computational & data grid, Parallel Computing Journal 30 (2004), pp. 1001-1017.
[64] M.L. Green and R. Miller, Evolutionary molecular structure determination using grid-enabled data mining, Parallel Computing Journal 30 (2004), pp. 1057-1071.
[65] M.L. Green and R. Miller, A client-server prototype for grid-enabling application template design, Parallel Processing Letters, Vol. 14, No. 2 (2004), pp. 241-253.
[66] C.L. Ruby, M.L. Green, and R. Miller, The Operations Dashboard: A Collaborative Environment for Monitoring Virtual Organization-Specific Compute Element Operational Status, Parallel Processing Letters, Vol. 16, No. 4 (2006), pp. 485-500.
[67] C.L. Ruby and R. Miller, Effectively Managing Data on a Grid, Handbook of Parallel Computing: Models, Algorithms, and Applications, S. Rajasekaran and J. Reif, eds., CRC Press, 2007, in press.
[68] What is Condor? (http://www.cs.wisc.edu/condor/description.html)

# Current Technology for Grids

## An overview of grid fabric

A grid requires a minimum set of basic services to function properly and be distinguishable from other forms of distributed computing. Though the particular needs of the community that will utilize the grid may prescribe additional or more detailed functionality, the following basic grid services provide a commonly useful foundation:

- User interface
- Access management (authentication and authorization)
- Resource discovery and management
- Data management
- Job scheduling and management
- Grid administration
- Monitoring

Several other grid services are desirable, though not necessary, and still relatively immature to the list above, given the current landscape of grid standards and products reflecting those standards. Among these are meta-scheduling, (coordination of job scheduling and submission across resources grid-wide), user account management and reporting, shared file systems and workflow management.

Even as grid standards are still being defined, there are already many products available for implementing a grid today. Considering this, any given grid is partially defined by the functionality, focus and features of the product(s) that are used to implement it — a computing versus data grid, for instance, or scheduled versus opportunistic use of resources. The sections below provide a bit more detail on each of the basic grid services and provide examples of products commonly in use today. In particular, several functions are discussed within the context of the Globus Toolkit [1], which is an open source product that has been available for many years and has become a dominant product for assembling and managing resources in a grid, particularly among the academic community.

## User interface

Resources on a grid remain accessible at an individual system level and can therefore be accessed and used through remote login to a user account. This is still a popular access method within a grid environment, particularly for researchers who already use computers in their work and are very familiar and comfortable with this type of access. Many of these researchers are often reluctant to change even if more "user friendly" options are made available to them. Once a grid user is logged in, grid commands can be entered and executed alongside other system commands, respective of the permission parameters of the user account, and no learning curve is necessary beyond an understanding of basic grid commands. Remote login, however is arguably not a new access method, nor one that requires use of grid technologies. For instance, users have to decide where to run their jobs, and track progress themselves. For users who are not as well versed in command line access, or would prefer more automated functionality, graphical user interfaces, such as web-based grid portals can provide a less cryptic, more customized and often more efficient user experience.

A grid portal can be defined as a web-based interface that provides users with access to grid resources and services via a standard Web browser. By leveraging the Web environment and technologies, grid functions such as resource discovery, job submission, and monitoring can be combined in a portal with other useful features such as documentation, collaboration tools and "MyPortal" style customization for group or individual user views. Although grid portals are typically designed to meet the needs of specific projects or communities, the resulting functionality is often similar. Today, portals are most useful for introducing people to the grid and for running and managing small to moderate numbers of jobs. A grid portal may, however, be a more difficult method for running and tracking very large numbers of jobs, which is a necessity for some grid users.

Initially, grid portals were also designed and implemented using quite different approaches in terms of their architecture and programming. This made it difficult if not impossible to reuse or leverage components to speed the development of similar or subsequent portals. Today, the JSR168 specification [2] serves as a standard to guide portal design and implementation. JSR 168: Portlet Specification v1.0 defines three major portal components [3]:

> PLT.2.1 What is a Portal?
> A portal is a web based application that commonly provides personalization, single sign on, content aggregation from different sources and hosts the presentation layer of Information Systems. Aggregation is the action of integrating content from different sources within a web page. A portal may have sophisticated personalization features to provide customized content to users. Portal pages may have different set of portlets creating content for different users.
>
> PLT.2.2 What is a Portlet?
> A portlet is a Java technology based web component, managed by a portlet container that processes requests and generates dynamic content. Portlets are used by portals as pluggable user interface components that provide a presentation layer to Information Systems.
>
> PLT.2.3 What is a Portlet Container?
> A portlet container runs portlets and provides them with the required runtime environment. A portlet container contains portlets and manages their lifecycle. It also provides persistent storage for portlet preferences. A portlet container receives requests from the portal to execute requests on the portlets hosted by it. A portal and a portlet container can be built together as a single component of an application suite or as two separate components of a portal application.
>
> PLT.2.4 An Example 5
> The following is a typical sequence of events, initiated when users access their portal page:
>
> - A client (e.g., a web browser) after being authenticated makes an HTTP request to the portal.
> - The request is received by the portal.
> - The portal determines if the request contains an action targeted to any of the portlets associated with the portal page.
> - If there is an action targeted to a portlet, the portal requests the portlet container to invoke the portlet to process the action.
> - A portal invokes portlets, through the portlet container, to obtain content fragments that can be included in the resulting portal page.
> - The portal aggregates the output of the portlets in the portal page and sends the portal page back to the client.

Although a well-designed portal can significantly enhance the accessibility of grid computing, particularly for non-technical users, even this improved user experience most often requires that users be aware of specific details of the available resources and make educated decisions in selecting a specific computational resource for their job. This problem is complicated by the fact that different users of the same portal may see the same set of resources but be authorized to use different subsets of resources, or have access to the same resources but with differing authorization levels. Effective grid usage today often requires users to be aware of which resources they are authorized to use and also explicitly check each resource's current operational status before picking one and submitting their job to it. A user would prefer to simply have their job run on whatever resource or combination of resources would ensure the best performance, a problem that can be solved through more full-featured portals, improved system monitoring and reporting, and intelligent metascheduling.

## Access management

As discussed in the earlier section "Who can use grid resources?" users must be both authenticated and authorized to access grid resources. Approaches to this vary greatly across grid-building products, especially if academic, government and commercial sectors are all considered. PKI (public key infrastructure) is becoming an authentication technology of choice for many government uses — both grid and non-grid based — and is also the basis for authentication within Globus, which is heavily used by the academic sector. Globus GSI (Grid Security Infrastructure) relies on PKI and its related exchange of certificates, including proxy certificates, for authentication, and provides for authorization through a "grid-mapfile" that is used to associate properly authenticated users with individual system accounts. Grid users obtain an acceptable certificate through a Certificate Authority (CA) that meets the operational standards and level of assurance (LoA) of the particular grid environment they are trying working within. A grid initiative may set up its own CA for this purpose, or use certificates from an existing CA that is compatible in practice and intent. Warning: some people do not realize how much work it can be to run a CA. Before deciding to do so, it is worthwhile to investigate use of existing CAs and also to talk to others who run their own CAs to learn more about the requirements.

Though the primary need in each grid initiative is to manage access to resources and applications within its own environment, grid-to-grid integration is rapidly becoming a high priority and is a prerequisite to creating a global grid infrastructure similar in pervasiveness to the Internet. Development of interoperable PKI "fabric" for grids worldwide is coming about through the efforts of the IGTF (International Grid Trust Federation [4]. This effort is complemented by a growing recognition that mechanisms being developed for inter-institutional sharing — via a grid or otherwise — should be compatible with middleware for identity management currently under development for the higher education community, and in collaboration with the federal government, through groups such as EDUCAUSE and Internet2 (e.g. HEBCA [5], FEBCA [6], USHER [7]. The goal is the availability of secure and authoritative campus-issued credentials that enable researchers to use their local identity within and beyond the institution instead of managing multiple credentials for different projects and environments.

There is also much effort underway in the grid and middleware communities to build and enhance tools for managing virtual organizations and augmenting the Globus toolkit so that it can make more direct use of emerging security assertion-based mechanisms for authentication and authorization decisions. These mechanisms can merge the more traditional virtual organization concepts of first authenticating the user and then looking up attributes that determine what the user is allowed to do into one process that uses other backend infrastructure to deliver signed assertions specifying a user's role and/or what they are allowed to do. These technologies become even more interesting when you consider that this type of technology is also becoming widely deployed in the community for non-grid purposes. It's safe to say, however, that no single grid initiative has yet found a universally useful and deployable solution in this area.

Access management and security is a complex topic and components providing these features vary greatly depending on the product(s) used to build any given grid. Some examples of security components available within a Globus grid are:

> Overview of Globus security components [8]
>
> Web Services Authentication and Authorization [9] provides message level security through the WS-Security standard and the WS-SecureConversation specification, Transport-level security (TLS) support, and an Authorization Framework that allows a number of different

authorization schemes.

In a pre-Web services mode of grid operation, Common Authorization Service (CAS) provides a means for virtual organizations to express policies covering distributed resources across multiple sites: GT 4.0: Security: Pre-Web Services Authentication and Authorization [10].

To delegate a single credential to be shared across multiple invocations of services on a hosting environment, Globus provides a Delegation Service. This can be used, for example, for multiple GRAM job submissions or Reliable File Transfer (RFT) submissions.

For setting up a Certificate Authority, the Globus Toolkit bundles in the SimpleCA package [11], designed by the VPN Consortium and based on OpenSSL Certificate Authority software.

MyProxy is a popular package for credential management that is widely used in grid environments and should be a serious consideration for any new grid. See http://grid.ncsa.uiuc.edu/myproxy/ [12] or http://www.globus.org/toolkit/docs/4.0/security/myproxy/ [13].

Additional software packages also work with MyProxy. For instance, the Grid Account Management Architecture (GAMA [14]) adds account management capability.

Virtual Organization Membership Service (VOMS [42]) is a service that provides authorization for users within virtual organizations, using concepts of membership and roles. It is currently maintained through the Enabling Grid for E-SciencE (EGEE) project and is in large-scale use by the Open Science Grid (OSG).

Portal-Based User Registration Service (PURSE [45]) is an integrated solution that combines the SimpleCA software and MyProxy components with a back-end database and an easy to use web portal to automate user registration.

# Resource registration, discovery, and management

Resource discovery and management is a necessity in a grid environment in order to determine information about which resources can be allocated for a given grid job. A resource management service can test the conditions of the allocated grid resources, launch the job if all of the conditions are met, and report back on what happened — possibly under conditions where real-time interaction with the user is impractical (e.g., remote location, time difference).

The Globus Toolkit provides a framework for discovery and management of grid resources that comprises grid services and libraries as well as a highly standards-based security subsystem that addresses message protection, authentication, delegation and authorization. Developers can use the software, services and libraries provided to build and customize a grid environment that meets the requirements of a targeted user community. Those implementing grids on behalf of particular user communities should review the goals of the intended grid environment in order to determine which components will be desirable and required.

The current version of Globus (as of November 2006) is GT 4.0.3, which is based on industry-standard Web services protocols and mechanisms. To accommodate established grids while migrating to Web Services (WS), GT 4.x versions support "legacy" components from prior versions: pre-WS Grid Resource Allocation and Management (GRAM) for execution management, pre-Monitor and Discovery System (MDS) for information services, and pre-WS Authentication and Authorization (AA).

The Globus Toolkit may be downloaded, built, installed and configured from source or a binary installer version may be used. The Globus Toolkit 4.0 Admin Guide provides comprehensive documentation covering all options of the toolkit, pre-requisite software required, environment variables that need to be set, etc. as well as information on migrating from older toolkit versions. Application Programming Interface (API) documentation is available for C and Java.

## Data management

Data movement and management are required to provide reliable access to stored data that is used or created by compute resources. The amount of data to be manipulated may be huge, depending on the particular application. Data transfer may occur under several scenarios:

- Autonomously — independent of any particular submitted job (e.g., ad hoc file transfer or a scheduled data transfer via dedicated network shares or data grids).
- Staging — manually uploading the data to the clusters to ensure that data is available when and where it is needed for a particular job.
- As a result of computation, conditional on the outcome of the computation.
- During a computation, as an intermediate stage of the computation "data pathways".

Data management overall is a complex topic and the development of grids that are optimized for the handling of distributed data is an evolving area of research, even as basic services are being developed and deployed. Some proprietary commercial approaches are available (e.g., Avaki [43]), as well as open-source (e.g., Globus components for grid data management [44]). We hope to expand on this important area of grid development and use in future versions of the Cookbook.

## Job scheduling and management

Job submission by end-users requires some method to define job parameters such as the location path of software or data, the chosen set of computational resources, any conditional execution or triggers/blocks, and any required authentication/authorization information. These collective details form the job description that is used to queue the job for execution on appropriate resources. Workload management systems, also called Distributed Resource Management systems (DRMs), provide resource management for jobs that are submitted to run on any given resource ("local scheduling" or use of resources at a single site versus grid-wide, or meta-scheduling.)

Workload management systems are available commercially as well as via open-source. High Performance Computing vendors generally prefer or recommend a workload management system for their products but other workload management systems are available. Some of the most well-known workload management systems include:

- Load Sharing Facility (LSF) [15], a commercial system from Platform computing
- Load Leveler (LL) [16], developed by IBM for their systems
- SUN Grid Engine (SGE) [17], available commercially from Sun Microsystems and also contributed by them in an open-source version.
- The Portable Batch System (PBS) from Altair [18], available in open-source and commercial versions, with the commercial version, PBS Pro, also available at no charge to degree-granting universities.
- Condor [19] is a batch job system that can take advantage of both dedicated and non-dedicated computers to run jobs. It focuses on high-throughput rather than high-performance, and provides a wide variety of features including checkpointing, transparent process migration, remote I/O, parallel programming with MPI, the ability to run large workflows, and more. Condor-G is designed to interact specifically with Globus and can provide a resource selection service to different and multiple

grid sites.

SGE and LSF both have a foundation in the Codine Distributed Queuing System. LL has been a product of IBM for a number of years. PBS was developed by the NAS Division of the NASA Ames Research Center in the early to mid-90's, specifically for parallel systems, including cluster systems. Condor is developed by the University of Wisconsin Madison.

Each of these workload management systems offers a wide range of configuration options. Most are designed primarily for time-sharing but offer some level of space sharing configuration options. (PBS is based on space sharing but offers time-sharing options also.) What works best for any given site in terms of functionality, configuration and even policy (ability to implement policy with the technology) can vary and user requirements should be gathered to determine a best fit as part of any new installation.

A site often uses the same workload management system for all of their resources, but workload management systems in use across a grid often vary. Ideally, which workload management system is ultimately used to submit a job should be transparent to the grid user. In addition to providing a suite of web services to submit, monitor and cancel jobs in a grid environment, Globus provides interface support for several common workload management systems and directions for developing an alternative interface to shield the user from system-specific detail.

## Administration and monitoring

Grid administration tools give the administrator the sense of localized control of resources even though the grid resources may not be geographically near the administrator. Grid administration tools today are mostly for controlling authorization and authentication, however, ideally, they will evolve to model the richness and functionality of those that have evolved for local workstation system administration.

Monitoring the state of grid resources, services and job activity is an important part of managing a grid environment. It is important for grid administrators to know the current state of the grid to provide operations and support but it also an important tool for grid users. Prior to job submission, job monitoring can provide grid users with important information about what resources are accessible via the grid and the existing workloads on each. Once a job is submitted, grid job monitoring becomes a necessity for keeping track of job progress and results. Grid job monitors gather vital information about job submissions on specific resources by harvesting data from local cluster job managers such as PBS, LSF, and Ganglia. Resource allocation is also facilitated by the use of grid monitoring, which enables grid services on the various resources to be dynamically instantiated and adjusted using constantly running background processes (daemons). In Globus, examples of these background processes include Grid Resource Allocation & Management (GRAM) for job submission, a Grid Resource Information Service (GRIS) that maintains information on software and hardware configuration for a specific node, and a Grid Index Information Service (GIIS) that aggregates GRIS information for a collection of nodes.

The Globus component for grid monitoring is the Monitoring and Discovery System (MDS) [21]. The latest version of Globus, GT4, includes Web-services-based components such as WS-Resource Properties, WS-BaseNotification and WS-ServiceGroup and provides WebMDS as an interface for accessing lower level services. Information provided may come from DRM systems of the grid resources, other Globus services such as GRAM, RFT or RLS, or cluster/system monitors such as Ganglia [22], Nagios [23], or Inca [24].

Monitoring Agents in A Large Integrated Services Architecture (MonALISA [25]) provides a distributed service for monitoring, control and global optimization of complex systems. MonALISA is based on a scalable Dynamic Distributed Services Architecture (DDSA) implemented using Java / JINI and Web Services technologies. The scalability of the system derives from the use of a multi-threaded execution engine to host a variety of loosely coupled, self describing, dynamic services or agents, and the ability of each service to register itself in order to be discovered and used by other services, or clients that require such information.

## Metascheduling

Metaschedulers operate at the grid level across potentially numerous resources, gathering and analyzing information from local schedulers in order to assign user jobs to the most suitable resources at any given time. As resources are added to a grid, basic information about the grid resource is provided to metascheduler to establish ongoing communication for more effective scheduling of grid resources. Implementing a metascheduler is an advanced use of the grid and somewhat of a moving target since the design and development of metaschedulers is an active area of grid technology research and development. A metascheduler operating in conjunction with a portal, however, can significantly improve both the usefulness and efficiency of the grid.

Designing and building metaschedulers is an active field of grid research concurrent with implementation and a variety of diverse approaches are in use or being explored:

Community Scheduler Framework (CSF) is WSRF-compliant and built upon the Globus Toolkit. CSF is WSRF compliant and built upon the Globus Toolkit. A grid user may use CSF to submit jobs, create advanced reservations and define preferred scheduling policies at the grid level to access different workload managers. CSF "meta-schedules" jobs between the job management system queues.

GridWay is an open source meta-scheduler and included in the Globus Incubator [27], a program for new projects to eventually become part of Globus. GridWay enables large-scale, secure, reliable sharing of compute resources across multiple systems that may be using various workload management systems, such as PBS, SGE, LSF, Condor or others.

As noted in the earlier section on workload management systems, Condor-G [28] is a workload management product that can also work with other DRMs to provide overall dynamic job management.

United Devices provides HPC Synergy [29], as a commercial solution for optimizing an organization's existing compute resources from desktops to servers and clusters to create an on-demand environment. Synergy works with other workload management systems including LL, Condor, Open PBS and PBS Pro, LSF and SGE.

Another commercial offering is the Moab Grid Suite from Cluster Resources [30].

The EGEE Workload Manager Service is under use and development within the gLite [31] (Lightweight Middleware for Grid Computing) project.

MARS [32] is a provisioning and workflow architecture being developed by the University of Michigan.

## Account management and reporting

Grid-centric account management and reporting is still in its infancy. It currently relies for the most part on local accounting data available from the different grid resources, aggregated as much as possible through pre-packaged accounting software, or through "home-grown" code and scripts. Ideal grid accounting should give grid users and administrators feedback on the resources used by various groups and users grid-wide. This is very important for users, contributors and other stakeholders to understand the impact and extent of grid usage, and also to support and verify policy implementation such as fair scheduling and prioritizing of future jobs based upon previous use. Products are emerging to better meet the needs of grid-wide account management and reporting but more comprehensive and standards-based packages are still needed for a true "meta-view" of a grid.

Grid-wide account management and reporting begins at the level of the individual grid user account. As discussed in previous sections, grid user accounts include validation at a local source, across trusted hosts to use specific resources, software, and data. User authentication and authorization are important components. Following up with management tasks, such as creating accounts and enforcing usage and file quotas, are normal requirements at the local level that should also be viewed at the grid level. And the logical next step is to report that use in various ways and to various people such as the owners of resources (particularly when they want to know how much they use versus how much is used across the grid at large) and to the sponsors who grant the funds.

While many of us are familiar with the UNIX account management and logging tools, management across the grid goes well beyond their capabilities. A number of systems are developing to accomplish these tasks. We will summarize two, showing some of their features and components.

### User accounts

In a grid environment today, it is likely the case that accounts will be created for users both local to the site and remote from the site. (We assume that any policies needed to distinguish their use have already been addressed.) In general:

- Each grid site appoints a grid administrator that is authorized to use a centralized authentication and authorization system.
- The grid administrator creates and maintains grid accounts via the centralized authentication and authorization system. This system maintains a grid [LDAP] directory.
- Under the Globus scheme, the local Unix administrator creates and maintains standard Unix user accounts and home directories on all systems. PKI Subject Distinguished Names (DNs) are then mapped to these local Unix user accounts. These mappings are maintained in a grid-mapfile on each Globus gatekeeper. (Some grid projects provide tools for automating this process.)
- Certificates are issued to each user and a *globus* subdirectory is created in the user home directory in which to keep the certificate. (Remote users use the certificate credentials issued by their home site.)
- Grid accounts are mapped to local accounts.
- Password synchronization is done as needed. (In some cases authentication is done via certificates, but in other cases passwords may be needed and synchronization may be provided for via local tools.)

### Accounting of use

Several grid-wide accounting packages are capable of meeting the needs of a large-scale grid today.

- Gratia is software developed for the Open Science Grid to collect accounting information. From the OSG Gratia twiki page [33] "The Grid Accounting Project has:
    - ♦ designed the schema for the accounting attributes,
    - ♦ is ensuring the necessary collectors and sensors are in place in the resource providers,
    - ♦ has defined and is deploying repository and access tools for the reporting and analysis of the grid wide accounting information."
- The SweGrid Accounting System [34] (SGAS) is a Java implementation of a resource allocation enforcement and tracking service , based on the latest Web services technologies. SGAS is a soft-state, non-intrusive Grid accounting solution that includes logging and tracking in GGF Usage Record XML format and a remote and scriptable management interface.

## Shared filesystems

The appearance and utility of a single file system across grid resources would be arguably be the most effective means for accessing and staging necessary data, libraries and executable within grids jobs, as well as managing and accessing job output. As with metascheduling, this is an area of active research and development and in is infancy in terms of implementation. As a forerunner and potential model for a grid-wide file system, many high performance computer systems, clusters in particular, use the Networked File System (NFS) to create and share a single file system across multiple compute systems. Examples include:

- Parallel Virtual File System [35] (PVFS) is a popular open source solution as a high-performance and scalable parallel file system for clusters that requires no special hardware or kernel modifications. PVFS capabilities include a consistent file name space across compute systems, transparent access for existing utilities, and physical distribution of data across multiple disks in multiple clusters, and a high-performance user space access for applications.
- Global Parallel File Space (GPFS)-Wide Area Network (WAN) is another high performance parallel file system that can span systems across a wide area network. An example of GPFS-WAN in use can be found on the TeraGrid.
- Gfarm [36] from the Asia Pacific Grid (ApGrid) Grid Data Farm project is a next-generation network-shared file system that is recommended for data farms as well as clusters.
- Lustre is a popular commercial solution designed and developed by Cluster File System, Inc [37].

# Workflow processing

A workflow can be thought of as a set of tasks with dependencies. Tasks that are part of a typical grid workflow include access management, discovery and movement of data, and job execution(s). Dependencies that are attached to such tasks may range from evaluation of particular user characteristics (appropriate assurances of authentication or authorization), availability and control of data, availability and control of resources. Defining a "grid job" at the level of workflow instead of job submission helps realize the benefits of grid technology at the level of an overall user problem or inquiry versus discrete operations

Some popular software packages available for defining and managing workflows include:

The Directed Acyclic Graph Manager [38] (DAGMan), available with Condor. Once dependencies are identified, DAGMan manages these automatically between Condor jobs.

Globus Community Scheduler Framework [40] (CSF) is actually a meta-scheduler but is sometimes included in workflow services. CSF is WSRF compliant and built upon the Globus Toolkit. A grid user may use CSF to submit jobs, create advanced reservations and define preferred scheduling policies at the grid level to access different workload managers.

Pegasus [41], from the University of California's Information Sciences Institute (ISI) is a flexible framework that enables the mapping of complex scientific workflows in a grid environment. Pegasus takes an XM-based abstract workflow as input and intelligently decides how to run the workflow on a grid.

# Bibliography

[1] Globus Toolkit (http://www.globus.org)
[2] JSR168 specification (http://jcp.org/aboutJava/communityprocess/final/jsr168/index.html)
[3] JSR 168: Portlet Specification v1.0, Major Portal Components (http://tinyurl.com/324qrg)
[4] International Grid Trust Federation (http://www.igtf.org)
[5] HEBCA (http://www.educause.edu/HigherEducationBridgeCertificationAuthority/623)
[6] FEBCA (http://www.cio.gov/fbca/)
[7] USHER (http://www.usherca.org/)
[8] Overview of Globus security components (http://www.globus.org/grid_software/security/)
[9] Web Services Authentication and Authorization

(http://www.globus.org/grid_software/security/ws-aa.php)
[10] GT 4.0: Security: Pre-Web Services Authentication and Authorization
(http://www.globus.org/toolkit/docs/4.0/security/prewsaa)
[11] SimpleCA (http://www.vpnc.org/SimpleCA)
[12] NCSA MyProxy Credential Management Service (http://grid.ncsa.uiuc.edu/myproxy/)
[13] GT 4.0: Credential Management: MyProxy (http://www.globus.org/toolkit/docs/4.0/security/myproxy/)
[14] Grid Account Management Architecture (GAMA)
(http://grid-devel.sdsc.edu/gridsphere/gridsphere?cid=gama)
[15] Load Sharing Facility (http://www.platform.com/Products/Platform)
[16] Load Leveler (http://www-306.ibm.com/software/tivoli/products/scheduler-loadleveler)
[17] SUN Grid Engine (http://www.sun.com/software/gridware)
[18] Altair Engineering, Inc. (http://www.altair.com/software/pbspro.htm)
[19] Condor Project (http://www.cs.wisc.edu/condor)
[20] NSF Middleware Initiative Grids Center software distribution (http://www.grids-center.org/)
[21] Monitoring and Discovery System (http://www.globus.org/toolkit/mds)
[22] Ganglia (http://ganglia.sourceforge.net/)
[23] Nagios (http://www.nagios.org)
[24] Inca (http://inca.sdsc.edu)
[25] MonALISA (http://monalisa.cacr.caltech.edu/monalisa.htm)
[27] Globus Incubator (http://dev.globus.org/wiki/Incubator/Incubator_Management)
[28] Condor-G (http://www.cs.wisc.edu/condor/condorg/)
[29] HPC Synergy (http://www.ud.com/products/hpcsynergy.phpa)
[30] Cluster Resources (http://www.clusterresources.com/pages/products/moab-grid-suite.php%20)
[31] gLite (http://glite.web.cern.ch/glite/wms/)
[32] MARS (http://www-personal.engin.umich.edu/%7Eabose/website/marshome.htm)
[33] Gratia twiki page (https://twiki.grid.iu.edu/twiki/bin/view/Accounting/WebHome)
[34] SweGrid Accounting System (http://www.sgas.se/)
[35] Parallel Virtual File System (http://www.pvfs.org/index.html)
[36] Gfarm (http://datafarm.apgrid.org)
[37] Cluster File System, Inc (http://www.clusterfs.com)
[38] Condor Directed Acyclic Graph Manager (http://www.cs.wisc.edu/condor/dagman/)
[40] Globus Community Scheduler Framework (http://www.globus.org/grid_software/computation/csf.php)
[41] Pegasus (http://pegasus.isi.edu)
[42] VOMS: Virtual Organization Membership Service
(http://www.globus.org/grid_software/security/voms.php)
[43] Avaki EII (http://www.sybase.com:80/products/allproductsa-z/avakieii)
[44] Globus Data Management: Key Concepts (http://www.globus.org/toolkit/docs/4.0/data/key/)
[45] Globus PURSE: Portal-based User Registration Service
(http://www.globus.org/grid_software/security/purse.php)

# Programming Concepts & Challenges

## Introduction

Earlier chapters have focused on the concepts and components that form the basis of grid environments. This chapter will provide additional depth on how these can be leveraged in grid-enabled applications, and the challenges inherent in the programming of such applications.

The advocates of grid computing promise a world where large, shared, scientific research instruments, experimental data, numerical simulations, analysis tools, research and development platforms, as well as people, are closely coordinated and integrated in 'virtual organizations'. Still, relatively few grid-enabled applications exist that exploit the full potential of grid environments. This may be largely attributed to the difficulties faced by application developers in trying to master the complex interplay of the various components, including resource reservation, security, accounting, and communication. Moreover, typical grid middleware (e.g., Globus [1], Condor [2], and Unicore [3]) provide relatively complex programming interfaces and are still in the development phase such that significantly new software releases appear frequently.

Dealing with complex and changing programming interfaces is difficult in and of itself and partially responsible for the fact that few applications have been grid-enabled. An additional aspect of the problem is that we are still learning how applications in general can benefit from running on a grid and the best ways to optimize individual applications to take maximum advantage of the grid environment. Unlike homogeneous parallel machines or clusters, grid environments are heterogeneous and dynamic in nature, and subject to change at various levels:

- on the hardware level, where the application programmer has to deal with different computer architectures, chipsets, execution speeds and models,
- on the software level, including different operating systems (and versions), different compilers, inhomogeneous software environments, etc., and
- on the administrative level, where the programmer faces various and incompatible administrative policies between different grid resources.

The current grid scenario consists of:

- services (and interfaces) that are upgraded on a regular basis
- institutions (i.e. resources, services, applications) that join and leave a grid without much notice
- Changes in the application environment at run time, including services that go down without warning, resources that get busy or become available without notice, and fluctuations in the capacity of available storage.

In grid environments, conditions constantly change and at a far greater rate than situations where activities are being controlled under a single administrative domain. Today's grid middleware allows you to cope with these changes but addressing them in the most effective way can be a major programming effort. In the end, a grid-enabled application requires additional code for handling transient problems, and portions of the application code can require very frequent maintenance.

Since environmental components can change unexpectedly, such changes can easily break applications that rely on a concrete configuration, or invalidate the results of such applications. Furthermore, to run efficiently, grid applications need to be scheduled and then executed in such a manner that the differences of the resources that are actually used are properly taken into account.

The bottom line is that application programmers have to incorporate completely new and complex paradigms into their applications, which requires significant experience and effort due to the steep learning curve. Additionally we have to take into account the fact that the average application programmer is not a grid expert, but typically a domain expert wishing to solve domain-related problems.

Most applications share these problems, but code reuse is very difficult, if not impossible, because of the fundamental differences in the way applications are written and the need to make use of different grid features. Reusing Globus [1] or other middleware-oriented libraries is surely an option, but in the end nearly all grid application programmers gravitate towards creating their own abstraction layers on top of these libraries.

Ideally grid applications would adapt to the changing environment, discover required grid services at run time, and use them as needed, independent of the particular interfaces used by the application programmer. Unfortunately this is not possible in grid environments today, mainly because of the lack of standardized, widely adopted programming interfaces that can hide most of these complexities from the programmer.

# Application interfaces today

All the grid services and middleware systems described earlier offer some form of programming interface, encompassing a large variety of technologies. SOAP [4] (Simple Object Access Protocol) services provide a SDL [5] (Specification and Description Language) description. Other services (for instance GridFTP [6]) can be accessed via a well defined protocol, or a client side C API. Yet others feature a rather complex API but with a set of easy-to-use user level tools (for instance GRAM [7]). In general, the diversity of the technologies is very broad but, for each service or concept, there exists an API or programming framework designed to support that particular approach.

The overall picture has improved slightly with the emerging web services technologies. The W3 [8] consortium has defined several standards that provide at least a unified syntactic description of the particular API (via WSDL [9] and WSRF [10]) and the standardization of SOAP [5] provides a unified transport layer for these. But even if web services solve some of the diverse problems and are helping to establish a common service infrastructure, they do not solve the problem of having many different service API's for similar purposes. Thus, learning (and teaching!) programming concepts for grid technology involves understanding a number of different frameworks and APIs.

The dominant APIs and technologies today are Globus [1], Condor [12], Unicode [13] and WSDL/WSRF based services.

# Working with specific grid services

Though the grid service landscape may appear diverse at first glance, many concepts and patterns are repeated or heavily complement or overlap one another. For instance, job submission is almost always rendered in some form of (1) describing the application, (2) describing the resources to use, and (3) submitting these descriptions to an execution service that has the required executable running on a target machine. Some of the common concepts are:

- **Security:** All grid-related activities need to comply with certain security needs of the application users. Making sure confidential information stays accessible only to correctly identified and authorized persons is crucial to modern computation and even more so for grid related applications
- **File handling:** This includes (1) handling of files as a whole (copying, moving, deleting of files), (2) accessing the file content (reading and writing) and (3) querying for different file characteristics such as name, size, and details on last access. The concepts of file handling are very well understood in current operating systems and are generally reused in the grid context.

- **Replica handling:** This pertains to the handling of file replicas, which are created to provide additional reliability or scalability. The service maintains and provides access to mapping information from a logical (arbitrary) file name to a target (actual) file name. The target file name typically represents the physical location of the data. This allows for the abstraction and separation of an application's execution from concrete names in a local file system. Additionally it provides means of providing data replication management in grids
- **Information services:** This provides for both the discovery and monitoring of grid resources (compute, network, storage, etc.) and services, including information on what services may be available from the different resources, and the state of resources or services at a point in time.
- **Inter-process communication:** This concept covers the information exchange between separate jobs generally running on different resources (remote procedure calls (RPC), monitoring and notification, data transfer, etc.). Most of this is well understood in established operating systems and programming languages, and generally reused in the grid context.
- **Workflow management:** This concept defines how an application flow or business process may be automated, in whole or part. A workflow includes the documents, information or tasks that are passed from one participant (or component) to another for action, according to a set of procedural rules and dependencies.

Additional detail and examples are provided later in this section.

As often seen in computer technologies, a level of indirection, or an abstraction layer [14], can be used to resolve some problems. The definition of a general grid API can expose common paradigms and can help shield the application programmer from unwanted dynamics and technological details that arise from having a multitude of implementations.

Many of the grid toolkits used today try to present a generic API that can provide a useful level of functionality to support a variety of applications and use cases. Others (i.e. Globus, Condor) are left open for extensions, which provides for flexibility over time but can introduce interoperability problems if compatibility is not maintained across version changes.

In addition to the described programming API, many toolkits provide stand-alone tools that are usable for task execution. The tool-based approach seems to be more stable, since it can manage API interoperability issues rather than leave these to the programmer. For instance, the *globus-job-submit* tool has not changed much through progressive Globus versions since implementation changes were handled by the toolkit itself.

While APIs and toolkits are easing the development of applications to interact with specific grid services, there is still a high degree of incompatibility in the running of applications and commands across different grid environments (e.g., Globus and Unicore). A uniform approach to grid APIs — or the development of a single grid API, could reduce the variation experienced across current technologies and bring the focus instead to programming techniques and advantages for running on a grid. Advantages of this approach are obvious:

- users and programmers will have a lower learning curve because they can focus on concepts only, not on concrete implementation techniques
- a great amount of uniformity will be provided with regard to different grid middleware toolkits that have similar functionality and concepts.

The uniform API approach does have a limitation in that it is a generalization over existing API's and so would hide details and specifics. Nevertheless, experience with existing grid-enabling toolkits such as the Grid Application Toolkit (GAT [15]) or the Simple API for Grid Applications SAGA [17] standardization effort at the Open Grid Forum (OGF [18]) shows that high level, application oriented programming interfaces provide a sensible way of tackling the above mentioned problems in today's grid application landscape. This approach promises easier development and adapting of applications to run in grid environments, and the possibility of building a common and widely available grid API.

Most importantly, the key objective for a grid application interface is simplicity for the application programmer. It should be easy to use and also easy to install, administer and maintain. Remember: an applications programmer is most often a "typical" domain scientist — a physicist, chemist, biologist, linguist, or similar.

## Access to information about resources - Information services

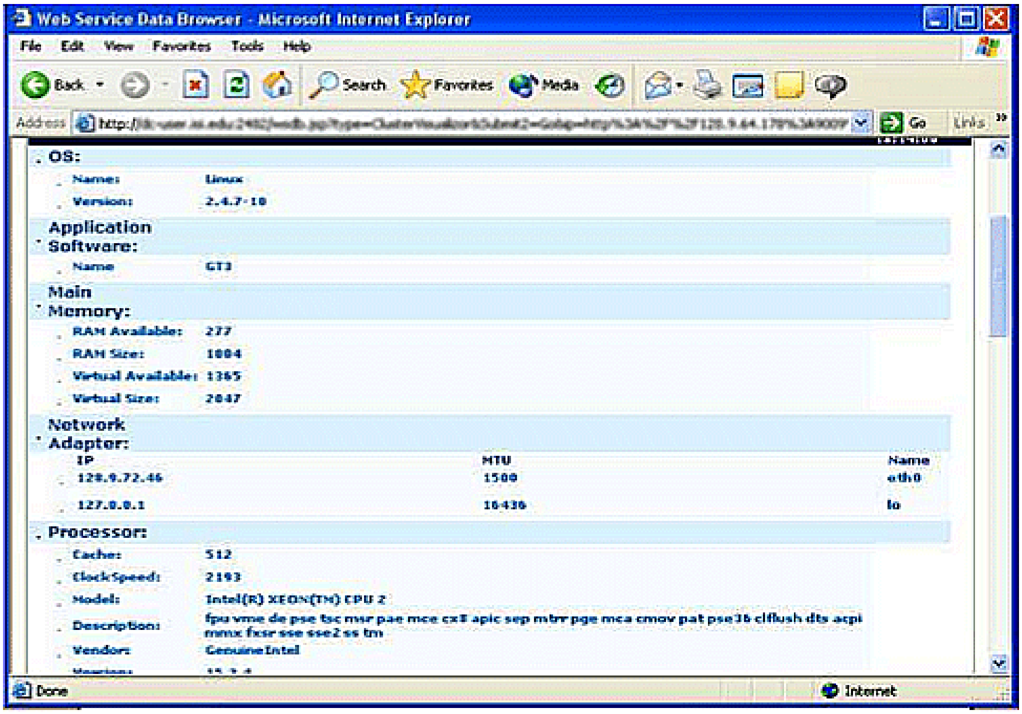
Whether you are a grid administrator or a grid user, having access to up-to-date information about the status of the grid is critical since network connections may be unreliable, resources within a vast and distributed collection may come and go, and virtual organizations can be dynamic. The grid users need to be able to determine which resources on the grid are relevant to their application requirements and available at any given time. The grid administrators must be able to monitor the "health" of the grid under their watch and make certain details of the grid available to the grid users. Standardized grid information services collect a great deal of (even customized) information about the grid, provide ways to query against that data, and then present the results in associated tables.

The Globus Toolkit Information Service Monitoring and Discovery System [22] (MDS) is probably one of the most well-known information services. The MDS system provides the capability to monitor and discover what resources are on the grid, and report status about resources as they are being used. For example, you may want to discover what computers are available, what the processor architectures are in each computer, what schedulers are in use, and what sort of load (compute, memory, disk, and so forth) is on each computer. Likewise you may need to monitor the resources on the grid to observe your job running and make sure it isn't experiencing any problems. Resource properties appropriate to specific monitoring and discovery needs can be defined via services such as GRAM, RFT, GridFTP, and RLS.

MDS collects information across multiple, distributed resources on a grid via **aggregator services** that collect real-time (or fairly recent) state information from registered information sources into an **index**. Collections of information can be queried Through various interfaces (browser, command line, and Web services.) The most recent version of MDS, MDS v4, uses XML and Web services interfaces to register the sources and locate and access information. This framework includes 1) explicit registration of the information source with the aggregator service, 2) expiration (automatic cleaning out) of registrations not renewed periodically, 3) collection by aggregator of up-to-date information from all registered information sources, and 4) support for query and publication of results.

As mentioned before, the "MDS-Index" service collects information on various Globus services and other protocol specific sources and then makes the data available in XML-based properties that can then be queried and published with standardized access methods. The data is published according to a schema that has been defined by the author/administrator or, in the case of a multi-institutional distributed grid, the collaborators. An example of the later is a schema called Grid Laboratory Uniform Environment [23] (GLUE). GLUE was developed by DataTAG [24] for interoperability between European and US Grids and is now under the Glue Working Group in the Open Grid Forum at GridForge [25].)

There are three ways of viewing MDS data: 1) Write your own application (in C, Java, Python, or .NET) using the standard Web services interface. 2) Use the command line tool *wsrf-get-property*. 3) Use the WebMDS tool, which is highly configurable, to view data using a standard web browser.



Web Service Data Browser - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address
Go
Links
Done
Internet

OS:

Name: Linux
Version: 2.4.7-10

Application Software:

Name GT3

Main Memory:

RAM Available: 277
RAM Size: 1004
Virtual Available: 1365
Virtual Size: 2047

Network Adapter:

| IP | MTU | Name |
|---|---|---|
| 128.9.72.46 | 1500 | eth0 |
| 127.0.0.1 | 16436 | lo |

Processor:

Cache: 512
ClockSpeed: 2193
Model: Intel(R) XEON(TM) CPU 2
Description: fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss tm
Vendor: GenuineIntel

Figure PC-1. A view of grid information ala MDS.

MDS also includes a **trigger** service that allows definition of rules for actions on the data (such as to whom email should be sent and what should be sent to them) and core Web services security to handle issues like who can access the indexes and data.

See A Globus Primer [26], the MDS web pages [27], and Globus Monitoring and Discover (2005 Globus World) [28] for more details about and features within MDS.

The European GridLab [29] project, in collaboration with US researchers at LSU, has developed an information service called iGrid [30]. The iGrid distributed architecture is based on two kind of Information Services, iServe and iStore GSI-enabled web services. The iServe services supplies information about a specific resource, while the iStore services aggregates information coming from registered iServe. iGrid is based on a relational DBMS and utilizes an efficient information caching policy. It can handle information extracted directly from the computational resource, where the server is running, and also user-supplied information. Thus iGrid has both system information providers and user information providers. The system provides information in XML format, while the user provides information via a web service registration method. The web service itself is based on the gSOAP toolkit, the GSI plug-in for gSOAP and the GrelC library. A push model is used to supply information to iStore from iServe services.

## Job submission and management

Two services are typically included in a computing system for processing jobs -- a job manager and a job scheduler. Sometimes these functions are handled by separate tools. In other cases one tool may have components that serve both functions. In this section we will give you a description of each service and some examples of software that performs these functions on a grid.

A *job manager* enables the site or grid administrator to define and enforce procedures and policies for running jobs on a resource based on a wide range of properties such as computing system or type, user groups, priorities, run time, queue types and lengths, and so forth. The job manager also provides the end user with methods for submitting, monitoring, and controlling jobs. In some cases the end user can define policies within his or her own collection of jobs. On a grid, local resource/job managers communicate with a global resource manager in order to provide status information to all administrators and users across the grid.

The *job scheduler* matches the job with the appropriate resources according to the requirements specified by the user. The requirements can include items like cpu type and number, run and/or wall time, memory and/or disk, restarts, checkpoints, and so forth. (And, in some cases, the job manager or scheduler can remove a job if the job requirements have been incorrectly specified.)

Job schedulers include products like PBSPro [50], and OpenPBS [35], LSF [36], LoadLeveler [37], Maui [38], Moab [39], and Globus Resource Allocation Manager [40] (GRAM). Job management is also included in PBSPro and OpenPBS, GRAM, LSF, and Torque [41].

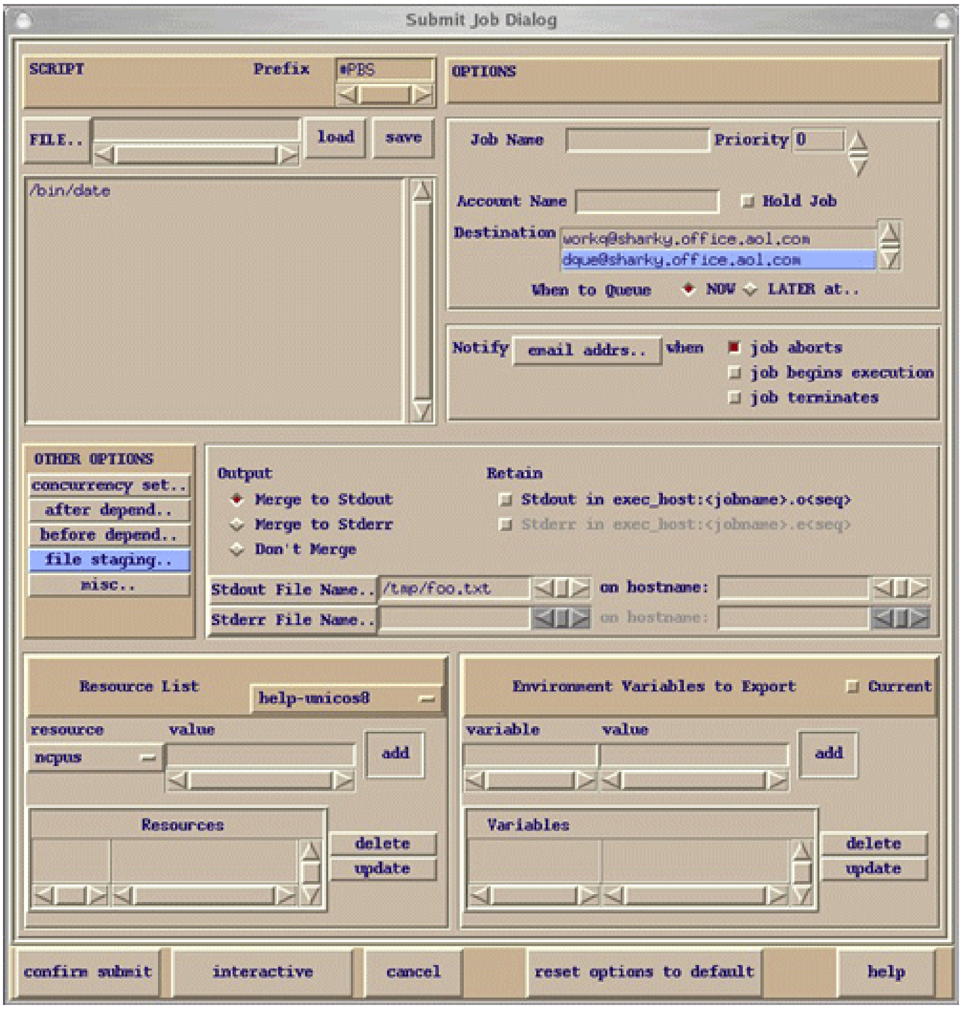
For example, PBSPro (from Altair Engineering) includes user commands such as *qsub* (submit job), *qstat* (check status of machine, queues, jobs), and *qdel* (delete job) for user management of jobs. A simple PBSPro job submission file would look something like

```
#PBS -N Strato-ozone
#PBS -l ncpus=128
#PBS -q flicker
#PBS -k oe
#PBS -m abe
cd ~/ozone
mpirun -np 128 transform
```

In this case:

- The job name is Strato-ozone (-*N*).
- The job requires 128 processors (*ncpus*) and is looking for a queue named flicker (-*q*).
- Standard standard output (*o*) and standard error (*e*) files should be kept (-*k*).
- The job owner wants email (-*m*) to be sent when the job begins (*b*), ends (*e*), or aborts (*a*).
- The job is in the owner's directory named *ozone*.
- The executable is named *transform* and has been developed as an MPI application (*mpirun*).

PBS also includes an X-Windowed interface, called xPBS. A job submission dialog interface can be used along with an interface where you can monitor hosts, queues, and jobs.



Prefix #PBS

/bin/date

Job Name
Priority 0

Destination
workq@sharky.office.aol.com
dque@sharky.office.aol.com

When to Queue ● NOW ○ LATER at..

when ☒ job aborts
☐ job begins execution
☐ job terminates

Output
● Merge to Stdout
○ Merge to Stderr
○ Don't Merge

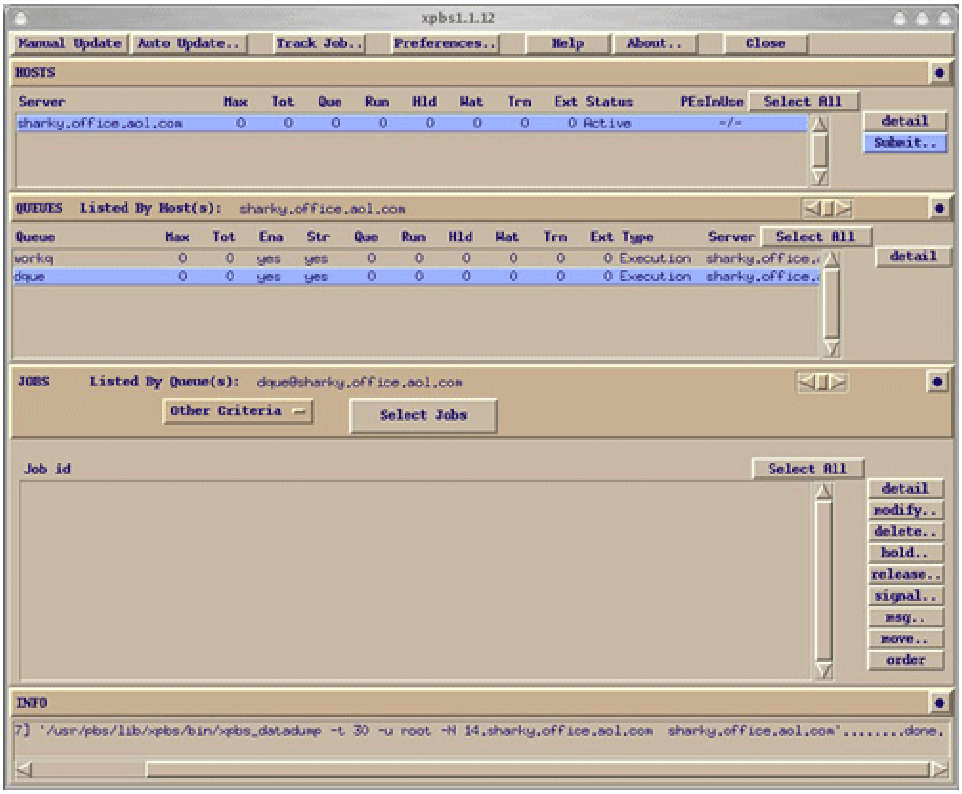
Retain
☐ Stdout in exec_host:<jobname>.o<seq>
☐ Stderr in exec_host:<jobname>.e<seq>

Stdout File Name.. /tmp/foo.txt on hostname:

Resource List help-unicos8

resource ncpus value

Figure PC-3. XPBS job submission interface.



HOSTS

| Server | Max | Tot | Que | Run | Hld | Wat | Trn | Ext | Status | PEsInUse |
|---|---|---|---|---|---|---|---|---|---|---|
| sharky.office.aol.com | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Active | -/- |

QUEUES Listed By Host(s): sharky.office.aol.com

| Queue | Max | Tot | Ena | Str | Que | Run | Hld | Wat | Trn | Ext | Type | Server |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| workq | 0 | 0 | yes | yes | 0 | 0 | 0 | 0 | 0 | 0 | Execution | sharky.office.a |
| dque | 0 | 0 | yes | yes | 0 | 0 | 0 | 0 | 0 | 0 | Execution | sharky.office.a |

JOBS Listed By Queue(s): dque@sharky.office.aol.com

Job id

INFO

```
7] '/usr/pbs/lib/xpbs/bin/xpbs_datadump -t 30 -u root -H 14.sharky.office.aol.com  sharky.office.aol.com'........done.
```

Figure PC-4. XPBS server, queue, and job information interface.

Altair Engineering also offers a browser portal called e-Compute [42] that works with PBSPro. Likewise, PBSPro provides for command line and windowed interfaces for the administrator to define queues and policies and to monitor the environment and health of the resources being managed.

The Condor project at the University of Wisconsin-Madison provides the ability to join collections of workstations and clusters together into a distributed high-throughput computing facility. Condor is also a resource scheduling system and management system for the collected resources. Condor has mechanisms for matchmaking to select an appropriate computer for a job, checkpointing and migration of jobs for reliability, running parallel jobs, and for running large workflows.

Condor can handle large numbers of jobs plus inter-job dependencies and both user and administrator defined job priorities. Condor jobs run in a number of pre-defined batch "universes", which specify how jobs are to be run (regular job, job with checkpointing, parallel job, etc.). Jobs are described in a scripting fashion similar to PBSPro and then submitted in a batch or background mode. A simple job description file would be:

```
# Example condor_submit input file
Universe = vanilla
Executable = /home/ozone/condor/transform.condor
Input = transform.stdin
Output = transform.stdout
Error = transform.stderr
Arguments = -arg1 -arg2
InitialDir = /home/ozone/condor/run
Queue
```

This file is then submitted to the universe via the condor_submit command line. The condor_submit command initiates parsing of the file and creation of a "ClassAd" that describes the job in terms of hardware architecture, operating system, memory, disk, and so forth. This ClassAd is then sent to the scheduler,which stores the job in its queue. Queues can be viewed with the *condor_q* command.

Condor submit files can describe multiple jobs which then become a "cluster" of jobs when submitted. Each job within a cluster is called a "process". This sort of feature is particularly useful in applications that require simple processing across hundreds of data files.

Condor includes additional commands to remove jobs (*condor_rm*), temporarily halt (*condor_hold*) and release (*condor_release*) a job, see the history of past jobs (*condor_history*), and specify priority order of your jobs (*condor_prio*). The Condor JobMonitor provides a viewer for job progress. Scripting options are available to enable email notification, log files, and more.

Condor can also schedule non-Condor resources through the grid-enabled version, Condor-G. In a typical scenario, Condor is layered over Globus to provide a "personal batch system" for the grid.

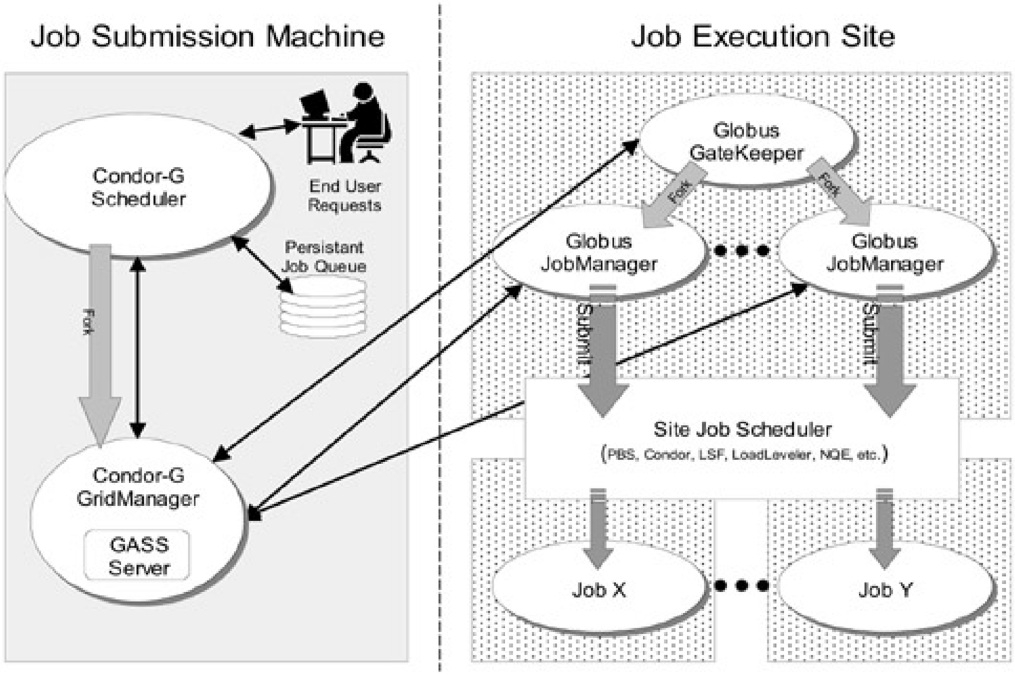


Figure PC-5. Condor/Condor-G scheduling system.

Condor-G maintains information to provide fault tolerance in case of local or remote crashes or network problems. It also provides a service called "GlideIn" that makes a wide-area grid appear to be a single Condor pool, and allows all of the Condor features, such as matchmaking, checkpointing and remote I/O, to work naturally in a grid environment.

Condor-G can also submit and manage jobs to Nordugrid [43], Oracle Database, Unicore, PBS, LSF, and remote Condor pools.

An excellent tutorial on Condor can be found at Condor User Tutorial, UK Condor Week, NeSC, October, 2004 [44]. The Condor manual [45] is also located at the Condor home page [12].

*Advance Reservation*

While schedulers and job managers continue to develop and improve, the advent of their use on distributed systems such as grids has caused interest in the concept of "advance reservation". As developing applications require more complex computational capabilities and significantly longer run-times, the ability to assure resources to successfully complete a job is becoming increasingly important.

Noteable approaches to advance reservation include:

- The AIST Grid Scheduling System [46] (GRS) for co-allocation of computing and networking resources. This approach consists of three components: a computing resource manager, a network resource manager, and a grid resource scheduler that handles requests from users via the other two.
- The NAREGI GridVM [47] which provides a virtual execution environment and advanced registration of compute nodes.
- Keahey's Virtual Workspace [48] which is an execution environment in terms of the hardware and software components required. These workspaces can be implemented in a number of ways with advance registration being explored.
- Globus Toolkit GRAM [49] which allows users to create and manage advance registration by leveraging the control provided by local resource managers.

For example, under GRAM, the reservation is a separate entity with a reservation ID. A grid user can request the reservation of specific resources for a period of time. The reservation has a specified lifetime and multiple jobs can be bound to the reservation throughout this lifetime, by the reservation owner via the reservation ID. A simple image depicting this advance registration process is provided by the Globus Alliance. Figure detail shows a client (user or administrator) creating and managing reservations through an Advanced Registration System (ARS) and Master Job Scheduler (MJS) that communicate through an adapter with the Local Resource Manager (such as PBSPro, LSF, Maui, etc.):

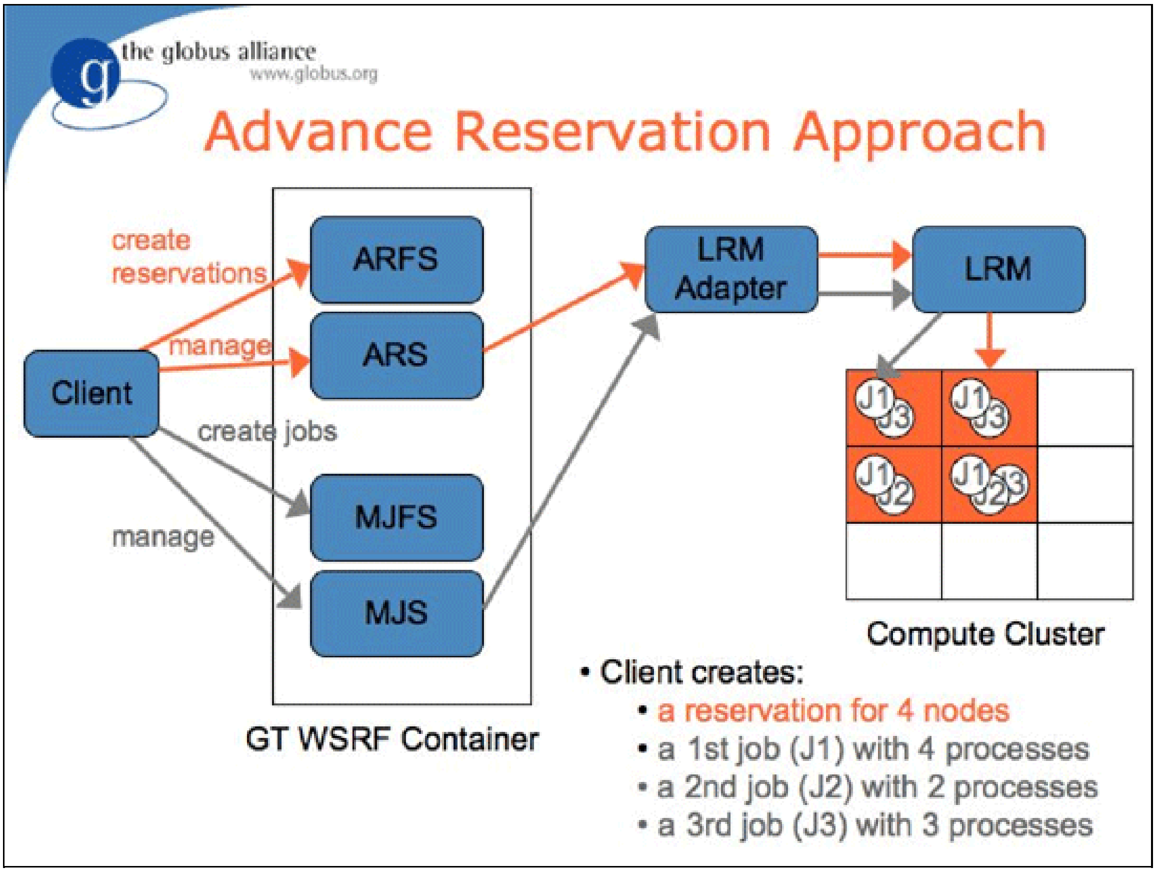


Figure PC-6. Globus advance reservation system.

## Data access, movement, and storage

Those interested in grid computing may be looking for increased computational capabilities but very frequently also have a need to process large amounts of data. To insure the movement of data where and when needed in a grid environment, bandwidth between disk, cache, memory, and CPUmust be considered.

A number of services are available to manage data in a grid environment, but they vary quite a bit within the context of different grid projects.

The Globus Toolkit GT4 divides the concept of data management into two categories: data movement and data replication. Data movement is handled by two services.

*GridFTP* [51]: GridFTP is a protocol defined by the Global Grid Forum. The toolkit provides a server implementation (with Data Storage Interface options for POSIX, SRB, HPSS, and Condor NeST systems), a command line client, and a set of development libraries for custom clients.

The command line client is called *globus-url-copy* and uses the standard get and put approach of standard ftp. For example,

```
globus-url-copy -vp -tcp-bs 5551234 -p 4 file:///mydir/mydata
gsiftp://faraway.site.org/tmp/mydata
```

will put the file "mydata" at my local machine to file "mydata" at the /tmp directory on a machine named "faraway" at site.org. Note that globus-rul-copy does not run interactively and should be part of a job script. Alternately, to get a file back, the "file" and "gsiftp" parameters are simply switched in order on the command line. Third party transfers are also supported. In this case, both files appear associated by the "gsiftp" parameter:

```
globus-url-copy -vp -tcp-bs 5551234 -p 4 gsiftp:///faraway.right.org/mydata
gsiftp://faraway.left.org/tmp/mydata
```

While GridFTP maintains a familiar concept in file transfer, it is not a web service protocol. GridFTP also requires an open socket throughout the transfer, meaning that a failure on either end cannot be recovered, which can be particularly problematic for large file transfers.

*Reliable File Transfer* [52] (RFT): RFT is a part of the web services framework and therefore provides more functionality in data movement. RFT uses standard SOAP messages over HTTP to submit and manage a set of 3rd party GridFTP transfers and to delete files using GridFTP. By submitting a list of URL pairs, the user can specify which files are to be transferred or deleted. Using this approach, the files are created after the user is properly authorized and authenticated. And since RFT keeps transfer state in a PostgreSQL database, the file transfer is recoverable in case of any failures.

There is currently no GUI interface for RFT and various command line examples can be found at the GT 4.0 RFT Command Reference [53] page.

Data replication is currently handled by the Replica Location Service (RLS). RLS is a simple registry that records where replicas exist on physical storage systems. The users of the system register the replicas and can follow up later with queries to find them. RLS is a distributed registry, making it more scalable and less vulnerable to single-point-failures (though it can be implemented as a centralized registry if preferred.) RLS maintains mappings between a logical file name and the associated physical replicas. Data replicas are very helpful in situations where large collections of data are used frequently by a group of people across distributed resources. For more information, see GT 4.0 RLS [54].

Condor [12] includes a software network called Network Storage Technology [55] (NeST) which negotiates guaranteed storage allocations (or contracts), in terms of "lots", between users and servers for specified periods of time. NeST provides flexibility in terms of size and duration of these lots as well as hierarchies (called sublots) and both user and group access control options. NeST provides multiple interfaces including protocols for HTTP, GSI-FTP (a Globus GridFTP collaboration), NFS, and its own "Chirp". And NeST provides administrators with the ability to define limits and policy as well as the automatic reclamation of storage at the end of the "contract".

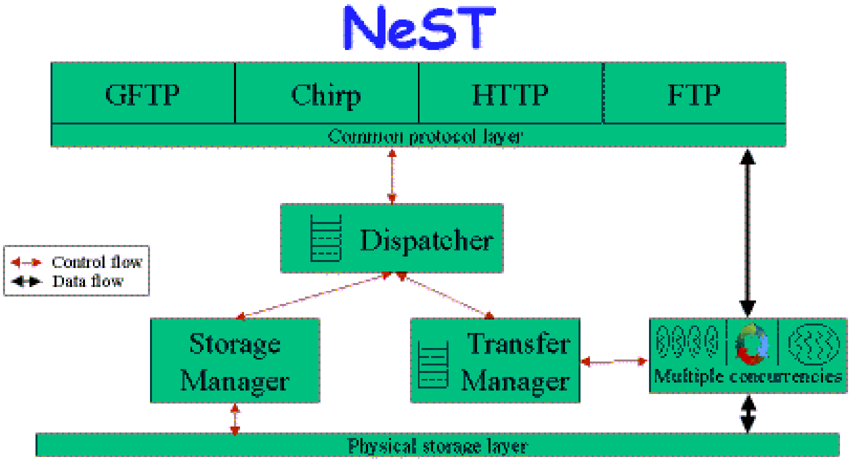


Figure PC-7. Condor NeST architecture.

Operations on a Lot include

- create, delete, and update
- movefile
- adduser, remove user
- attach/detach (binding to specific file or path)

More information on NeST is available in the following paper from the development team: Flexibility, Manageability, and Performance in a Grid Storage Appliance [56].

## Reporting grid usage

Gratia and SweGrid Accounting System (SGAS) are examples of grid-wide accounting packages that are capable of meeting the needs of a large-scale grid today.

### Gratia

Gratia is in large-scale operation on the Open Science Grid to collect accounting information. The software was developed for the Open Science Grid to meet several system requirements, as documented in the [59]:

> "The Grid Accounting Project has:
>
> - designed the schema for the accounting attributes,
> - is ensuring the necessary collectors and sensors are in place in the resource providers,
> - has defined and is deploying repository and access tools for the reporting and analysis of the grid wide accounting information.
>
> The Accounting system will properly determine a confidence level in the existing accounting information and adequately address and present erroneous or missing accounting data.
>
> The accounting system will adequately protect the privacy of the users and organizations involved.

> The auditing system will use information from the accounting system and link it to information from other sources to allow full tracking and analysis of the actions and events related to a user's resource usage.
>
> The auditing system needs to be able to present the immediate and short term information of the state and transitions in a user's use of a resource.
>
> The initial main goal for the accounting system will be to track VO members' resource usage and to present that information in a consistent Grid-wide view, focusing in particular on CPU and Disk Storage utilization".

Data is collected via a standard process, running on each node, which generates daily usage logs containing information on the jobs that ran and how many resources they consumed. This data can be used by Gratia for accounting purposes, and needs to be sent to the Gratia collector to be stored in a reporting database. The purpose of the probe is to read generated files and convert them to usage records that the Gratia program can then send to the Gratia collector.

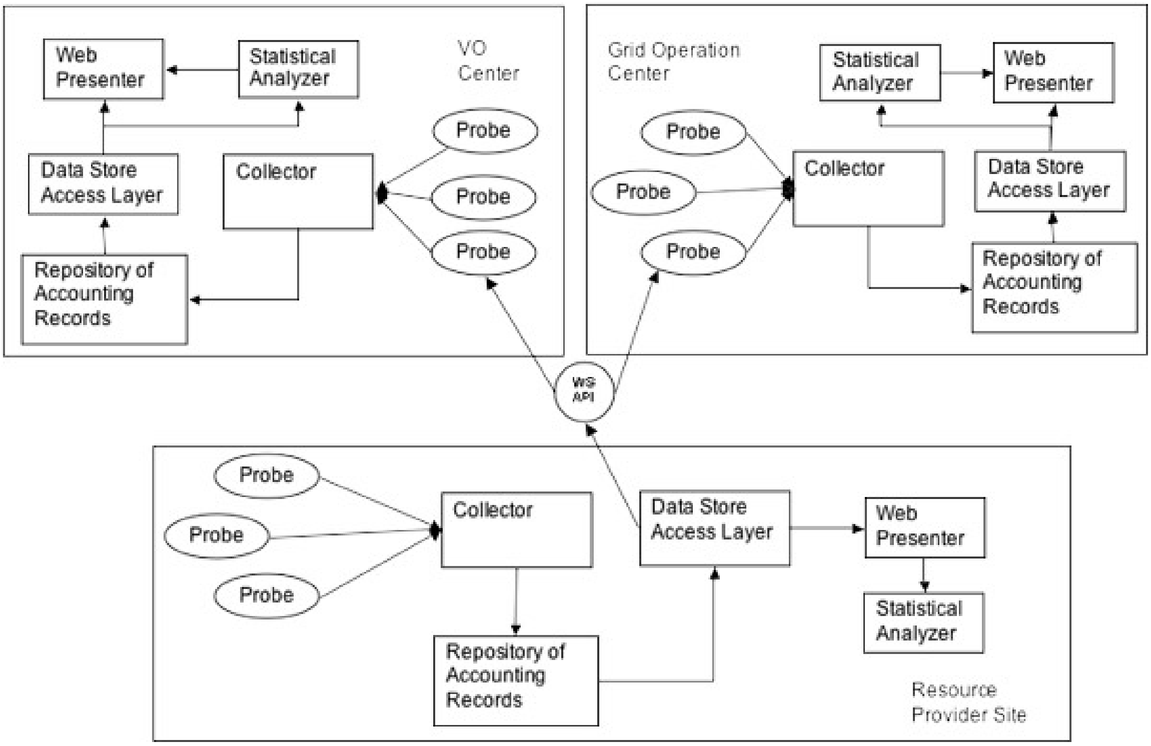


Figure PC-8. The Gratia architecture.

Gratia collects job counts and wall/cpu time used by a user, for a site, and for a VO.

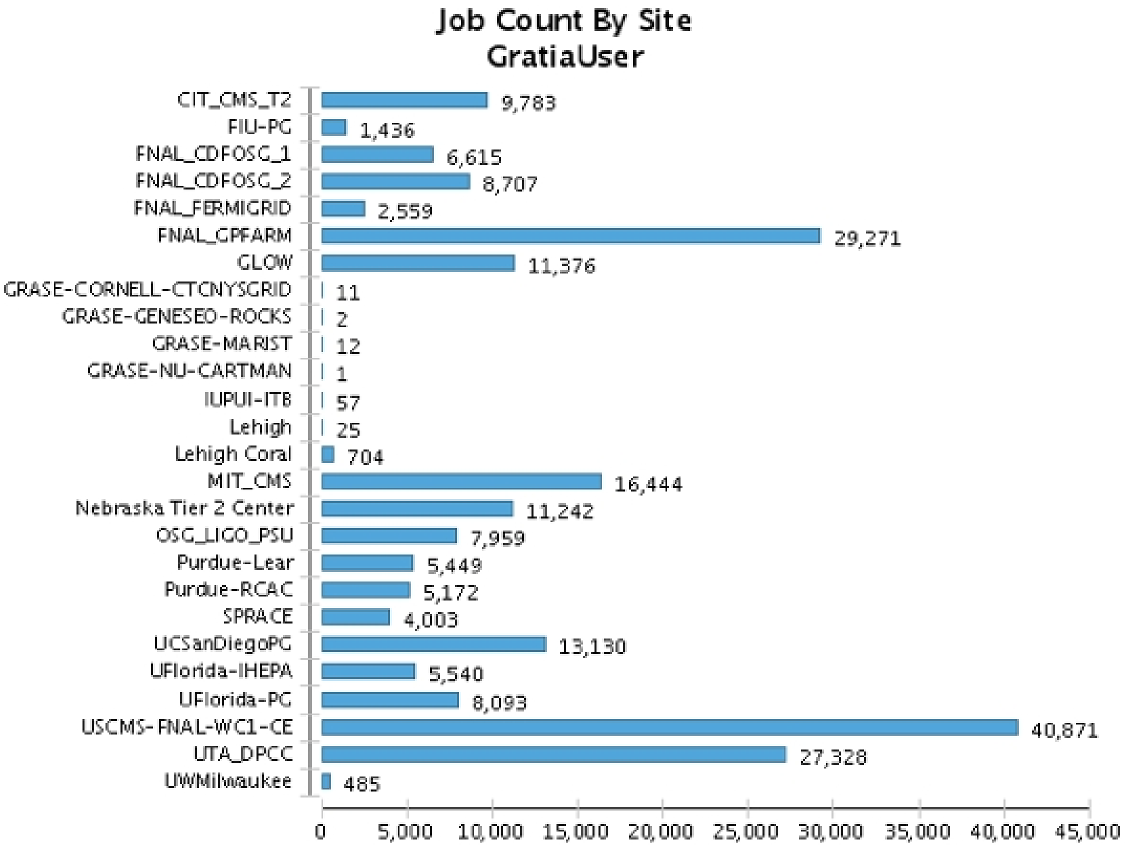


Figure PC-9. An example Gratia report.

Installation and implementation information as well as the Gratia mailing list may be found at the twiki page. See the Full Project Definition [60] for additional information as well.

### SGAS

The SweGrid Accounting System [61] (SGAS) is a Java implementation of a resource allocation enforcement and tracking service based on the latest Web services technologies. SGAS is a soft-state, non-intrusive Grid accounting solution that includes logging and tracking in GGF Usage Record XML format and a remote and scriptable management interface.

SGAS is made up of several components:

- Bank - the central service of the accounting system that maintains and enforces allocation quotas.
- Logging and Usage Tracking Service (LUTS) - a general purpose logging system for tracking resource usage in SGAS. It allows secure publication and query-based retrieval of usage data in the format of GGF UsageRecord XML.
- Job Account Reservation Manager (JARM) - a component responsible for integrating various workload managers, schedulers and local accounting systems deployed at the resource sites with SGAS. JARM is typically used as a callout to the bank during the job submission phase. The bank then issues a time-limited reservation to run the job, based on user, resource and bank policy. After the job has completed the job is logged in LUTS, and if a valid account reservation was made, JARM also charges the account in the Bank, and releases the reservation on behalf of the resource.
- Policy Administration Tool (PAT) - a component designed to manage the security policies of all of the SGAS services. It contains a command line tool that can be run in interactive or batch mode for easy scripting.

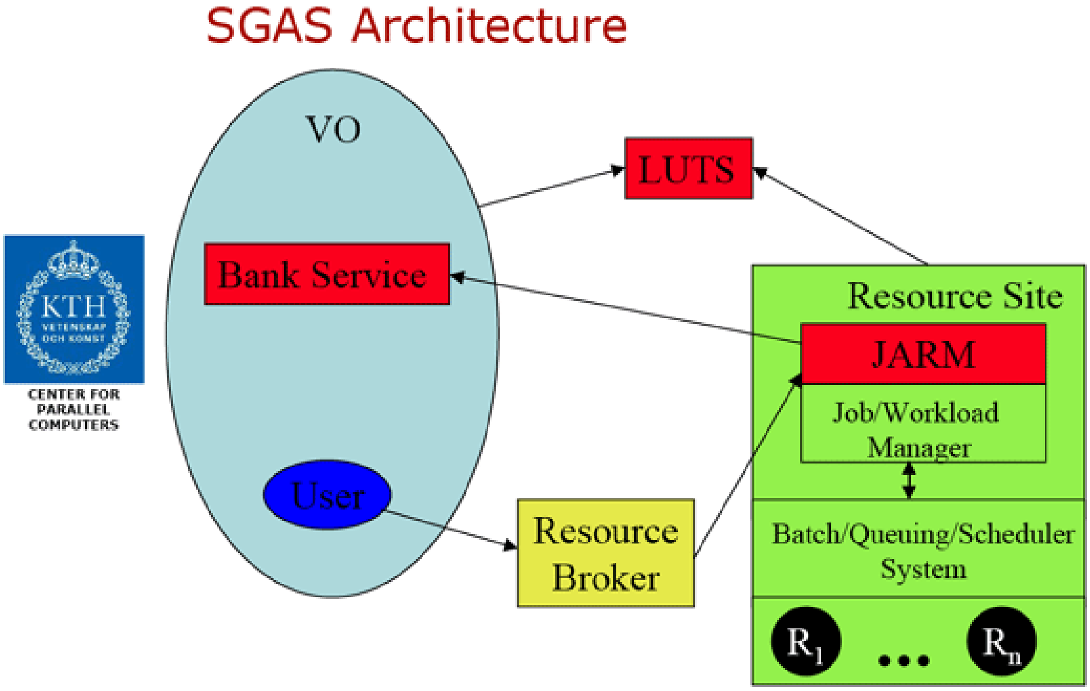


Figure PC-10. The SGAS architecture.

SGAS runs on all platforms supporting JRE 1.5.

The Globus Toolkit (GT 4) includes SGAS as part of the the available download. [62]. The BalticGrid project is using SGAS in conjunction with Globus. Their Virtual User System (VUS) requires few authorization mechanisms (VOMS, gridmap file, banned list, and SGAG) and handles privilege enforcement on several levels (meta scheduler, operation system and local scheduler, and application), with a job/account isolation level. Data is stored in the context of global user identity and VO and data is gathered for VOs as well as resource owners. Their system can be summarized in the following diagram.

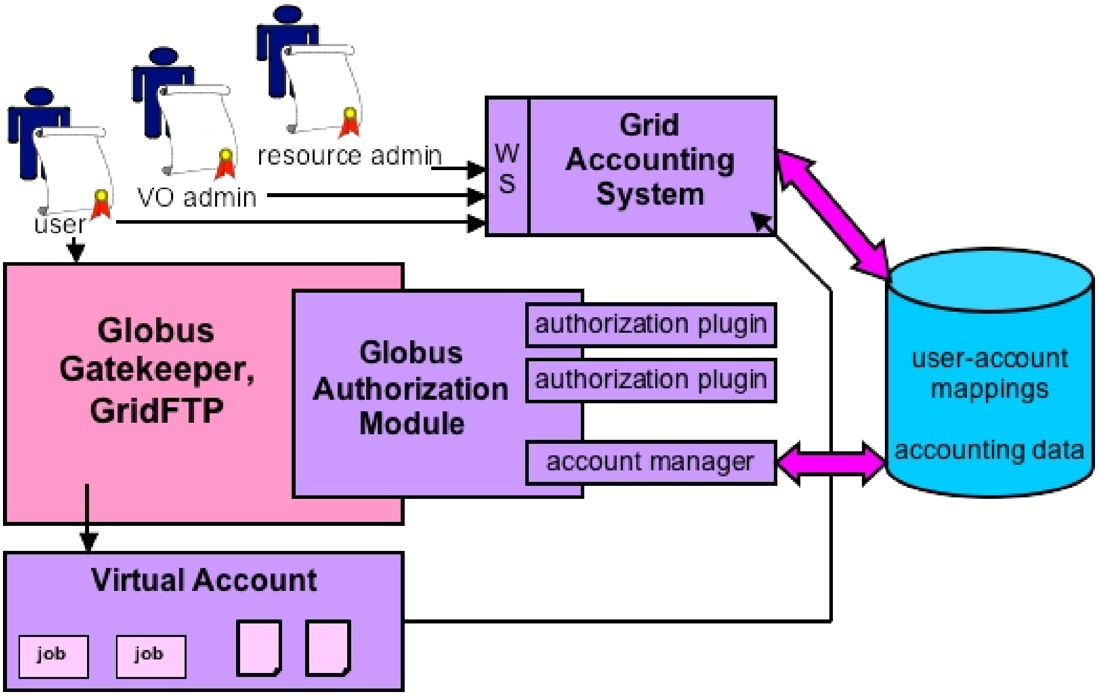


Figure PC-11. The Globus and SGAS connection.

See the SGAS Accounting System Installation and Administration Guide [63] and Administration Guide [64] for more information.

Upcoming challenges for these accounting systems include things like understanding the discrepancies between different tools, sometimes obscured by failures in the collection processes. Since the grid computing systems are made up of such diverse processors and architectures, some form of normalization is under investigation. Such tools might include better descriptions of the processors to the accounting system, performance index information, and so forth. Data transport is another area of interest. The OSG Gratia Project [65] is looking at these things.

## Workflow processing

According to Wikipedia, "Workflow at its simplest is the movement of documents and/or tasks through a work process. More specifically, workflow is the operational aspect of a work procedure: how tasks are structured, who performs them, what their relative order is, how they are synchronized, how information flows to support the tasks (wordflow) and how tasks are being tracked. As the dimension of time is considered in workflow, workflow considers "throughput" as a distinct measure. Workflow problems can be modeled and analyzed using graph-based formalisms like Petri nets."

Clearly, as computing resources become available in more distributed fashion, as the tools to use them expand, and as our expertise in using them develops, the concept of managing our workflow across this grid of software and hardware resources emerges. In this section we will give several examples of grid products and services that support this concept of workflow management.

### Condor DAGman

Condor's Directed Acyclic Graph Manager [66] (DAGman) allows you to specify the dependencies between Condor jobs. (For example, jobs can be ordered chronologically.) DAGman works through a data structure called the "DAG", a dependency graph where each job is a node and can have multiple parents and children providing no loops are created. A DAG is created via a ".dag" file such as:

```
# ozone.dag
Job A jacobian.sub
Job B vadvection.sub
Job C xadvection.sub
Job D xdiff.sub
Job E thdiff.sub
Job F predict.sub
Parent A Child B C
Parent B C Child D, E
Parent D E Child F
```

Here we have six condor jobs (*Job A* through *Job F*) where each job is specified in its associated condor_submit file (*jacobian.sub* through *predict.sub*.) The DAG is put into action via the *condor_submit_dag* command which runs DAGMan itself as a Condor job to benefit from Condor's reliability mechanisms. (See the "Job submission and management" section above for a brief example of the Condor universe.)

A visualization of this process, from the Condor tutorial job, shows that Parent A starts and, upon completion, child jobs B and C start up.

```
# diamond.dag
Job A a.sub
Job B b.sub
Job C c.sub
Job D d.sub
Parent A Child B C
Parent B C Child D
```

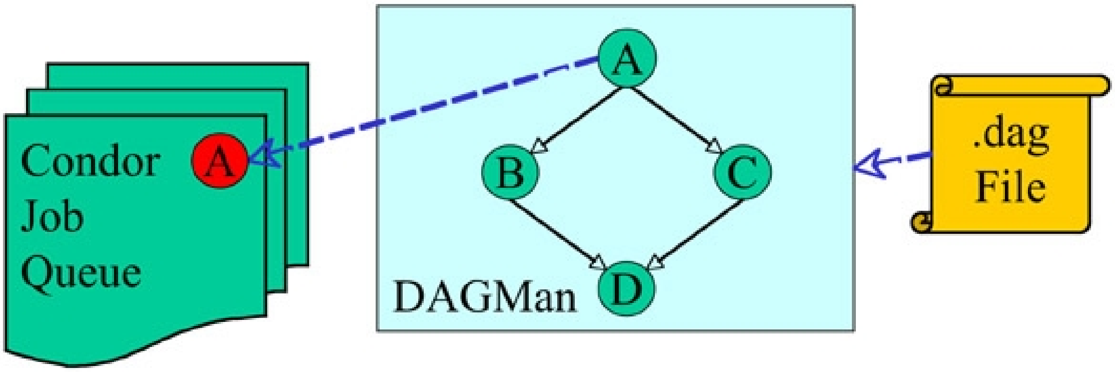


Figure PC-12. DAGman workflow diagram.

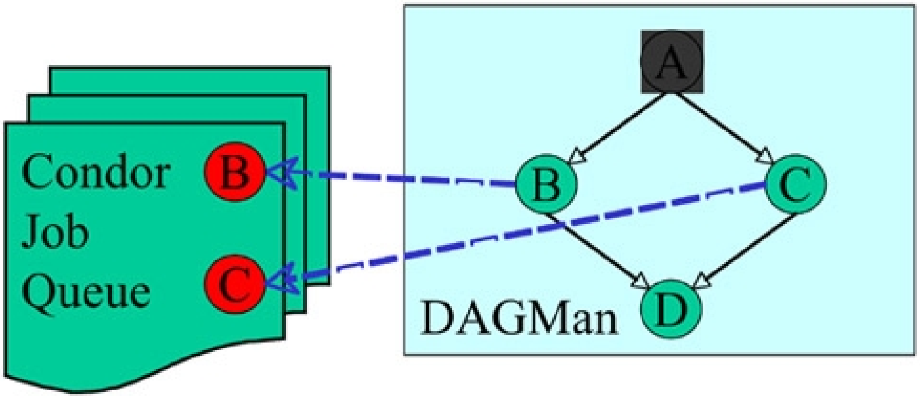


Figure PC-13. DAGman workflow progress.

The process continues until all jobs complete. In case of failure at any step, DAGman will continue as far as possible and then create a "Rescue File" which holds the current state of the DAG job. Once the problem has been resolved, the rescue file can be used to restore the DAG to its prior state. DAGman will continue in this manner until the entire DAG job completes.

DAGMan has been used to run DAGs of tens of thousands of nodes in production. When running so many nodes on a grid, many failures are likely to occur, therefore DAGMan provides a variety of features to reliably run and scale large DAGs.

See the Condor manual [67] for a complete list of DAGman features.

### Swift

Swift [68] is a system that builds on and includes technology previously distributed as the GriPhyN Virtual Data System [69]. It provides for the specification, execution, and management of large-scale science and engineering workflows. It supports applications that execute many tasks coupled by disk-resident datasets, for example, when analyzing large quantities of data or performing parameter studies or ensemble simulations.

Swift is open source software that combines:

- a simple scripting language to enable the concise, high-level specifications of complex parallel computations, and mappers for accessing diverse data formats in a convenient manner. Simple examples of use can be seen at A Swift Tutorial [70]. Swift also provides visualizations through generation of provenance graphs. Swift scripts can be run locally or on remote systems. The same script files can be used in both cases with modifications made only to a "site catalog" file that is in XML format.
- an execution engine that can manage the dispatch of many (100,000+) tasks to many (1000+) processors, whether on parallel computers, campus grids, or multi-site grids. The runtime engine is configured through properties. Properties are define at the global, user, and command line levels. Properties include such things as site and transformation catalog locations, IP address of GRAM service, caching algorithm information, provenance graph settings, job clustering information and settings for kickstart information gathering and throttling (setting limits for concurrent activities such as workflow instances, tasks/jobs, file transfers, and so forth).

For more information on swift, see The SwiftScript User Guide [71] and the Swiftscript Language Reference Manual [72].

### Pegasus

Planning for Execution in Grids [73] (Pegasus) is a workflow-mapping engine developed and used as part of several NSF ITR projects (GriPhyN, NVO, and SCEC-CME). Pegasus automatically maps high-level workflow descriptions onto distributed infrastructures such as the TeraGrid and Open Science Grid. Pegasus:

- enables scientists to construct workflows in abstract terms without worrying about the details of the underlying Cyberinfrastructure,
- provides robustness and reliability through dynamic workflow remapping,
- automatically manages data generated during workflow execution and capturing their provenance information,
- is used in a variety of scientific applications ranging from astronomy, biology, earthquake science, gravitational-wave physics and others,
- is used day-to-day to map complex, large-scale scientific workflows with thousands of tasks processing TeraBytes of data onto the Grid.

Pegasus improves the performance of applications through data reuse to avoid duplication and increase reliability, workflow restructuring to improve resource allocation, and automated task and data transfer scheduling. Pegasus provides reliability through dynamic workflow remapping and DAGman workflow execution. Pegasus uses Condor and Globus middleware for distributed environments and:

- provides a level of abstraction above gridftp, condor-submit, globus-job-run, etc commands
- provides automated mapping and execution (via DAGMan) of workflow applications onto distributed resources
- manages data files, can store and catalog intermediate and final data products
- improves successful application execution
- improves application performance

- provides provenance tracking capabilities
- supplies client-side tools
- provides an OSG-aware workflow management tool

Pegasus usage examples are beyond the scope of version 1 of this cookboo but can be found in the GriPhyN Virtual Data System Quick Guide [74].

## Security and security integration through authn/authz

Today we are encouraged to use more and more Internet-based services, be they online banking, concert ticket purchases, attractive free software options, instant messaging, blogs and wikis, or grid computing. As these services and their sources proliferate, the more nervous we might (and should) become about whether or not interaction with a service is secure. Available grid technologies provide various levels of and mechanisms for security within the grid environment they support.

As has been discussed already, grids often result when multiple institutions form virtual organizations to accomplish tasks beyond the ability of any single institution. These virtual organizations are structured with different members having various privileges, often based on roles and relying on agreed-upon methods to determine and enforce both roles and privileges. For example, some members might only have the right to develop and run software, while others might serve as community administrators. Through these processes, virtual organizations are better able to maintain the integrity of their resources and data, at least with respect to the VO itself.

The more difficult aspect is to maintain security across grid environments, or for resources that are connected to more than one grid environment. Standards development in several related security areas is taking place within the Open Grid Forum, including user authentication, authorization and firewall issues [76]. In addition, organizations such as the IGTF (International Grid Trust Federation), are working to synchronize policies across grid initiatives in order to develop and maintain a global "trust fabric" that supports scaleable and reliable identification of grid users and resources [77]. As this work is ongoing, research projects are also underway to develop necessary supporting architecture and software. One notable example is the GridShib project [31], an NSF funded project of NCSA and the University of Chicago, to integrate the federated authorization infrastructure of Shibboleth [32] with the Globus Toolkit.

As discussed earlier in the Cookbook, the Globus Toolkit provides security via X.509 credentials. Identity-based authorization is provided via access control lists ("gridmaps") mapping to local identities (Unix logins) and a Community Authorization Service (CAS). The Shibboleth project offers a large base of campus use around the world via a standards-based and open source implementation and a standard vocabulary for describing user attributes. With this, Shibboleth has resulted in a well-developed, federated identity management structure.

From the GridShib website [31], "The goal of GridShib is to allow interoperability between the Globus Toolkit® from the Globus Alliance [1] with Shibboleth [32] from Internet2 [33]. As a result, GridShib enables secure attribute sharing between Grid virtual organizations and higher-educational institutions." GridShib provides attribute-based authorization based on Shibboleth.

In addition, while basic security measures are in place to support virtual organizations, the size and complexity of the virtual organizations they can support are limited by the ability of resource managers to manage the privileges of each user in the virtual organization. To address this scaling issue, using GridShib, virtual organizations can use access control methods based on user attributes instead of identity. As a result, resource managers need not know all of the users in the virtual organization, just their attributes (for example, Data Analyst or Software Developer).

The GridShib project has five basic goals:

- Integrate X.509 and SAML [75] to provide enhanced Grid Security Infrastructure (GSI).
- Enable attribute sharing between virtual organizations and higher-educational institutions.
- Develop and implement profiles to securely share attributes across administrative domains.
- Investigate attribute-based access policy enforcement for grids.
- Generalize attribute-based authorization policies in the Globus Toolkit runtime environment.

And GridShib has developed around three use cases: established grid user, new grid user, and portal grid user. See the Technical Overview [34] for more details on current use cases and plug-ins.

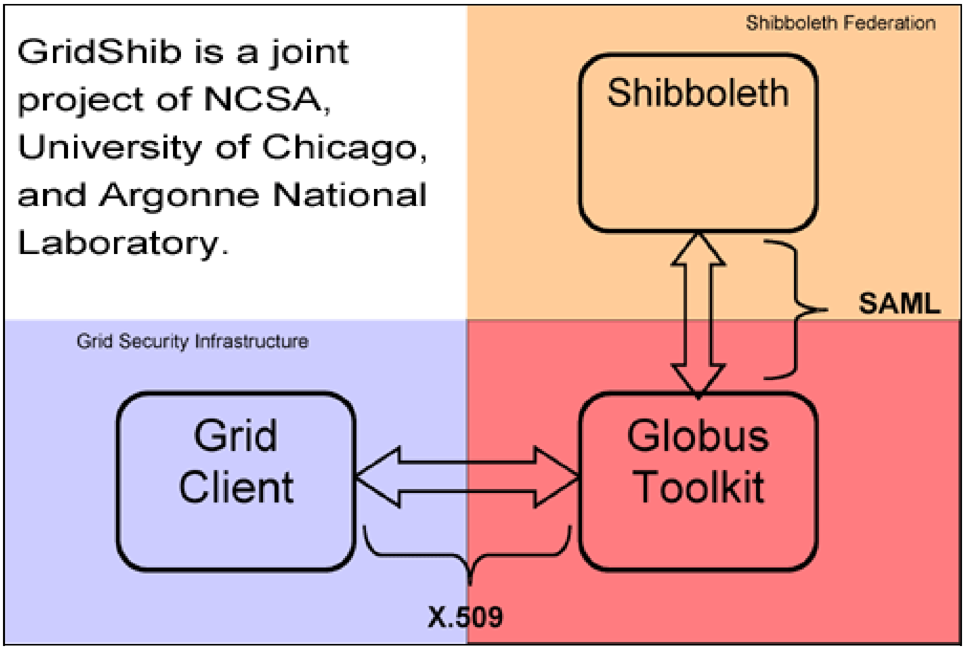

GridShib is a joint project of NCSA, University of Chicago, and Argonne National Laboratory.
Shibboleth Federation
Shibboleth
SAML
Grid Security Infrastructure
Grid Client
Globus Toolkit
X.509

Figure PC-2. The GridShib relationship.

# Grid-enabling application toolkits

## Overview of existing frameworks

There have been several attempts to build grid enabling application toolkits in the past. These toolkits aim to provide client side abstraction layers mainly for grid middleware services and related dynamic 'features' in order to increase both the speed and ease with which grid applications can be deployed. Figure PC-14 illustrates the place of an application within a grid environment and its main interface to the grid (shown in red),which is the focus area for application-oriented grid-enabling toolkits.

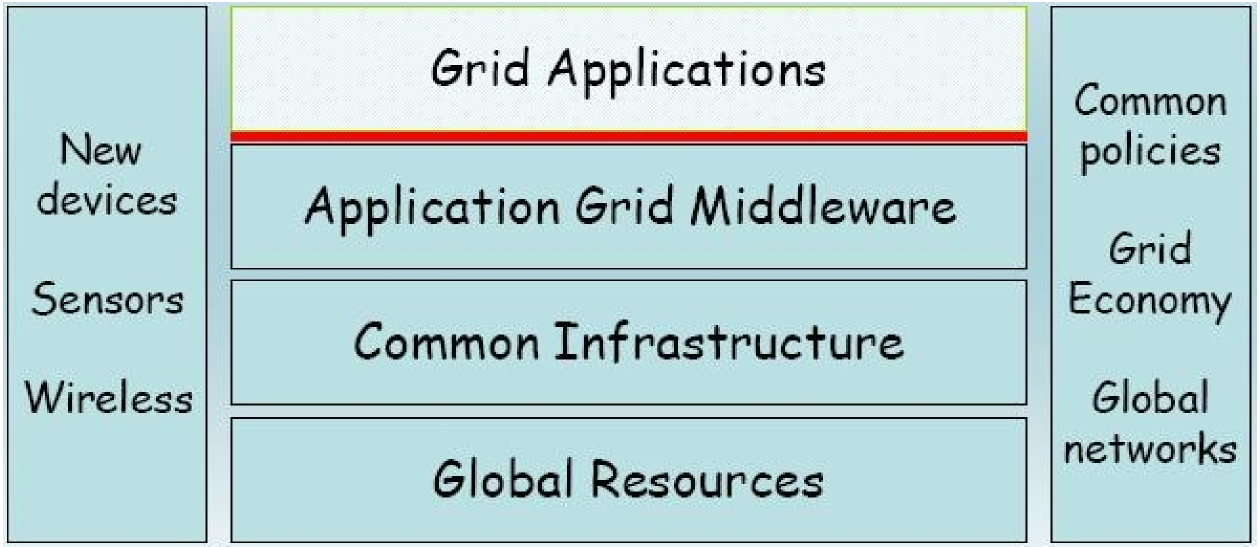


Figure PC-14. An application in a grid infrastructure.

Experience in the development of different grid enabling application toolkits suggests that a required main feature is that they be easy to use. Ease of use includes:

- Exposing a simple and consistent API which allows error tracing to be invariant,
- Making upgrades easy to perform and not reliant on specific versions of grid middleware,
- Exposing a well defined API that is designed to change rarely and to be upward compatible if changes are required,
- Supporting implementations that allow dynamic exchange of key elements (possibly at runtime) and provide runtime abstractions,
- Avoiding refactoring/recoding/recompilation whenever some underlying middleware component may have been changed,
- Ideally, a grid application should be designed to run reliably locally and on the grid, over time, and in light of differences encountered in the grid environment (e.g. the operating system of various resources, versions of grid middleware, etc.).

Applications should also utilize well-known programming paradigms. For example, a file API should provide expected functionality (namely open, close, read, write, seek) versus the introduction of less-straightforward mechanisms (e.g., asking a discovery middleware service to tell the application the location of a middleware service that can then give the location of the requested file).

## Toolkit example: Simple API for Grid Applications (SAGA)

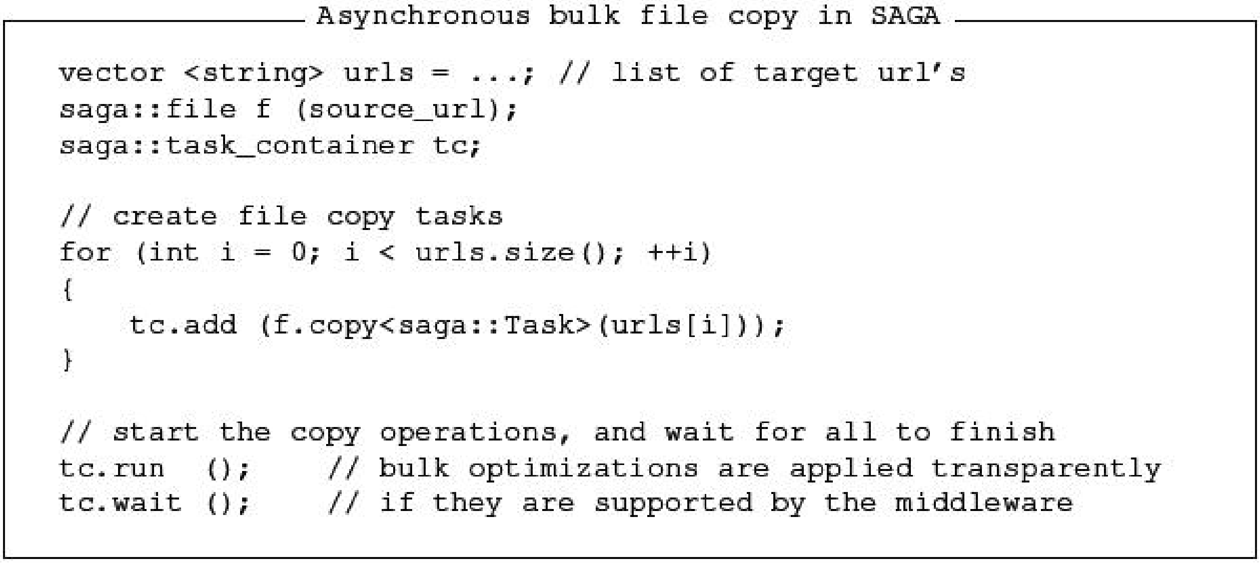
SAGA (Simple API for Grid Applications) has been defined by the Open Grid Forum (OGF) [17] as a high-level API that directly addresses the needs of application developers.The purpose of SAGA is two-fold:

1. Provide a simple API that can be used with much less effort compared to the vanilla interfaces of existing grid middleware. A guiding principle for achieving this simplicity is the 80-20 rule: serve 80% of the use cases with 20% of the effort needed for serving 100% of all possible requirements and
2. Provide a standardized, portable, common interface across various grid middleware systems and their versions.

SAGA is a prominent recent API standardization effort that intends to simplify the development of grid-enabled applications, even for scientists having no background in computer science or grid computing. SAGA was heavily influenced by the work undertaken in the Gridlab [19] project, in particular by the Grid

Application Toolkit (GAT) [15] — one of the first major attempts to build a high level API to grid services. A public call for use cases produced about 25 different examples that served as input to the SAGA development team.

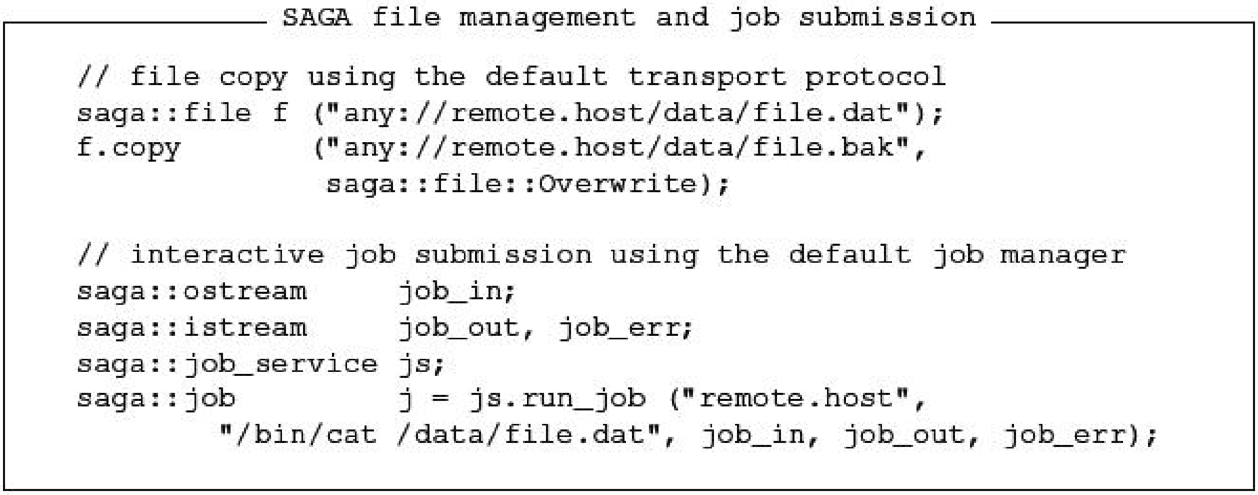
The following examples of code show typical SAGA use cases and illustrate the intended simplicity of SAGA. The code is based on the SAGA C++ reference implementation [21] currently developed at the Center for Computation and Technology at LSU, Baton Rouge. Note, that the code presented is completely independant from the underlying middleware services that are used.



```
Asynchronous bulk file copy in SAGA

vector <string> urls = ...; // list of target url's
saga::file f (source_url);
saga::task_container tc;

// create file copy tasks
for (int i = 0; i < urls.size(); ++i)
{
    tc.add (f.copy<saga::Task>(urls[i]));
}

// start the copy operations, and wait for all to finish
tc.run  ();    // bulk optimizations are applied transparently
tc.wait ();    // if they are supported by the middleware
```

Figure PC-15. Asynchronous bulk file copy in SAGA: a file is copied to a number of remote locations using well-known C++ programming paradigms.

Figure PC-15 above shows how easy it is to copy a set of files to different remote locations. Interestingly enough this code even applies certain optimization techniques, for instance the use of bulk operations if the available middleware services support this.



```
SAGA file management and job submission

// file copy using the default transport protocol
saga::file f ("any://remote.host/data/file.dat");
f.copy       ("any://remote.host/data/file.bak",
              saga::file::Overwrite);

// interactive job submission using the default job manager
saga::ostream     job_in;
saga::istream     job_out, job_err;
saga::job_service js;
saga::job         j = js.run_job ("remote.host",
        "/bin/cat /data/file.dat", job_in, job_out, job_err);
```

Figure PC-16. SAGA file management and job submission. The code is dessigned to be independent from the deployed Grid middleware.

Figure PC-16 shows code that first backs up a remote file and then starts a (in this case trivial) job operating on the copied file, intercepting the standard input and output (console) streams this remote job may use.

## Requirements Analysis

As outlined above, any grid enabling application toolkit must cope with a number of very dynamic requirements while providing a "simple" and "easy-to-use" API. The following explains these requirements in additional detail.

### Dynamic Specification Landscape

The Open Grid Forum (OGF) [17] is an international standardization body whose primary objective is to define a set of standards in the emerging field of grid computing. OGF specifications will cover grid architectures, protocols, interfaces, and APIs. However, the whole field is young and the complexity of grids is not yet completely understood, in terms of academic research or for industrial and commercial applicability and impact. This fact, along with the complexity of the problem itself, is causing the grid specification landscape to evolve slowly. There are several significant gaps in the scope of standards being explored, and it is also generally expected that existing specifications will change. The time needed for grid standards to stabilize is estimated to be 5 to 10 years, however, the expectation for grid computing to solve real world problems remains very high, partly due to the initial enthusiasm (or hyping) in the field. This dichotomy creates frustration for end users in particular since scalability and interoperability are multi-layered problems and difficult to solve. These observations imply the necessity of an interface abstraction for early adopters to shield grid application development and deployment from the evolving grid landscape and provide a reasonable migration path to future grid systems. Thus, any SAGA implementation must include mechanisms for coping with evolving grid standards and changing grid environments.

### Evolving Grid Specifications

The SAGA specification itself is currently limited and expected to expand in scope over time. In particular, it is expected that new SAGA extensions will be required to provide programming paradigms for emerging grid standards to the application developers. The general look and feel of the SAGA specification, however, is thought to be more stable and that extensions will be merely semantic (new objects, new method calls) but with limited or no syntactical additions (no change to the object or task models, for instance). Any given SAGA implementation must be able to cope with future SAGA extensions easily, without breaking support and backward compatibility for early SAGA adopters and applications.

### Dynamic Grid Environment

As grid middleware evolves, deployed grid environments face constant changes of middleware deployments (e.g., new versions and services are rolled out frequently, often with unclear migration paths). Grid environments are also dynamic by design, with respect to the availability of services and other resources. Any application designed to run on grids needs to implement fail safety mechanisms for coping with such changes and not rely on the static configuration or availability of resources. Much of dynamism, however, can be hidden from the application programmer through the use of APIs and toolkits. For example, an upgrade in a services protocol version could be handled in the client libraries communicating to the service and not at the application level. Resource discovery, fail safety on service failures and simple fallbacks such as redundant service deployments are other examples of mechanisms that are vital to the successful running of a grid applicationbut should ideally not need to be provided in application code. A SAGA implementation must allow for and, where possible, actively support fail safety mechanisms, and hide the dynamic nature of grid resource availability from the application.

### Heterogeneous Grid Environment

The dynamism of grid environments is also reflected in their potentially heterogeneous nature. Although many current grids focus or are heavily based on Linux based clusters, grids conceptually are designed to cope with any OS (real or virtual) running on any platform. (The predominance of Linux is more an indication of the state of grid middleware development today than an intentional design.). A SAGA implementation must be portable and platform independent, both syntactically and semantically.

### Distributed Grid Applications

Within the domain of distributed applications, which always imply remote communication, latency considerations play a major role in application design and implementation. A number of application domains have emerged that cope particularly well with latencies present in distributed environments, by loosely coupling distributed components or utilizing latency hiding techniques. Latency hiding techniques (such as caches, bulk operations, and interleaving of computation and communication) often require application level information (e.g. concurrency information of operations) to be effective. A library designed for distributed applications must allow these and other latency hiding techniques to be implemented.

### End User Requirements

As previously noted, the SAGA specification was developed based on the responses to a call for use cases from the grid community and is designed to meet the resulting end user requirements. An API implementation must meet other end user requirements outside the scope of the actual API specification, such as ease of deployment, ease of configuration, documentation, and support of multiple language bindings. If any of these properties is missing, acceptance and utility within the targeted user community can be severely limited.

## The SAGA C++ Reference Implementation

Thus far, we have covered the motivation and design objectives for a SAGA implementation. This section will summarize the resulting properties of the SAGA C++ reference implementation from an end user perspective. The following picture shows the overall architecture of the SAGA implementation.

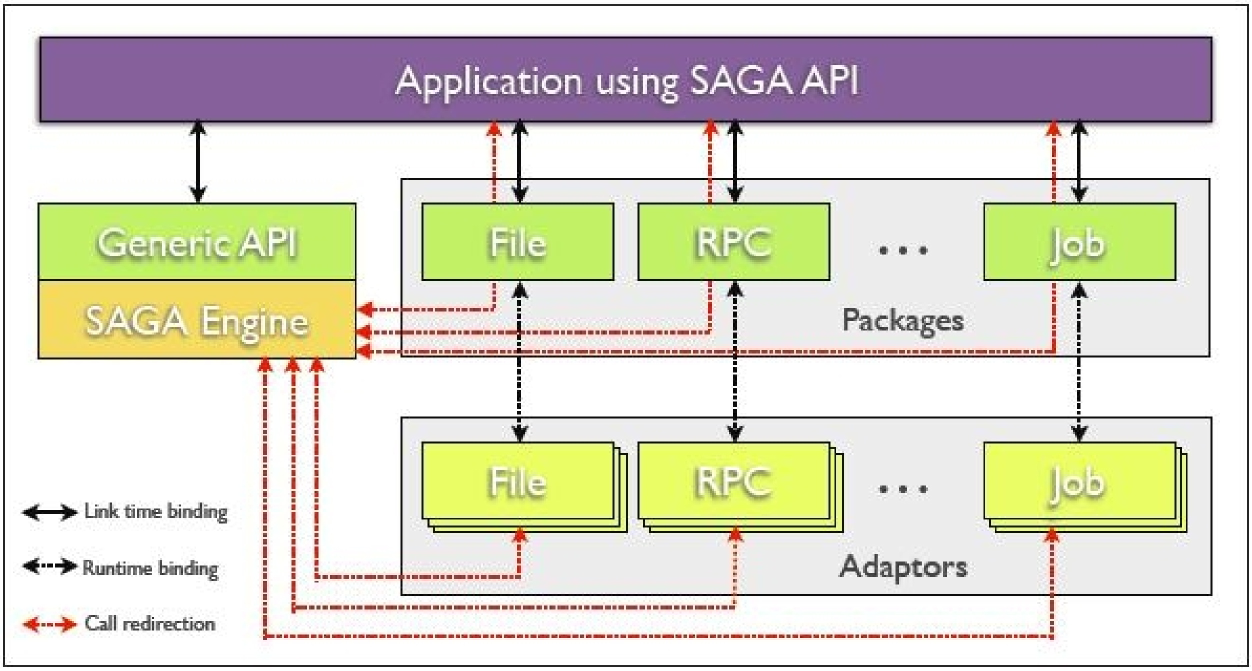


Figure PC-17. SAGA architecture: A lightweight engine dispatches SAGA calls do dynamically loaded middleware adaptors.

**Design Objectives**

Although SAGA by definition is intended to be simple for application developers, this doesn't imply that the implementation itself has to be simple. Logic and functionality built into the SAGA library core provide common functionality that can be extended through minimal effort. Ideally, adding a new API class is orthogonal to all other properties of the implementation, and also immediately benefits from those.The library is also designed to be easy to build, use, and deploy. As described above, a SAGA implementation must cope with a multitude of different dynamic requirements. A major design objective was to maximize decoupling of different components of the developed library to provide as much flexibility, adaptability and modularity as possible. The SAGA C++ Reference Implementation was also designed for maximum portability, anticipating use on different platforms and operating systems.

The SAGA C++ Reference Implementation library is divided into three dimensions, which are described below. These three dimensions are completely orthogonal — the user of the library may use and combine these freely and develop additional suitable components usable in tight integration with the provided modules.

*Horizontal Extensibility — API Packages*

The SAGA specification is object oriented and defines a set of API groups keeping objects of related functionality together (packages). The SAGA C++ Reference Implementation uses this functional grouping to define API packages. Current packages are: file management, job management, remote procedure calls, replica management, and data streaming. Each of these packages constitutes a separate and independent module. These modules depend only on the SAGA engine; the user is free to use and link only those modules actually needed by the application, minimizing the memory footprint. New API packages are expected to be added as the SAGA specification evolves. Adding new packages is straightforward due to the fact that all necessary common operations (such as adaptor loading and selection, or method call routing) are imported from the SAGA engine.

*Vertical Extensibility — Middleware Bindings*

A layered architecture allows for vertical decoupling of the SAGA API from the middleware. Separate adaptors, either loaded at runtime or pre-bound at link time, dispatch the various API function calls to the appropriate middleware. Usually there will be a separate set of adaptors for each type of supported middleware. These adaptors implement a well-defined Capability Provider Interface (CPI) and expose that to the top layer of the library, which makes it possible to switch adaptors at runtime and switch between different (and even concurrent) middleware services providing the requested functionality. The top library layer dispatches the API function calls to the corresponding CPI function. It also contains the SAGA engine module, which implements:

- core SAGA objects such as session, context, task or task container — these objects are responsible for the SAGA look and feel, and are needed by all API packages,
- common functions to load and select matching adaptors, to perform generic call routing from API functions to the selected adaptor, to provide necessary fall back implementations for the synchronous and asynchronous variants of the API functions (if these are not supported by the selected adaptor).

The dynamic nature of this layered architecture enables easy future extensions through the addition of new adaptors, helping to cope with emerging grid standards and new grid middleware.

*Extensibility for Optimization and Features*

Many features of the engine module are implemented by intercepting, analyzing, managing, and rerouting function calls between the API packages (where they are issued) and the adaptors (where they are executed and forwarded to the middleware). To generalize this management layer, a PIMPL (Private Implementation) idiom was chosen, and is rigorously used throughout the SAGA implementation.

This PIMPL layering allows for a number of additional properties to be transparently implemented, and experimented with, without any change in the API packages or adaptor layers. These features include:

- generic call routing
- task monitoring and optimization
- security management
- late binding
- fallback on adaptor invocation errors
- latency hiding mechanisms

These features can essentially be decoupled from the API and the adaptors because these properties affect only the IMPL side of the PIMPL layers. Firstly, the private implementation classes all inherit from the same base class but that base class is handled in the central engine module, so the engine can automatically cope with new API packages and adaptors. Secondly, all method calls are also handled generically in the engine, which is loosely coupled to both the API and adaptor layers. Any changes to the engine, all optimization, latency hiding techniques, monitoring features etc. can be implemented in the engine generically, and are orthogonal to the API and adaptor extensions. Hence, the extensibility of the engine represents the third orthogonal axis in the libraries extensibility scheme.

### Uniform for Programming Languages

The SAGA API specification is language-independent, however, the goal is to define language bindings that provide both a language-native look and feel to the API user, and strive for syntactic and semantic similarity over all SAGA language bindings. One of the consequences of this goal is that the API specification does not use language specific constructs, for instance C++ templates, which are thought to be too difficult to express uniformly over many languages. Also, the specification tries to be concise about object state management, and expresses semantics for shallow and deep copies. The SAGA C++ Reference Implementation follows the SAGA API specification closely in this respect. It is designed to accommodate wrappers in other languages, to provide the same semantics, and similar look and feel to other language bindings. A Python wrapper is currently developed and in alpha status, and there are plans to add similar thin wrappers to provide bindings to C, FORTRAN, Perl, and possibly others. From another point of view, it is extremely convenient to be able to implement adaptors in different languages. The Grid Application Toolkit (GAT, [15]), a C-based API predecessor of SAGA, already allows adaptors in different languages, and similar mechanisms may be implemented to allow Python or C based adaptors as well. In particular, Python based adaptors have been extremely useful for rapid prototyping of middleware bindings for GAT.

### Generic with Respect to Middleware, and Adaptable to Dynamic Environments

The dynamism of grid middleware has already been mentioned as a central dominating property of grid environments. This is addressed in the SAGA C++ Reference Implementation by the described adaptor mechanism that binds to diverse middleware. Additionally, late binding, fall back mechanisms, and flexible adaptor selection allow for additional resilience against a dynamic and evolving run time environment. It is noted, however, that adaptors need to deploy mechanisms like resource discovery and to implement fully asynchronous operations, if the complete software stack is to be able to cope with dynamic grids. SAGA implementation usability can be severely impacted if the quality of adaptors undermines the libraries mechanisms.

### Modularity makes the Implementation Extensible

We have described how the SAGA C++ Reference Implementation will be able to cope with the expected evolution and extension of the SAGA API. Further, the adaptor mechanism allows for easy extensions of the library to provide additional middleware bindings. In fact, the major future work for this SAGA implementation will be to provide multiple sets of stable adaptors for the major grid environments. This task, however, requires considerably more effort than the implementation of the present library and it is hoped that grid middleware vendors will be motivated to support and maintain these adaptors. Ideally, middleware vendors will implement adaptors for SAGA and deliver them as part of their client side software stack in the same way that they provide MPI implementations. This would be a major step towards wide spread adoption and benefit to grid applications.

### Portability and Scalability

Heterogeneous distributed systems naturally require portable code bases. The SAGA C++ Reference Implementation library strictly adheres to the C++ standard and portable libraries. To further insure compatibility, the library is developed on Windows and Linux concurrently as the two major target platforms. Problems on other platforms are also not expected, however, it should be noted that the portability of the SAGA implementation depends on the portability of the adaptors, and thus on the portability of the grid middleware client interfaces, which can be a much greater problem when compared to the library code itself.

Distributed applications are often sensitive to scalability issues too, particularly with respect to remote communications. As SAGA introduces a number of communication mechanisms, scalability concerns are naturally raised in respect to SAGA implementations. First, the SAGA API is not targeting high performance communication schemes, but tries to utilize simple communication paradigms. In no sense, does SAGA intend to replace MPI or other distributed communication libraries. Having said that, the design allows for zero-copy implementations of the SAGA communication APIs and also for fast asynchronous notification on events. Both of these are deemed critical for implementing scalable distributed applications.

### Simplicity for the End User

SAGA is designed to be simple to use. However, simplicity in use of an API is not only determined by the API specification, but also by its implementation. Characteristics that need attention while implementing the SAGA API include simple deployment and configuration, resilience against lower level failures, adaptability to diverse environments, stability, correctness, and peaceful coexistence with other programming paradigms, tools and libraries.. It is a challenge to keep a library implementation simple, with readable code but a modular approach helps. For example, it is simple to hide the generic call routing or the adaptor selection in the engine module since these features are not usually exposed to the user or adaptor programmer. However, modeling these central properties as modules can significantly increase the readability and maintainability of the code. Due to its notion of asynchronous operations, or tasks, the SAGA API implicitly introduces a concurrent programming model, The C++ language binding of the API allows for combination of that model with arbitrary mechanisms for managing concurrent program elements (i.e. to ensure object state consistency in all circumstances, to ensure thread safety, and to allow for application level semaphores and mutexes).

More information about the SAGA C++ Reference Implementation (currently being developed at the Center for Computation and Technology at the Louisiana State University) and various aspects of grid enabling toolkits is available on the SAGA implementation home page [20]. There you also will find additional information with regard to different aspects of grid enabling toolkits.

## Programming examples

As with any complex task, of course, the best way to learn is by doing and, in conjunction with that, examining how others have approached and handled common situations. In future versions of this Cookbook, we hope to include here several examples of code and scripts to illustrate common programming techniques and some that can also serve as building blocks to be adapted and customized towards specific applications. Some topics we will be looking to cover include:

- Performing grid operations
- Grid-enabled applications
- OpenMP and MPI
- VO and Experiment implementation examples

If you have expertise in any of these or related areas, or experience in successful grid application programming, and are willing to contribute explanations, working examples, code snippets, or "hint & tips", please contact the co-editor s to let us know!

# Bibliography

[1] Globus home page (http://www.globus.org/)
[2] Condor GT 4.0 Pre WS GRAM (http://www.globus.org/toolkit/docs/4.0/execution/prewsgram/)
[3] Unicore home page (http://www.unicore.org/)
[4] SOAP (Simple Object Access Protocol) (http://www.w3.org/TR/soap/)
[5] SDL (Specification and Description Language) (http://www.sdl-forum.org/)
[6] GT 4.0 GridFTP (http://www.globus.org/toolkit/docs/4.0/data/gridftp/)
[7] GRAM (GT 4.0 Pre WS GRAM) (http://www.globus.org/toolkit/docs/4.0/execution/prewsgram/)
[8] W3 (World Wide Web Consortium) (http://www.w3.org/)
[9] WSDL (Web Services Description Language) (http://www.w3.org/TR/wsdl)
[10] WSRF (Web Services Resource Framework) download (http://www.oasis-open.org/committees/download.php/16654/wsrf-cs-01.zip)
[11] SOAP (Simple Object Access Protocol) (http://www.w3.org/TR/soap/)
[12] Condor home page (http://www.cs.wisc.edu/condor/)
[13] Unicode home page (http://www.unicore.org/)
[14] abstraction layer (Wikipedia definition) (http://en.wikipedia.org/wiki/Abstraction_layer)
[15] GAT (Grid Application Toolkit and Testbed) (http://www.gridlab.org/wp-1)
[16] SAGA (Simple API for Grid Apps) (https://forge.gridforum.org/projects/saga-rg/)
[17] OGF (Open Grid Forum) (http://www.ogf.org)
[19] Gridlab (http://www.gridlab.org)
[20] SAGA implementation home page (http://fortytwo.cct.lsu.edu:8000/SAGA)
[21] SAGA C++ reference implementation (http://www.cct.lsu.edu/projects/Grid+Application+Toolkit)
[22] Monitoring and Discovery System (http://www.globus.org/toolkit/mds/)
[23] Grid Laboratory Uniform Environment (http://forge.gridforum.org/sf/projects/glue-wg)
[24] DataTAG (http://datatag.web.cern.ch/datatag/")
[25] GridForgea (http://forge.gridforum.org/sf/sfmain/do/home)
[26] A Globus Primer (http://www.globus.org/toolkit/docs/4.0/key/GT4_Primer_0.6.pdf)
[27] MDS web pages (http://www.globus.org/mds)
[28] Globus Monitoring and Discover (2005 Globus World) (http://www.globus.org/toolkit/presentations/GlobusWorld_2005_Session_9c.pdf)
[29] GridLab (http://www.gridlab.org/)
[30] iGrid (http://sara.unile.it/%7Ecafaro/software.html)
[31] GridShib website (http://gridshib.globus.org/about.html)
[32] Shibboleth (http://shibboleth.internet2.edu/)
[33] Internet2 (http://www.internet2.edu/)
[34] GridShib Technical Overview (http://grid.ncsa.uiuc.edu/presentations/gridshib-tech-overview-apr06.ppt)

[35] OpenPBS (http://www-unix.mcs.anl.gov/openpbs/)
[36] LSF (http://www.platform.com/Products/Platform.LSF.Family/)
[37] LoadLeveler (http://www-128.ibm.com/developerworks/eserver/library/es-loadlevel/index.html)
[38] Maui (http://www.clusterresources.com/pages/products/maui-cluster-scheduler.php)
[39] Moab (http://www.clusterresources.com/pages/products/moab-cluster-suite.php)
[40] Globus Resource Allocation Manager (http://tinyurl.com/2shtao)
[41] Torque (http://www.clusterresources.com/pages/products/torque-resource-manager.php)
[42] e-Compute (http://www.altair.com/software/ecompute.htm)
[43] Nordugrid (http://www.nordugrid.org/)
[44] Condor User Tutorial,^KUK Condor Week, ^KNeSC,^KOctober, 2004 (http://www.nesc.ac.uk/talks/438/11th/user_tutorial.ppt)
[45] Condor manual (http://www.cs.wisc.edu/condor/manual/v6.4/)
[46] AIST Grid Scheduling System (http://www.aist.go.jp/aist_e/aist_today/2006_20/hot_line/hot_line_21.html)
[47] NAREGI GridVM (http://tinyurl.com/24nenc)
[48] Keahey's Virtual Workspace (http://workspace.globus.org/papers/)
[49] Globus Toolkit GRAM (http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=4045)
[50] PBSPro (http://www.altair.com/software/pbspro.htm)
[51] GridFTP (http://www.globus.org/toolkit/docs/4.0/data/gridftp/)
[52] Reliable File Transfer (http://www.globus.org/toolkit/docs/4.0/data/rft/)
[53] GT 4.0 RFT Command Reference (http://www.globus.org/toolkit/docs/4.0/data/rft/RFT_Commandline_Frag.html)
[54] GT 4.0 RLS (http://www.globus.org/toolkit/docs/4.0/data/rls/)
[55] Network Storage Technology (http://www.cs.wisc.edu/condor/nest/)
[56] Flexibility, Manageability, and Performance in a Grid Storage Appliance (http://www.cs.wisc.edu/condor/nest/papers/nest-hpdc-02.pdf)
[57] SRM/DRM (https://twiki.grid.iu.edu/twiki/bin/view/Integration/SrmDrm)
[58] srmcp (https://twiki.grid.iu.edu/twiki/bin/view/Documentation/StorageSrmcpUsing)
[59] Gratia twiki page (https://twiki.grid.iu.edu/twiki/bin/view/Accounting/WebHome)
[60] Full Project Definition (https://twiki.grid.iu.edu/twiki/bin/viewfile/Accounting/WebHome?filename=AccountingProjectDefinition1.doc)
[61] SweGrid Accounting System (http://www.sgas.se/)
[62] SGAS Download (http://www-unix.globus.org/toolkit/docs/4.0/techpreview/sgas/)
[63] SGAS Installation and Administration Guide (http://www.sgas.se/docs/SGASInstallConfig.pdf)
[64] SGAS Administration Guide (http://www.sgas.se/docs/SGASAdmin.pdf)
[65] OSG Gratia Project (https://twiki.grid.iu.edu/twiki/bin/view/Accounting/WebHome)
[66] Directed Acyclic Graph Manager (DAGman) (http://www.cs.wisc.edu/condor/dagman/)
[67] Condor manual (http://www.cs.wisc.edu/condor/manual/v6.4/)
[68] Swift (http://www.ci.uchicago.edu/swift/index.php)
[69] GriPhyN Virtual Data System (http://www.griphyn.org/news/index.html)
[70] A Swift Tutorial (http://www.ci.uchicago.edu/swift/guides/tutorial.php)
[71] The SwiftScript User Guide (http://www.ci.uchicago.edu/swift/guides/userguide.php#engineconfiguration)
[72] Swiftscript Language Reference Manual (http://www.ci.uchicago.edu/swift/guides/languagespec.php)
[73] Planning for Execution in Grids (http://pegasus.isi.edu/)
[74] GriPhyN Virtual Data System Quick Guide (http://pegasus.isi.edu/docs/QuickGuide.pdf)
[75] Security Assertion Markup Language (SAML) (http://xml.coverpages.org/saml.html)
[76] Open Grid Forum Security groups (http://www.ogf.org/gf/group_info/areasgroups.php?area_id=7)
[77] International Grid Trust Federation (IGTF) — Grid;s Policy Management Authority (http://www.gridpma.org/)

# Joining a Grid: Procedures & Examples

## Introduction

One the of most effective ways to become familiar with the ins and outs of grid technology and usage is to join a grid initiative with goals and objectives that encompass or overlap with those of your institution, and with opportunities to develop peer collaboration and support. Through participation in such initiatives, you can leverage shared resources to meet institutional goals and begin contributing your perspective and increasing expertise back to the community for the collective improvement and advancement of effective use of grid technology. Several grid initiatives invite this type of participation today; two notable examples are described below.

## SURAgrid: A regional-scale multi-institutional grid

SURAgrid is a consortium of organizations collaborating and combining resources to help bring grid technology to the level of seamless, shared infrastructure. The project arose from the desire for ongoing collaboration among institutions that had been participating with SURA (Southeastern Universities Research Association) in NSF Middleware Initiative (NMI) Integration Testbed, a program that provided comprehensive evaluation of NMI middleware in the earliest years of that initiative. Facilitated by SURA, the vision for SURAgrid is to orchestrate access to a rich set of distributed capabilities in order to meet diverse users' needs. Capabilities to be cultivated include locally contributed resources, project-specific tools and environments, highly specialized or HPC access, and gateways to national and international cyberinfrastructure.

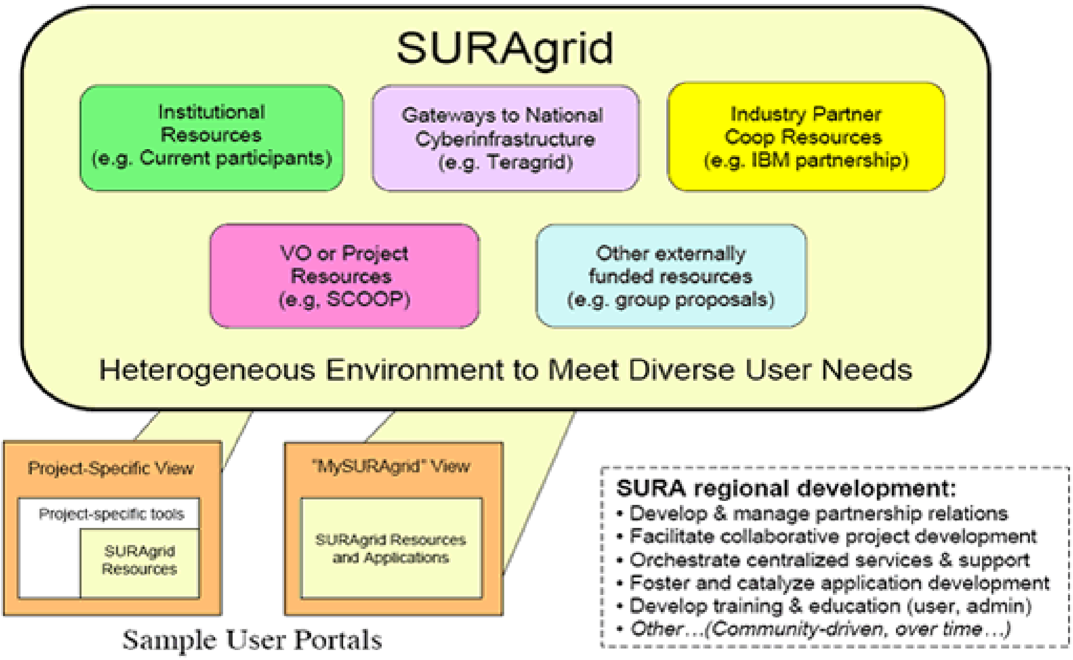

SURAgrid
Institutional Resources (e.g. Current participants)
Gateways to National Cyberinfrastructure (e.g. Teragrid)
Industry Partner Coop Resources (e.g. IBM partnership)
VO or Project Resources (e.g, SCOOP)
Other externally funded resources (e.g. group proposals)
Heterogeneous Environment to Meet Diverse User Needs
Project-Specific View
Project-specific tools
SURAgrid Resources
"MySURAgrid" View
SURAgrid Resources and Applications
Sample User Portals
SURA regional development:
• Develop & manage partnership relations
• Facilitate collaborative project development
• Orchestrate centralized services & support
• Foster and catalyze application development
• Develop training & education (user, admin)
• Other…(Community-driven, over time…)

Figure JG-1. An overview of SURAgrid.

To meet the needs of its broad participant and user community, SURAgrid focused on three primary goals:

- Develop a scalable infrastructure that leverages local institutional identity and authorization while managing access to shared resources across institutional boundaries.
- Promote the use of this infrastructure for the broad research and education community, creating a whole that is greater than the sum of its parts.
- Provide a forum for participating institutions to gain additional experience with grid technology and to promote collaborative project development.

With the long-term view of grids as generalized infrastructure, an emphasis on diversity and inclusion, and a persistent objective to discover and understand grid use outside the scope of expected or typical use today, SURAgrid is positioned to become an essential tool to build the scientific and educational capacity of the Southeastern U.S. and beyond.

## Applications on SURAgrid

The identification of research applications that can be significantly advantaged through the application of grid technologies is a key factor in fostering grid development and deployment and also a key factor to grow and sustain SURAgrid. The deployment of an intentionally diverse set of applications is contributing to the advancement of research and education within a variety of disciplines. Applications under development on SURAgrid are detailed on the SURAgrid Web site, with a few notable examples listed below:

- *SCOOP (SURA Coastal Ocean Observing & Prediction) Coastal Ocean Modeling)*
  The SCOOP (http://scoop.sura.org) Cyberinfrastructure (CI) is being developed to support coastal research and operations, by providing a modular, distributed system for real time prediction and visualization of the impacts of extreme atmospheric events on coastal areas, and enabling advances in multi-scale, multi-model, and DDDAS science. The SCOOP CI enables complex workflows, which integrate coastal models such as ADCIRC, ELCIRC, WW3, and CH3D with various wind models and sensor information. SCOOP presently uses SURAgrid resources for added computational capacity.

- *EPANET Simulation-Optimization for Threat Management in Urban Water Systems*
  This application incorporates dynamic demand data, in real-time, into a simulation-optimization process for contamination threat management in drinking water distribution systems. The nature of this work is highly compute-intensive and requires multi-level parallel processing via computer clusters and high-performance computing architectures such as SURAgrid. Simulation-Optimization with EPANET is part of a multi-disciplinary, three-year NSF-funded DDDAS (Dynamic Data-Driven Application Systems) research project to develop a cyberinfrastructure system that will both adapt to and control changing needs in data, models, computer resources and management choices facilitated by a dynamic workflow design. Project Partners: North Carolina State University; University of Chicago; University of Cincinnati; University of South Carolina.

- *Grid-Enabled Distributed BLAST*
  BLAST is a database search application for matching protein and nucleotide sequences. Maximizing the throughput of searches is key to improving research results. This distributed implementation of BLAST developed by the University of Alabama at Birmingham uses the DynamicBLAST Meta-scheduler to select appropriate grid resources for select query strings. Globus is used for job staging, submission and retrieval. ncbiBLAST performs the computations. Jobs are submitted using a web-based interface that leverages campus identity credentials via Pubcookie and manages grid authentication on behalf of the user via MyProxy, providing a simplified user authentication experience.

- *SURAgrid Teaching Environment*
  Effective teaching about grids, within Computer Science as well as other disciplines, is greatly enhanced by students and instructors having hands-on access to a stable grid environment. Through coordinated commitment, operation and support across a subset of SURAgrid resources, SURAgrid is developing a predictable, secure and reliable grid-based teaching facility for use by SURAgrid sites in their grid course development and/or delivery. Old Dominion University has made initial use of this capability by providing basic grid access for students to supplement theory in a Distributed Computing course during Spring 2007. Targeted improvements include more scalable group account management, accommodation for varying levels of access, and space for faculty to participate in joint course development.

## How SURAgrid works

As an inter-institutional grid infrastructure, SURAgrid provides a variety of application users with a common point of access to a shared set of distributed, heterogeneous resources. As of August 2007, thirty academic organizations and institutions are participating in SURAgrid. Most but not all are members of SURA, athough SURA membership is not a requirement.

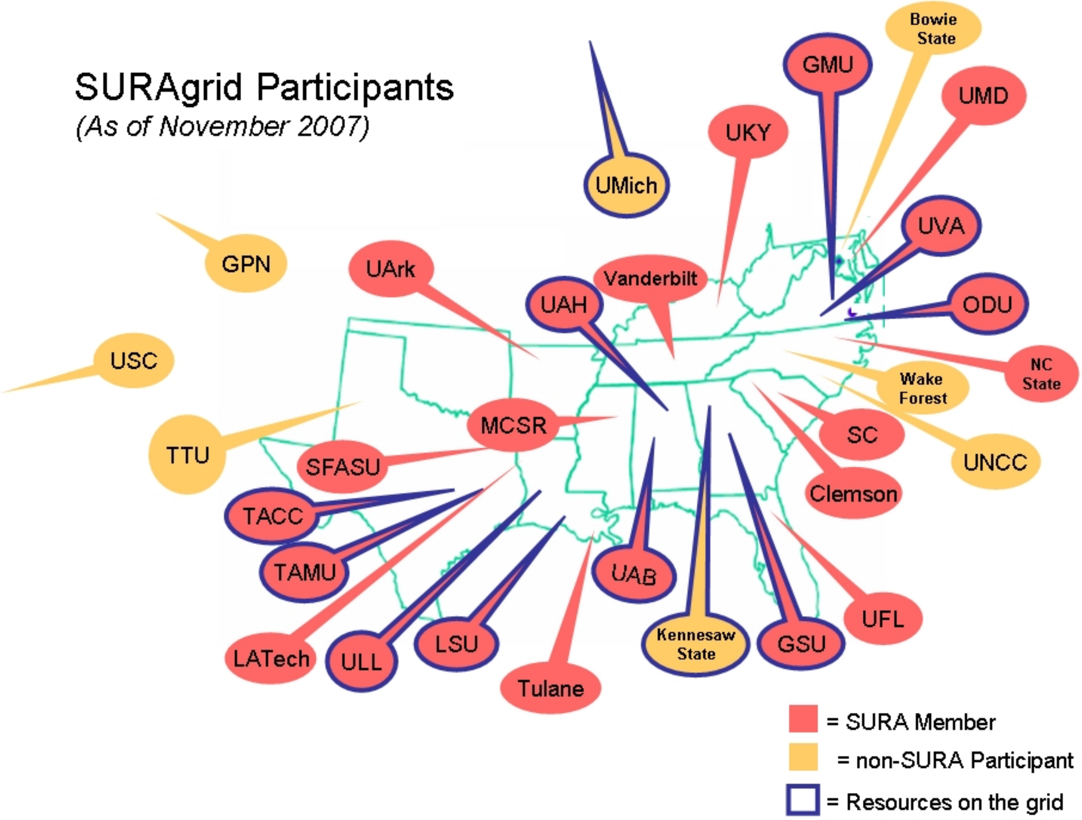


Figure JG-2. The SURAgrid map.

Resources to be shared are contributed by the participating organizations and remain under autonomous control of the resource owner, with shared access enabled through grid-wide coordination of authentication and authorization mechanisms, and operational procedures.

Most of the resources being contributed are computational in nature, providing just over 10.5 Teraflops of combined capability as of April 2007, for sharing among the SURAgrid community (although capacity does fluctuate as resources are added, swapped, upgraded, etc.). More diverse resources such as databases, instruments, storage, and application services are anticipated in the future. SURAgrid resources can be viewed and accessed through the SURAgrid portal at https://gridportal.sura.org, which is maintained by the Texas Advanced Computing Center (TACC).

Institutions that participate in SURAgrid are also expected to share in its organization, planning and development, as a cooperative effort to foster collaboration and build a shared asset to help meet local, regional and national goals for the advancement of science through grid technologies. Informal and "grass roots" structure and procedures for governance and decision-making are gradually being replaced by more formal components and processes, while retaining the spirit of community and collaboration that is the foundation of the initiative. New organizations can join SURAgrid by following the contact process detailed on the SURAgrid Web site [1], see tab/menu item "Join SURAgrid".)

## The SURAgrid infrastructure

The SURAgrid software stack, grid services and application environment have evolved to include a minimal set of requirements and recommendations intended to be as loose as possible while providing a foundation of interoperability. Originally, the primary need was for management and coordination of resources and applications within SURAgrid itself. More recently, grid-to-grid integration has become of greater importance to SURAgrid participants who need to share resources with or bridge access to other grid projects such as TeraGrid, Open Science Grid, TIGRE (Texas Internet Grid for Research and Education), and project-specific grids such as GridChem.

SURAgrid presently uses Globus middleware to facilitate access to a variety of computational resources, such as Linux-based clusters, IBM P575 HPC systems, Condor pools, and virtualized resources. Adding resources to SURAgrid is facilitated through user documentation, peer support and some direct assistance from SURA staff.

Continued development of the SURAgrid environment is required to accommodate new user communities, future integration with other grid initiatives, and an anticipated increase in corporate partnerships (such as the SURAgrid-IBM partnership [2]. The SURAgrid stack specification is in alignment with middleware that is in use by these and many other major academic grid initiatives. Different applications may have requirements beyond the currently specified software stack. Such requirements are treated as application-specific needs until they are shown to be more commonly required and so should be incorporated into overall resource requirements.

SURAgrid resource requirements & recommendations (server side) as of June 2007:

- Required: Globus 4.x, WS-GRAM, gridFTP, WS-MDS and RFT. Pre-WS GRAM and MDS are strongly recommended to support existing legacy applications.
- GSI-OpenSSH is strongly recommended for application staging. If enabled, it is required that you advertise the port through WS or system detail in the SURAgrid portal. We recommend using either port 22 or 2222.
- Any version of operating system that supports the required services above, with Linux 2.4 or higher recommended in order to provide a common platform for application development.
- Addition of resource and relevant system detail to the resource monitor (GPIR) of the SURAgrid portal.
- A scheduler installed as part of your underlying resource configuration.
- Cross-certification with SURAgrid Bridge CA — strongly recommended at this time and likely to be required in the future. (See https://www.pki.virginia.edu/nmi-bridge [3])
- Configuration of the required environment variables as defined in SURAgrid Environment Variables. Configuration of the optional environment variables also recommended.

Users must be both authenticated and authorized to access SURAgrid resources. The Globus GSI (Grid Security Infrastructure) relies on PKI (public key infrastructure) and its related exchange of certificates for authentication and provides for authorization through a "grid-mapfile" that associates identities with individual system accounts. SURAgrid augments this authentication process by leveraging authoritative campus identity management where possible for user authentication between participating sites. Scalable exchange of this trusted information is enabled through the use of the SURAgrid Bridge Certificate Authority (Bridge CA), maintained by the University of Virginia, SURAgrid's lead in this area. Each site establishes a trusted relationship with the SURAgrid Bridge CA, which essentially then "vouches" for each site to the others. In the absence of a Bridge CA, each site within a PKI infrastructure must establish a trusted relationship with all the other participating sites, which can become exceedingly difficult, if not impossible, to manage effectively as the number of participants increase. Within SURAgrid, participating sites typically run their own Certificate Authority (CA) to provide both user and system certificates for participation in the SURAgrid PKI. A SURAgrid CA is also under development, to provide certificates for sites that are not running their own CA or do not have access to one, SURAgrid guest access, etc.

Once a SURAgrid user is successfully authenticated, he or she accesses SURAgrid resources through use of a pre-established individual SURAgrid user account. This account is recognized on all properly configured SURAgrid resources and the permissions inherent in the user account determine the levels of authorization (what the user is able to do). The setup and management of SURAgrid user accounts is facilitated through several tools developed for SURAgrid by the University of Virginia. These tools include Web-based account management, a shared LDAP directory that maintains SURAgrid user information, and scripts that provide various levels of automation to be used for mapping user information to the Globus GSI, to the degree desired by each site. Account access mechanisms in use on SURAgrid range from user access through the SURAgrid portal, remote login by the user to individual resources, and software-automated access through applications and scripts.

## Implementation closeup: Installing the SURAgrid server stack

### SURAgrid Server Software Stack

To accommodate heterogeneity, the SURAgrid software stack, grid services and application environment evolve based on setting a minimal set of requirements and recommendations that increase in specificity as needs dictate. However, SURA has defined a common set of software that should be available on all SURA server systems at this point in time, to insure interoperability among systems and support for the current and near-term application set. To facilitate installation of the appropriate software, the SURAgrid team is collaborating with the TIGRE [4] project in the development of a "one-button" installation with stack for SURAgrid. This installation package includes both services and clients for those services, and leverages the Virtual Data Toolkit [5] (VDT) to provide a convenient way to install and configure this software. The excerpt below illustrates parts of this automated process for adding a resource to SURAgrid.

Please check the official SURAgrid Server Stack website [6] for the most current material.

### Contents

The SURAgrid software stack consists of the following components:

- Globus Toolkit 4.0 [7] (servers and clients)
- Grid Proxy programs. For obtaining X.509 credentials.
- Pre-WS and WS-GRAM. The GRAM2 (pre-web services) and Gram4 (web services) Globus client and server components. These components provide remote job submission. Also included are supporting services such as the Reliable File Transfer Service and the Delegation Service.
- GridFTP. GridFTP server and clients that provide secure, high-bandwidth file transfers.
- GSI OpenSSH [8]. Provides ssh access to SURAgrid systems using grid credentials.
- UberFTP [9]. An interactive command line client for GridFTP.
- MyProxy [10] client. One way for caching proxies obtained from grid credentials.

- Condor-G [11]. Job submission and management.

**Requirements**

VDT supports a variety of operating system and OS versions. Please make sure your platform is one of the supported operating systems [12]. The SURAgrid software stacks require the following underlying software to be available:

- Perl 5.8.0 or greater
- tar (any version)
- diff+patch (any recent version should suffice)
- Python 2.2 or greater (pacman itself will install if necessary)

The disk space requirements vary per platform but generally 1-2 GB of free disk space will suffice.

**General Preparations and Steps**

The basic steps in an installation scenario include:

- Install pacman

  The SURAgrid software stack is installed and managed with pacman [13]. pacman is a utility which manages software packages in Linux. It uses simple compressed files as a package format, and maintains a text-based package database (more of a hierarchy), just in case some hand tweaking is necessary.
- Install the SURAgrid server software stack

  The root directory, where the SURAgrid server software stack is installed, is created. Then pacman is used to begin installing the server software stack. pacman asks you questions and downloads a relatively large number of packages.
- Configure the SURAgrid Software Stack

  After the installation is complete, there are a number of post-install configuration steps to perform before the SURAgrid server software is fully functional. They include:

  - ♦ *Install Credentials*

    Credentials are required for your host. The SURAgrid authentication and authorization infrastructure is based on a two-tier PKI approach coupled with an optional LDAP-based PAM callout. See the SURAgrid PKI Bridge Certification Authority and User Management System [14] pages for more information.
  - ♦ *Map SURAgrid Users*

    You have the option of either using the SURAgrid LDAP callout to control local account mapping and authorization or simply setting up a grid-mapfile to do so. Such mapping is required to associate the subject distinguished name for a particular user in their X.509 certificate to a local Unix account. For further information, see the grid-mapfile section [15] of the SURAgrid PKI Bridge Certification Authority and User Management System pages. (Note that this topic is undergoing some evolution within SURagrid, and we hope in particular to provide a mechanism for a fully-accredited approach in cooperation with the International Grid Trust Federation [16] soon.)
  - ♦ *Configure GSISSH*

The GSI version of SSH [17] allows users to ssh into SURAgrid system using their grid credentials. They do not have to provide a password and are automatically mapped to their local assigned grid userid upon gsissh login. To set this up, enable your gsissh server [18].

- *Configure GridFTP*

  The Globus GridFTP provides high-performance file movement between SURAgrid systems.

- *Configure Globus account*

  For added security, the Globus web services container should be run as an ordinary user. The typical userid used is 'globus'.

- *Configure WS-GRAM*

  The Web Services GRAM (WS-GRAM) component allows remote users to execute applications on each SURAgrid system.

- *Set special local conditions*

  The globus tcp port range can be set to match local preferred values, as well as any other special local conditions for your installation.

  Other commands needed to handle variations that apply to a local cluser environment may also be added.

- *Start services*

  Globus services can now be started, including WS-GRAM.

**To get help**

If you have any problems installing the SURAgrid software, please contact the SURAgrid Support e-mail list [19]. There is a VDT Discuss and Announce list which might be helpful for specific advanced usage scenarios. Please see the VDT Support page [20] for a addtional information.

# The Open Science Grid

(The following Open Science Grid example borrows liberally from the information available through the OSG website [21].)

The Open Science Grid, formed in 2004 is a distributed computing infrastructure for large-scale scientific research, built and operated by a consortium of universities, national laboratories, scientific collaborations and software developers. The goal of the OSG Consortium to enable diverse communities of scientists to access a common grid infrastructure and shared resources,

The OSG is supported by the National Science Foundation and the U.S. Department of Energy's Office of Science.

Members of the OSG Consortium [22] contribute effort and resources to the OSG infrastructure and reap the benefits of a shared infrastructure that integrates computing and storage resources from more than 50 sites in the United States, Asia and South America. OSG also has partners [23], including campus, regional, national and international grids.

Researchers from many fields [24], including astrophysics, bioinformatics, computer science, medical imaging, nanotechnology and physics use the OSG infrastructure to advance their research. OSG provides help for new communities to adapt their applications to use the distributed facility and make their resources accessible. The OSG also works to enable scientists to seamlessly harness grid-computing resources worldwide and interoperates with multiple other Grid infrastructures.

The OSG is a continuation of Grid3 [25], a community grid built in 2003 as a joint project of the U.S. LHC software and computing programs, the National Science Foundation's Grid Physics Network (GriPhyN) and International Virtual Data Grid Laboratory (iVDGL) projects, and the U.S. Department of Energy's Particle Physics Data Grid (PPDG) project.

The OSG includes two grids: an Integration Grid and a Production Grid. The Integration Grid is used to test new grid applications, sites and technologies, while the Production Grid provides a stable, supported environment on which researchers run their scientific applications. OSG partners, include campus, regional, national and international grids. The OSG also works to enable scientists to seamlessly harness grid-computing resources worldwide and interoperates with multiple other Grid infrastructures.

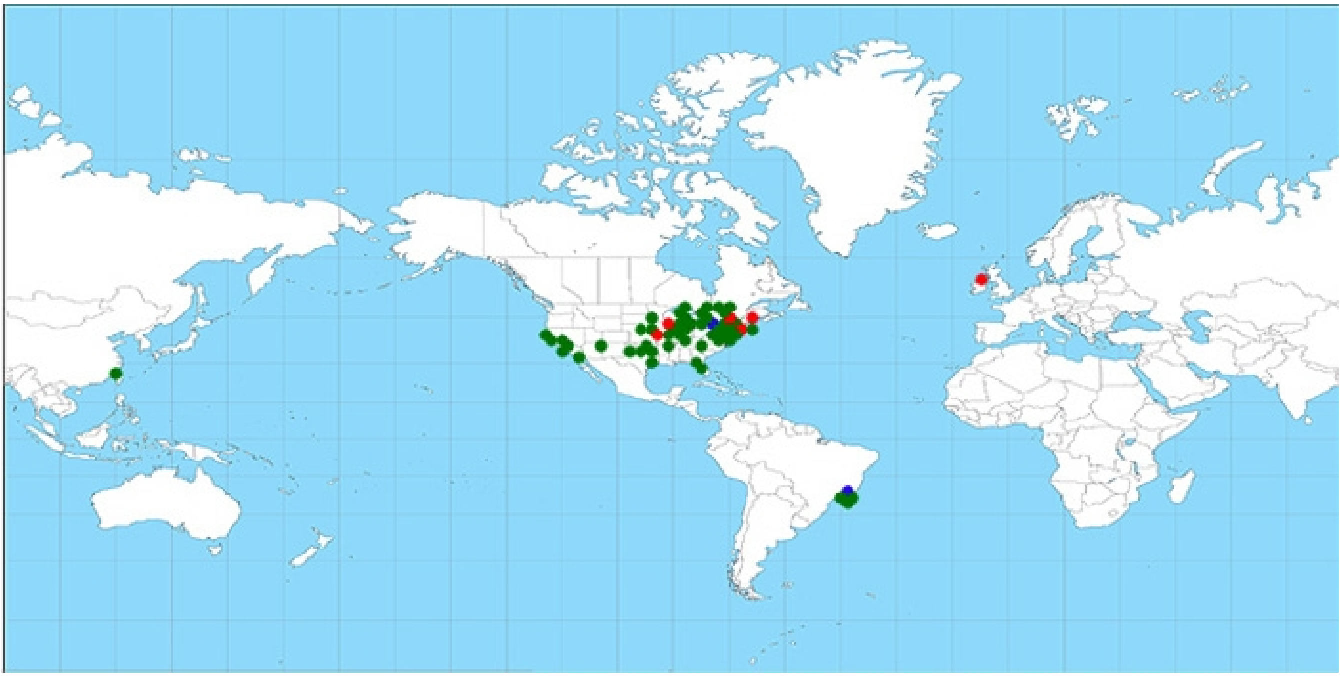


Figure JG-3. Location of the Open Science Grid Production Resources.

## Software

The Open Science Grid provides and supports a reference set of software called the "OSG Software Stack" for download and use by administrators and users of the OSG.

The software stack relies on the Virtual Data Toolkit (VDT) [26] middleware, which is itself a packaging and distribution based on the NSF Middleware Institute (NMI) [27] releases of Condor, Globus and other standard Grid middleware.

OSG@Work [28] pages provide detailed instructions on how to prepare a facility and/or resource and how to download and configure the OSG Software Stack in order to provide or access resources on the OSG.

A production release of the OSG Software Stack comes only after validation of the proposed software on the Integration Grid, and is based on released versions of the VDT.

## Applications on OSG

Scientists from many different fields use the Open Science Grid to advance their research. The OSG Consortium includes members from particle and nuclear physics, astrophysics, bioinformatics, gravitational-wave science and computer science collaborations. Consortium members contribute to the development of the OSG and benefit from advances in grid technology. Applications in other areas of science, such as mathematics, medical imaging and nanotechnology, benefit from the OSG through its partnership with local and regional grids or their communities' use of the Virtual Data Toolkit software stack.

The Consortium members contribute the resources available to the OSG. The owners of the resources control their use, with an expectation that 10-20% are on average available for opportunistic use, and with policies such that OSG members can use any unused cycles.

Thus the existence of the OSG does not obviate the need for the purchase of hardware and building of computational facilities by and for each science community. The scope of OSG is to:

- Enable scientists to use a greater % of the available compute and storage cycles.
- Help scientists to use distributed systems and software with less effort.
- Enable more sharing and reuse of software and reduce duplication through providing effort in integration and extensions.
- Establish an "open-source" community working together to communicate knowledge and experience and reduce overheads for new participants.

The benefits come from reducing risk in and sharing support for large, complex systems, which must be run for many years with a short lifetime workforce. And also from leveraging the expertise and support for such systems to enable new communities to more easily participate in distributed science including:

- Savings in effort for integration, system and software support,
- Opportunity and flexibility to distribute load and address peak needs.
- Maintenance of an experienced workforce in a common system.
- Lowering the cost of entry to new contributors.
- Enabling of new computational opportunities to communities that would not otherwise have access to such resources.

The deliverables and milestones of OSG are driven directly by the needs of the current set of scientific stakeholders and evolve through balancing of their needs and those of the new communities to the available effort.

The OSG Grid Operations Center at Indiana University provides front line support for all areas of the OSG and the OSG web site document repository and @Work Twiki site give a lot of information about all aspects of the facility. We will only touch on a few representative areas of activity here.

## Use of the OSG

There are more than sixty active computational sites on the OSG. The infrastructure supports job throughput of more than a hundred thousand CPUhours a day and supports several hundred users. About twenty of the sites are part of the US LHC distributed data handling and analysis systems (Brookhaven and Fermilab Tier-1, University Tier-2s). These sites are supporting ongoing data distribution at aggregate of tens of terabytes a day. Four sites are owned by LIGO and are being used for transitioning analysis codes from the existing LIGO data grid to full production on the common infrastructure. Four sites are owned (or partially owned) by STAR and are being used to bring STAR data distribution, simulation and production codes. The Tevatron experiments are also making good opportunistic use of the OSG

## Bringing new users onto the OSG

The Open Science Grid (OSG) engagement activity works closely with new user communities over periods of several months to bring them to production running. These activities include: providing an understanding of how to use the distributed infrastructure; adapting applications to run effectively on OSG sites; engaging in the deployment of community owned distributed infrastructures; working with the OSG Facility to ensure the needs of the new community are met; providing common tools and services in support of the new communities; and working directly with and in support of the new end users with the goal to have them transition to be full contributing members of the OSG. To date there are the following Engagement users:

- Adaptation and production running opportunistically using more than a hundred thousand CPUhours of the Rosetta application from the Kuhlman Laboratory in North Carolina across more than thirteen OSG sites which has resulted in structure predictions for more than ten proteins. We have so far tested the robustness of the system to the submission of up to about three thousand jobs simultaneously
- Production runs of the Weather Research and Forecast (WRF) application using more than one hundred and fifty thousand CPUhours on the NERSC OSG site at Lawrence Berkeley National Laboratory
- Improvement of the performance of the nanoWire application from the nanoHUB project on sites on the OSG and TeraGrid, such that stable running of batches of five hundred jobs across more than five sites is routine. Work was also done in support of nanoHUB scientists to use OSG resources to run BioMoca simulation jobs last year and the first couple months of this year.
- Production running using more than twenty thousand CPU hours of the CHARMM molecular dynamic simulation to the problem of water penetration in staphylococcal nuclease using the ATLAS workload management system, PANDA and opportunistically available resources across more than ten OSG sites.

## Sites and VOs

A Site is a set of processing and/or storage resources and/or services co-located and centrally administered. A Virtual Organization (VO) is an organization that includes people using the resources (users, developers, administrators, and managers), the services needed by the organization and the resources owned by the organization. The OSG architecture defines interfaces between sites and VOs to the common infrastructure. The OSG provides an integrated and tested reference set of software the Virtual Data Toolkit (VDT) for the sites and the VOs to use to interface to the OSG distributed facility. Both sites and VOs have responsibility and authority over the resources, software and services that they own and they control and mange their use.

The OSG implementation architecture is cognizant that any resource may be supporting use through multiple interfaces — from local submission and access, from the OSG, and from other similar infrastructures such as Campus Grids (e.g. FermiGrid) or other national infrastructures such as TeraGrid. Similarly, the implementation architecture is cognizant that any VO may be using multiple infrastructures simultaneously and may have a deep set of (sometimes complex) shared software and services that are specific to the VO and operate across these infrastructures. These are additional drivers to the model that sites retain local control and management for all use, services and processes, and that VOs have control and management over their internal processes, priorities and use. In addition to levels of service and resource use being agreed between resource owners and users, the site and VO processes are implemented to support sharing and opportunistic use of the resources accessible to the OSG.

## OSG services

The OSG provides common services across the distributed facility in support of VOs and Sites: monitoring, validation and information about the full infrastructure; tracking of any and all problems and ensuring they are resolved; the Virtual Data Toolkit software packaging and support; integration and testing facilities; security;

troubleshooting of the end-to-end system; and support for existing and new user communities. The OSG also provides effort to bring new services and software into the facility and to collaborate with the external projects, as well as documentation and training of site and VO administrators and users.

## Benefits from a common, integrated software stack

OSG software releases consist of the collection of software integrated and distributed as the Virtual Data Toolkit (VDT) with a thin layer of additional OSG specific configuration scripts. Modules in the VDT are included at the request of the stakeholders. The Condor and Globus software provide the base technologies. VDT includes about thirty additional modules, including components from other computer science groups, the Enabling Grids for EScience (EGEE), DOE Laboratory facilities (Brookhaven, Fermilab and LBNL), and the application communities themselves, as well as standard open source software components such as Apache and Tomcat. OSG also supports the VDT for external projects. For example, VDT is used by the EGEE and Australian distributed computing infrastructures. Additionally, in support of interoperability across their infrastructures, the OSG and TeraGrid software stacks include aligned versions of the Condor and Globus software.

The VDT provides a reference software stack for use by OSG sites. Once the software is installed a site supports remote job submission, shared storage at a site, data transfer between sites, has services to manage priorities and access between VOs, and can participate in the OSG monitoring, validation and accounting services. OSG supports use of the reference software. Sites must provide the common interfaces to OSG services but the actual implementation is not dictated. The VDT also provides client libraries and tools for the applications to use to access OSG resources and services.

## Operational security and the security infrastructure

OSG is well aware of the essential and integrated nature of security operations and management. We have comprehensive risk analysis and security plans. We respond to software security notifications by a prompt analysis of the problem and assigning high priority to patches and fixes. Over the past year we have had about ten such notifications which have resulted in new software, downloads within between a day and a week or two. The VDT, Condor, Globus, and EGEE software teams communicate security risks as soon as they are identified and work together on patches and solutions. The collaborative nature of the OSG means that communication is natural and happening all the time and security is part of the day-to-day normal processes. OSG has mechanisms to deny user's access to sites and resources. The grouping of users into VOs gives us a small number of well-identified responsible managers who control user entry to the infrastructure. This leads us to a model of trust between sites, VOs and the OSG, with delegated trust between the VO and the end users.

The OSG security infrastructure is based on: X509 user, host and service certificates gained through one of the International Grid Trust Federation accredited Certificate Authorities; user identity proxy certificates obtained through the VO Management Service (EGEE VOMS) which provides checking of the user as part of a VO; management of extended certificate attributes to assign "roles" to a particular access by a user; flexible definition of mapping of user certificates to accounts and ACLs (access control lists) on a site; and policy (including blacklist) enforcement points at the site (processing and storage) services themselves.

## Jobs, data, and storage

### Job Management and Execution

OSG sites present interfaces allowing remotely submitted jobs to be accepted, queued and executed locally. The priority and policies of execution are controlled both by the VO and the site itself. VO policies are defined through "roles" given to the user through the VOMS service. Site policies and priorities are defined through mapping the user and their roles to specific accounts used to submit the job to the batch queue. OSG supports the Condor-G job submission client which interfaces to either the pre-web service or web services GRAM Globus interface at the executing site. Job managers at the backend of the GRAM gatekeeper support job execution by local Condor, LSF, PBS, or SGE batch systems.

### Data Transport, Storage and Access

Many of the OSG physics user communities have large file based data transport and application level high data I/O needs. The data transport, access, and storage implementations on OSG take account of these needs. OSG relies on GridFTP protocol for the raw transport of the data — using Globus GridFTP in all cases except where interfaces to storage management systems (rather than file systems) dictate individual implementations. The community has been heavily involved in the early testing of new versions of Globus GridFTP as well as defining needed changes in the GridFTP protocol.

### Storage Resource Management

OSG supports the Storage Resource Management (SRM) interface to storage resources to enable management of space and data transfers to prevent unexpected errors due to running out of space, to prevent overload of the GridFTP services, and to provide capabilities for pre-staging, pinning and retention of the data files. OSG currently provides reference implementations of two storage systems the LBNL Disk Berkeley Storage Manager (BeStMan) [34] and dCache [35].

In addition, because functionalities to support space reservation and sharing are not yet available through grid interfaces, OSG defines a set of environmental variables that a site must implement and a VO can rely on to point them to available space, space shared between all nodes on a compute cluster, and for the use of high performance I/O disk caches.

## Gateways to other facilities and grids

We are seeing a rapid growth in the interest and deployment of shared computational infrastructures at the local and regional level. We are also seeing a rapid growth in research communities' needing to move data and jobs between heterogeneous facilities and build integrated community computational systems across high performance computing (HPC) facilities and more traditional computing clusters.

OSG provides interfaces to these HPC facilities to support these use cases. The OSG also federates with other large infrastructures — notably the TeraGrid and EGEE — by providing gateways between them and OSG, and supporting groups to submit jobs across and move data between them. For example, the OSG collects information from the resources and publishes them in the format needed by the EGEE.  The CMS VO "Resource Broker" job dispatcher then submits jobs transparently across EGEE and OSG resources.

## Participating in the OSG

New organizations contribute to OSG by providing resources, using the infrastructure, working with the communities to provide software, participating in training and documentation activities, and/or participating in the security, troubleshooting, or other OSG activity areas. The overhead to participation is low and the benefit is matched to the principle that "what you get out depends on what you put in". An organization registers with the Grid Operations Center and provides contact and planned usage information. The OSG staff then helps to interface the resources to the OSG and provides support for the VOs usage.

## Training on the OSG

The OSG Education and Training [29] program provides training for student users, researchers and educators of the OSG and site administrators.At the core of the student education program are the Workshops [30], organized by OSG and its partners. These grid schools give advanced undergraduate and graduate students a basic foundation in distributed computing and provide valuable hands-on training in distributed and grid computing techniques. The schools introduce essential skills that will be needed by students in the fields of natural and applied science, engineering and computer science to conduct and support scientific analysis in grid computing environments.

See OSG's Research Highlights [31] for more details.

# Bibliography

[1] SURAgrid Web site (http://www.sura.org/suragrid)
[2] SURAgrid-IBM partnership (http://www.sura.org/news/docs/IBMSURAgrid.doc)
[3] SURAgrid User Management and PKI Bridge Certification Authority (https://www.pki.virginia.edu/nmi-bridge)
[4] TIGRE (http://tigreportal.hipcat.net)
[5] Virtual Data Toolkit (http://vdt.cs.wisc.edu/)
[6] SURAgrid Server Stack website (http://omnius.hpcc.ttu.edu/SURAgrid_wiki/ServerStack)
[7] Globus Toolkit 4.0 (http://www.globus.org/toolkit/docs/4.0/)
[8] GSI OpenSSH (http://grid.ncsa.uiuc.edu/ssh/)
[9] UberFTP (http://dims.ncsa.uiuc.edu/set/uberftp/)
[10] MyProxy (http://grid.ncsa.uiuc.edu/myproxy/)
[11] Condor-G (http://www.cs.wisc.edu/condor/condorg/)
[12] VDT supported operating systems (http://vdt.cs.wisc.edu/releases/1.6.1/requirements.html)
[13] pacman (http://www.archlinux.org/pacman/)
[14] SURAgrid PKI Bridge Certification Authority and User Management System (https://www.pki.virginia.edu/sura-bridge/scl/)
[15] grid-mapfile section (https://www.pki.virginia.edu/nmi-bridge/scl/#gridmapfile)
[16] International Grid Trust Federation (http://gridpma.org)
[17] GSI version of SSH (http://grid.ncsa.uiuc.edu/ssh/)
[18] Step 6: Install the GSI-OpenSSH Server (http://grid.ncsa.uiuc.edu/ssh/install.html#install_server)
[19] SURAgrid Support e-mail list (mailto:suragrid-support@sura.org)
[20] VDT Support page (http://vdt.cs.wisc.edu/support.html)
[21] OSG website (http://www.opensciencegrid.org)
[22] Members of the OSG Consortium (http://www.opensciencegrid.org/About/Who_is_the_Open_Science_Grid%3F/Consortium_Members)
[23] OSG partners (http://www.opensciencegrid.org/About/Who_is_the_Open_Science_Grid%3F/Partners)
[24] OSG Researcher fields (http://www.opensciencegrid.org/Science_on_the_OSG/Research_Highlights)
[25] Grid3 (http://www.ivdgl.org/grid2003/)
[26] Virtual Data Toolkit (VDT) (http://vdt.cs.wisc.edu//index.html)
[27] NSF Middleware Institute (NMI) (http://www.nsf-middleware.org/default.aspx)
[28] OSG@Work (http://twiki.grid.iu.edu/twiki/bin/view)
[29] OSG Education and Training (http://twiki.grid.iu.edu/twiki/bin/view/Education/WebHome)
[30] OSG Workshops (http://twiki.grid.iu.edu/twiki/bin/view/Education/GridWorkshops)
[31] OSG Research Highlights (http://www.opensciencegrid.org/Science_on_the_OSG/Research_Highlights)
[32] SRM collaboration working group (http://sdm.lbl.gov/srm-wg)
[33https://twiki.grid.iu.edu/twiki/bin/view/Storage] Storage Group (OSG)
[34] BeStMan (http://datagrid.lbl.gov/bestman)
[35] dCache (http://www.dcache.org)

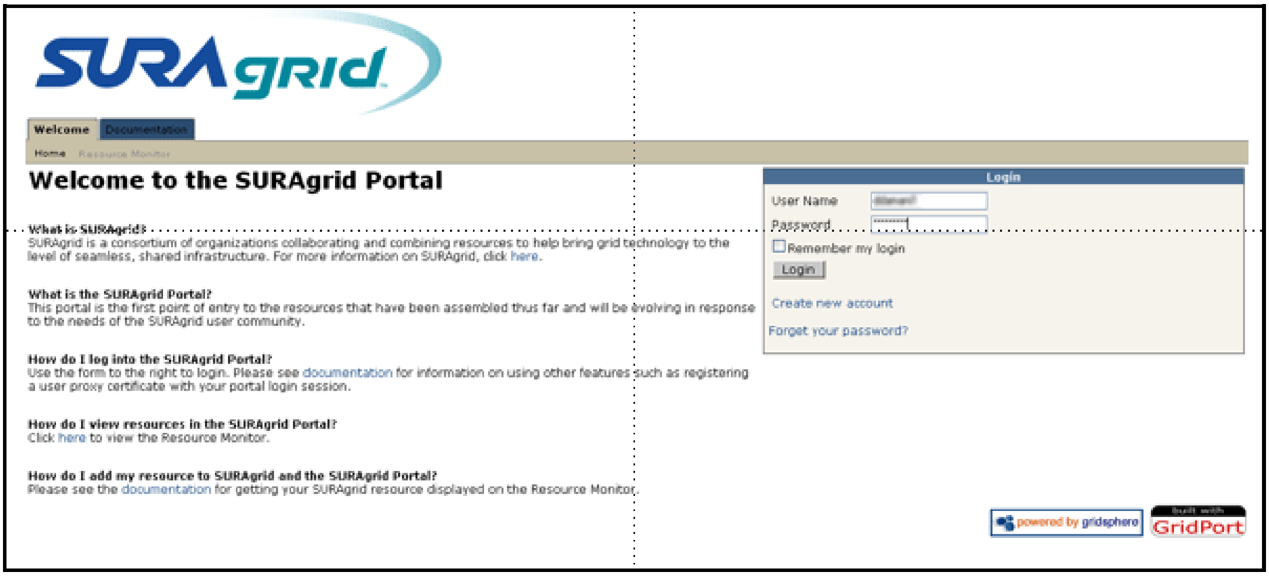
# Typical Usage Examples

The experience of using a grid can vary quite a bit from grid to grid given the possible variations in areas such as the user interface (e.g., command line, web portal, through an application), grid middleware (e.g., by grid product, Web services vs. pre-Web services), available applications and connected resources. As grid usage increases and diversifies, commonalities in the user experience are likely to emerge, similar to the way that basic personal computer skills currently transfer from one platform to another. The aspects most likely to homogenize over time can be previewed through large scale, multi-purpose grids today, which strive to develop portals that provide customized "MyPortal" views but efficient reuse of underlying functional components. The following examples from SURAgrid provide merely a glimpse of this range, with each having a different purpose and being intended for a specific user community. In future versions of the Cookbook, we look to expand this section with more examples of the variety of applications possible and the diversity of application environments.

## Job Submission on SURAgrid: Multiple Genome Alignment

In this example we show the steps that a researcher uses to submit a job through the SURAgrid portal for the Multiple Genome Alignment application at Georgia State University (GSU). This application takes multiple genome sequences as input and gives an aligned sequence based on structure. This is done using a memory efficient pair-wise alignment algorithm and parallelized code that can run on a grid.

As you will notice in this example, several levels of authentication are required to reach the grid, file storage, and compute resources. Our first step is to authenticate with (log into) SURAgrid. This is done by using a typical login window.



Welcome to the SURAgrid Portal

**What is SURAgrid?**
SURAgrid is a consortium of organizations collaborating and combining resources to help bring grid technology to the level of seamless, shared infrastructure. For more information on SURAgrid, click here.

**What is the SURAgrid Portal?**
This portal is the first point of entry to the resources that have been assembled thus far and will be evolving in response to the needs of the SURAgrid user community.

**How do I log into the SURAgrid Portal?**
Use the form to the right to login. Please see documentation for information on using other features such as registering a user proxy certificate with your portal login session.

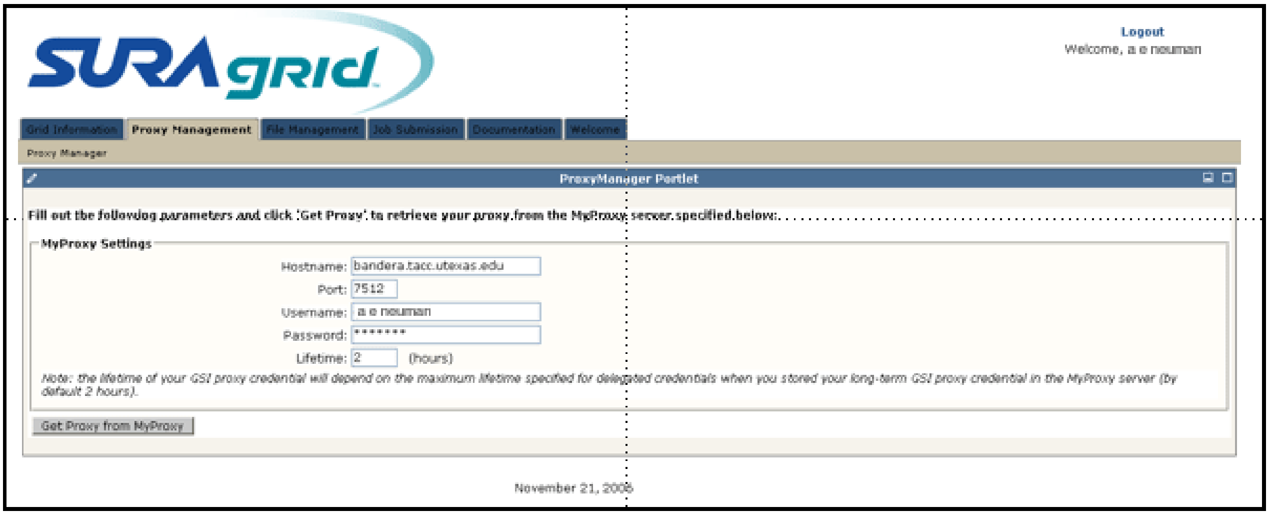
**How do I view resources in the SURAgrid Portal?**
Click here to view the Resource Monitor.

**How do I add my resource to SURAgrid and the SURAgrid Portal?**
Please see the documentation for getting your SURAgrid resource displayed on the Resource Monitor.

Login
User Name [redacted]
Password *********
□ Remember my login

Figure EX-1. Login window.

Our next step is to get a proxy from the proxy server MyProxy for the machine within the grid that holds our source data. Here we specify the grid resource (banderas.tacc), the port to use (7512), who we are (a e neuman [1]), and how long we want this access (2 hours). Upon clicking the "Get Proxy from MyProxy" button



Fill out the following parameters and click 'Get Proxy' to retrieve your proxy from the MyProxy server specified below:

MyProxy Settings

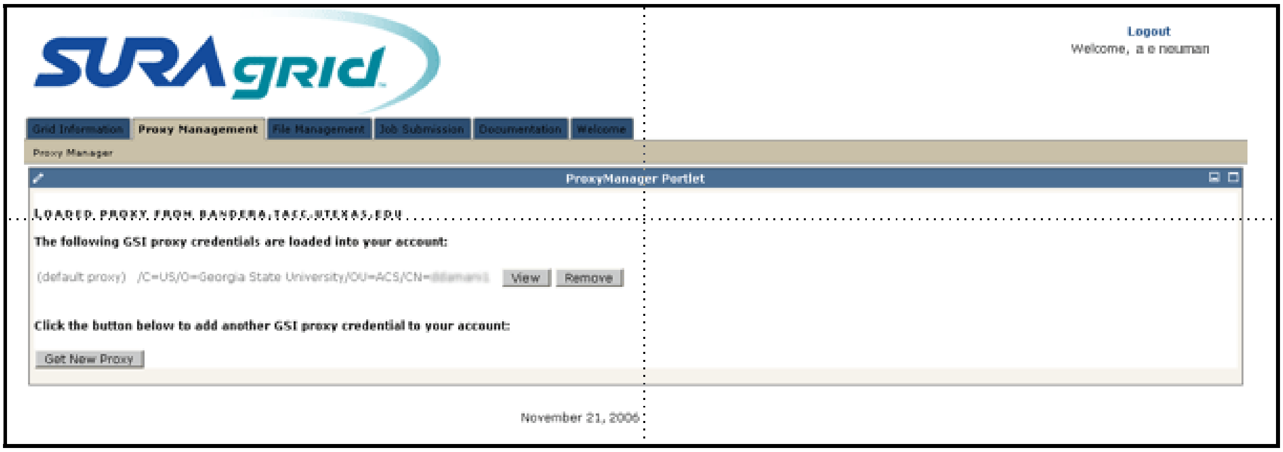
Hostname: bandera.tacc.utexas.edu
Port: 7512
Username: a e neuman
Password: *******
Lifetime: 2 (hours)

*Note: the lifetime of your GSI proxy credential will depend on the maximum lifetime specified for delegated credentials when you stored your long-term GSI proxy credential in the MyProxy server (by default 2 hours).*

November 21, 2006

Figure EX-2. Proxy request window.

the portal returns with information about our proxy.



LOADED PROXY FROM BANDERA.TACC.UTEXAS.EDU

The following GSI proxy credentials are loaded into your account:

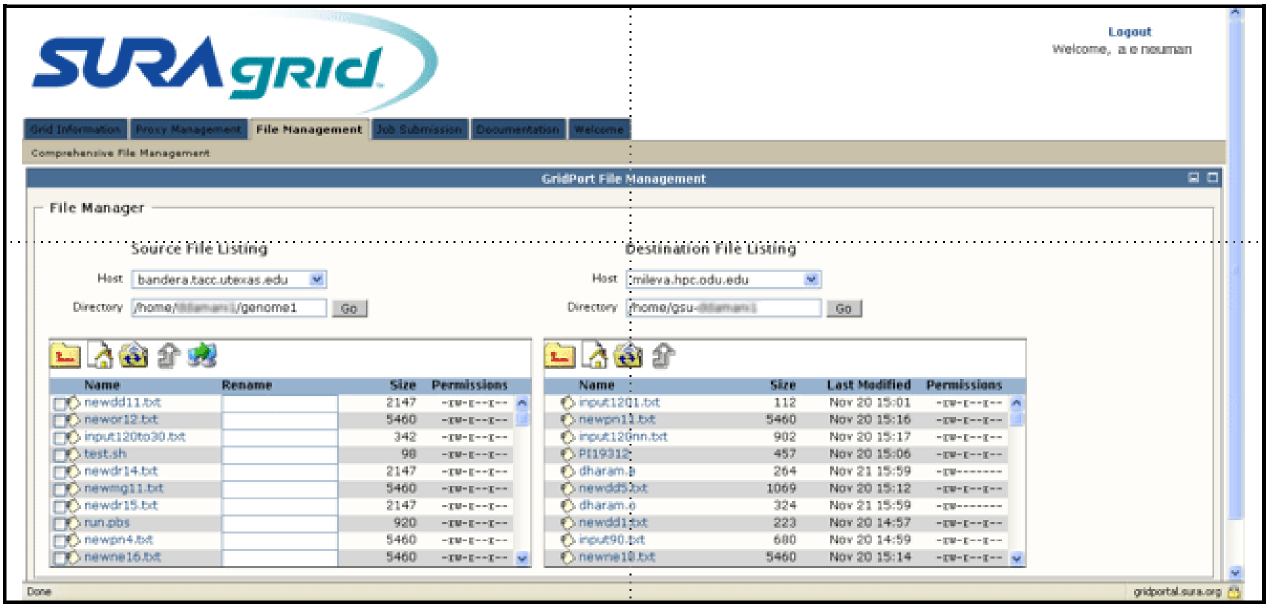
(default proxy) /C=US/O=Georgia State University/OU=ACS/CN=[redacted]

Click the button below to add another GSI proxy credential to your account:

November 21, 2006

Figure EX-3. Proxy response window.

Note the grid options in the tabs across the top. We can progress toward job submission via each of these tabs. Next we move our source data files over to the grid resource on which the computation is to be performed. Under the File Management tab, we find a secure copy tool. First we specify source machine (banderas), destination machine (mileva), and the relevant directories. SURAgrid responds with the directory contents menus.



Logout
Welcome, a e neuman

Comprehensive File Management

GridPort File Management

File Manager

Source File Listing

Host: bandera.tacc.utexas.edu
Directory: /home/[redacted]/genome1 Go

| Name | Rename | Size | Permissions |
|---|---|---|---|
| newdd11.txt | | 2147 | -rw-r--r-- |
| newor12.txt | | 5460 | -rw-r--r-- |
| input120to30.txt | | 342 | -rw-r--r-- |
| test.sh | | 98 | -rw-r--r-- |
| newdr14.txt | | 2147 | -rw-r--r-- |
| newmg11.txt | | 5460 | -rw-r--r-- |
| newdr15.txt | | 2147 | -rw-r--r-- |
| run.pbs | | 920 | -rw-r--r-- |
| newpn4.txt | | 5460 | -rw-r--r-- |
| newne16.txt | | 5460 | -rw-r--r-- |

Destination File Listing

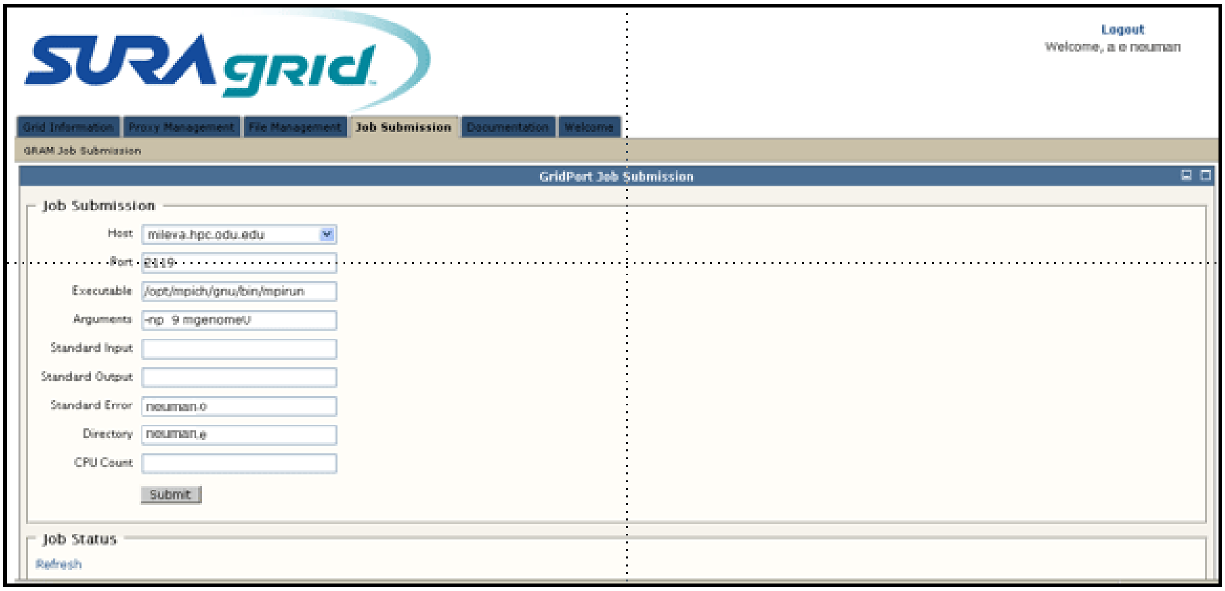
Host: mileva.hpc.odu.edu
Directory: /home/gsu-[redacted] Go

| Name | Size | Last Modified | Permissions |
|---|---|---|---|
| input1201.txt | 112 | Nov 20 15:01 | -rw-r--r-- |
| newpn11.txt | 5460 | Nov 20 15:16 | -rw-r--r-- |
| input120nn.txt | 902 | Nov 20 15:17 | -rw-r--r-- |
| PI19312 | 457 | Nov 20 15:06 | -rw-r--r-- |
| dharam.e | 264 | Nov 21 15:59 | -rw------- |
| newdd5.txt | 1069 | Nov 20 15:12 | -rw-r--r-- |
| dharam.o | 324 | Nov 21 15:59 | -rw------- |
| newdd1.txt | 223 | Nov 20 14:57 | -rw-r--r-- |
| input90.txt | 680 | Nov 20 14:59 | -rw-r--r-- |
| newne10.txt | 5460 | Nov 20 15:14 | -rw-r--r-- |

Done
gridportal.sura.org

Figure EX-4. Directory contents window.

At this point we select which files to move over to mileva. Now we're ready to submit the job via the Job Submission tab.



Logout
Welcome, a e neuman

GRAM Job Submission

GridPort Job Submission

Job Submission

| | |
|---|---|
| Host | mileva.hpc.odu.edu |
| Port | 2119 |
| Executable | /opt/mpich/gnu/bin/mpirun |
| Arguments | -np 9 mgenomeU |
| Standard Input | |
| Standard Output | |
| Standard Error | neuman.o |
| Directory | neuman.e |
| CPU Count | |

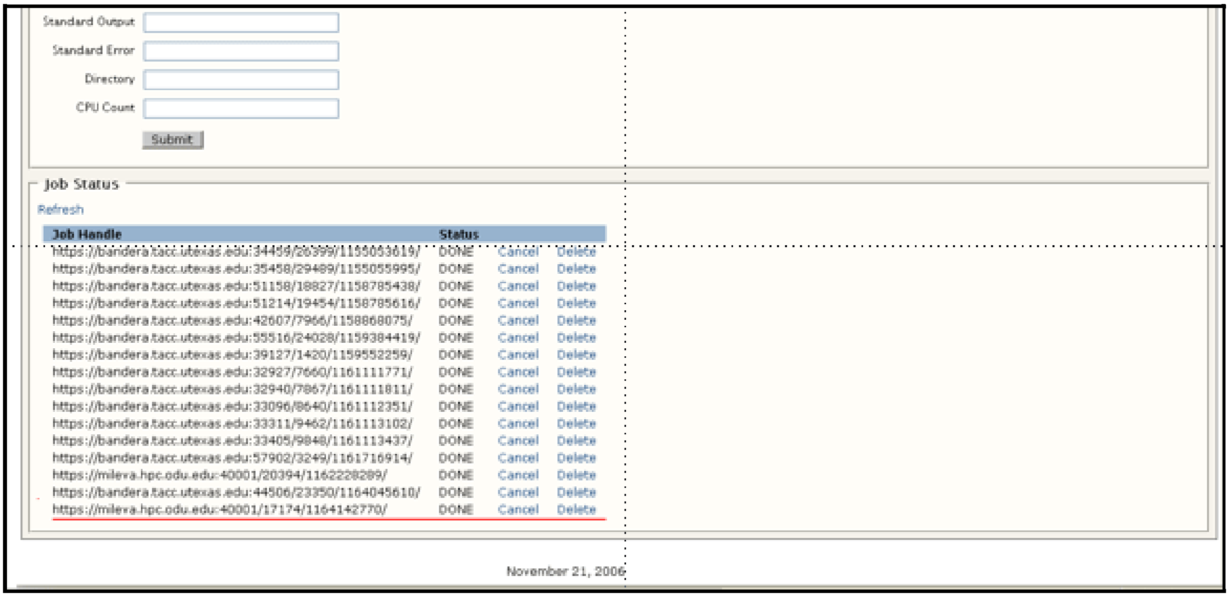
Submit

Job Status

Refresh

Figure EX-5. Job submission window.

When we submit the job, a "job handle" (underlined in the figure below) is returned to us so we might track the job via the Job Status window under the Job Submission tab.



| | |
|---|---|
| Standard Output | |
| Standard Error | |
| Directory | |
| CPU Count | |

Job Status

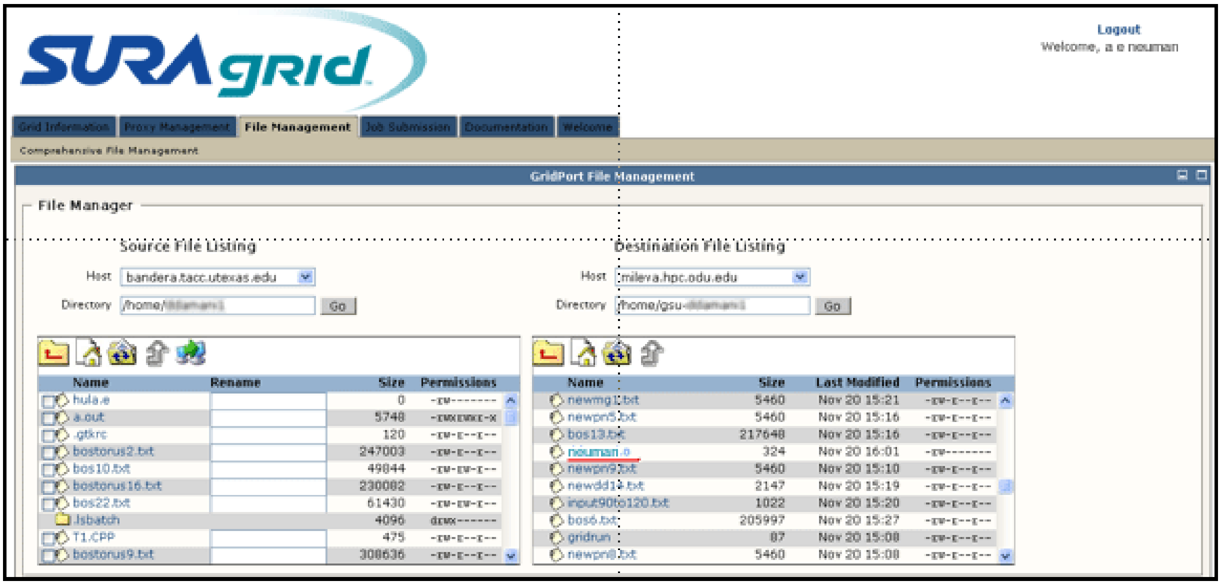
| Job Handle | Status | | |
|---|---|---|---|
| https://bandera.tacc.utexas.edu:34459/26399/1155053619/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:35458/29489/1155055995/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:51158/18827/1158785438/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:51214/19454/1158785616/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:42607/7966/1158868075/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:55516/24028/1159384419/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:39127/1420/1159552259/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:32927/7660/1161111771/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:32940/7867/1161111811/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:33096/8640/1161112351/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:33311/9462/1161113102/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:33405/9848/1161113437/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:57902/3249/1161716914/ | DONE | Cancel | Delete |
| https://mileva.hpc.odu.edu:40001/20394/1162228289/ | DONE | Cancel | Delete |
| https://bandera.tacc.utexas.edu:44506/23350/1164045610/ | DONE | Cancel | Delete |
| https://mileva.hpc.odu.edu:40001/17174/1164142770/ | DONE | Cancel | Delete |

November 21, 2006

Figure EX-6. Job status window.

From this vantage point we can track the job via the Status column, compare it with progress of other jobs on the grid, Cancel the job, or Delete it completely.

When the job completes (is DONE), we can retrieve our output files



Welcome, a e neuman

Comprehensive File Management

GridPort File Management

File Manager

Source File Listing

Host: bandera.tacc.utexas.edu

Directory: /home/[redacted]

| Name | Rename | Size | Permissions |
|---|---|---|---|
| □ hula.e | | 0 | -rw------- |
| □ a.out | | 5748 | -rwxrwxr-x |
| □ .gtkrc | | 120 | -rw-r--r-- |
| □ bostorus2.txt | | 247003 | -rw-r--r-- |
| □ bos10.txt | | 49844 | -rw-rw-r-- |
| □ bostorus16.txt | | 230082 | -rw-r--r-- |
| □ bos22.txt | | 61430 | -rw-rw-r-- |
| .lsbatch | | 4096 | drwx------ |
| □ T1.CPP | | 475 | -rw-r--r-- |
| □ bostorus9.txt | | 308636 | -rw-r--r-- |

Destination File Listing

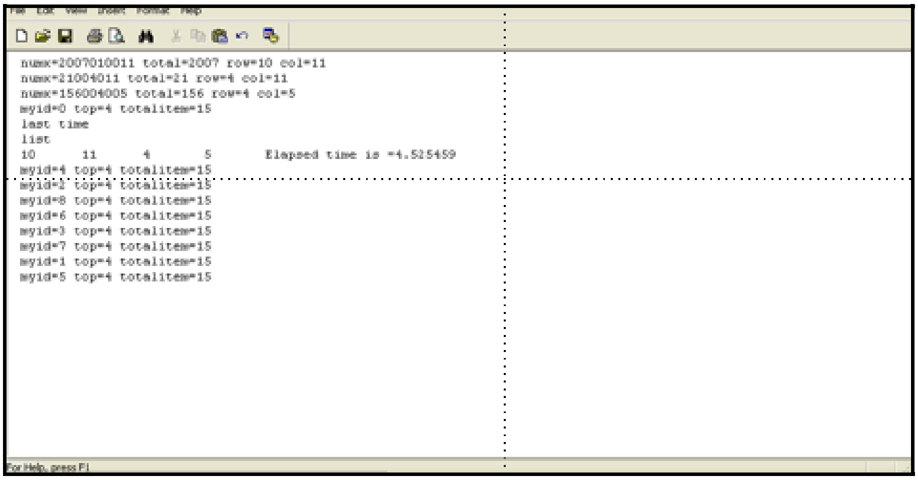
Host: mileva.hpc.odu.edu

Directory: /home/gsu-[redacted]

| Name | Size | Last Modified | Permissions |
|---|---|---|---|
| newmg1.txt | 5460 | Nov 20 15:21 | -rw-r--r-- |
| newpn5.txt | 5460 | Nov 20 15:16 | -rw-r--r-- |
| bos13.txt | 217648 | Nov 20 15:16 | -rw-r--r-- |
| neuman.o | 324 | Nov 20 16:01 | -rw------- |
| newpn9.txt | 5460 | Nov 20 15:10 | -rw-r--r-- |
| newdd14.txt | 2147 | Nov 20 15:19 | -rw-r--r-- |
| input90to120.txt | 1022 | Nov 20 15:20 | -rw-r--r-- |
| bos6.txt | 205997 | Nov 20 15:27 | -rw-r--r-- |
| gridrun | 87 | Nov 20 15:08 | -rw-r--r-- |
| newpn8.txt | 5460 | Nov 20 15:08 | -rw-r--r-- |

Figure EX-7. Directory contents window.

similarly via the File Management tab. We can now look at our output file.



```
numk=2007010011 total=2007 row=10 col=11
numk=21004011 total=21 row=4 col=11
numk=156004005 total=156 row=4 col=5
myid=0 top=4 totalitem=15
last time
list
10      11      4       5       Elapsed time is =4.525459
myid=4 top=4 totalitem=15
myid=2 top=4 totalitem=15
myid=8 top=4 totalitem=15
myid=6 top=4 totalitem=15
myid=3 top=4 totalitem=15
myid=7 top=4 totalitem=15
myid=1 top=4 totalitem=15
myid=5 top=4 totalitem=15
```

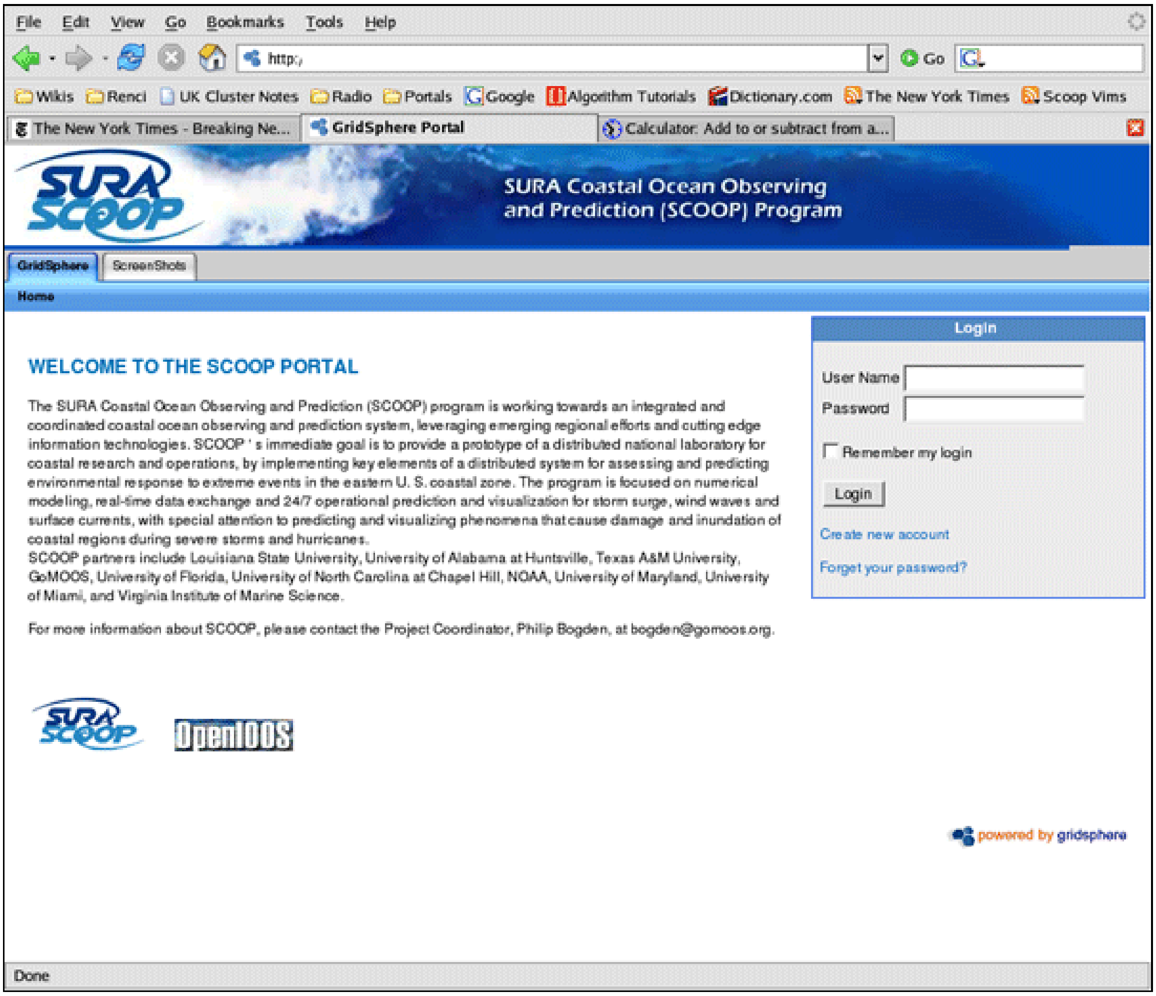
Figure EX-8. Job output window.

For more information on using or joining SURAgrid, see About SURAgrid [2].

## SCOOP (SURA Coastal Ocean Observing & Prediction) Demonstration Portal

The following example is a prototype of a distributed national laboratory for coastal research and operations, as being developed by the SCOOP program (scoop.sura.org.) SCOOP overall is focused on numerical modeling, real-time data exchange and 24/7 operational prediction and visualization for storm surge, wind, waves, and surface currents, with special attention to predicting and visualizing phenomena that cause damage and inundation of coastal regions during severe storm and hurricanes.

In this section we show some shots of the SCOOP Gridsphere demonstration portal as of Fall 2006. The portal is web accessible and we start with a login page.



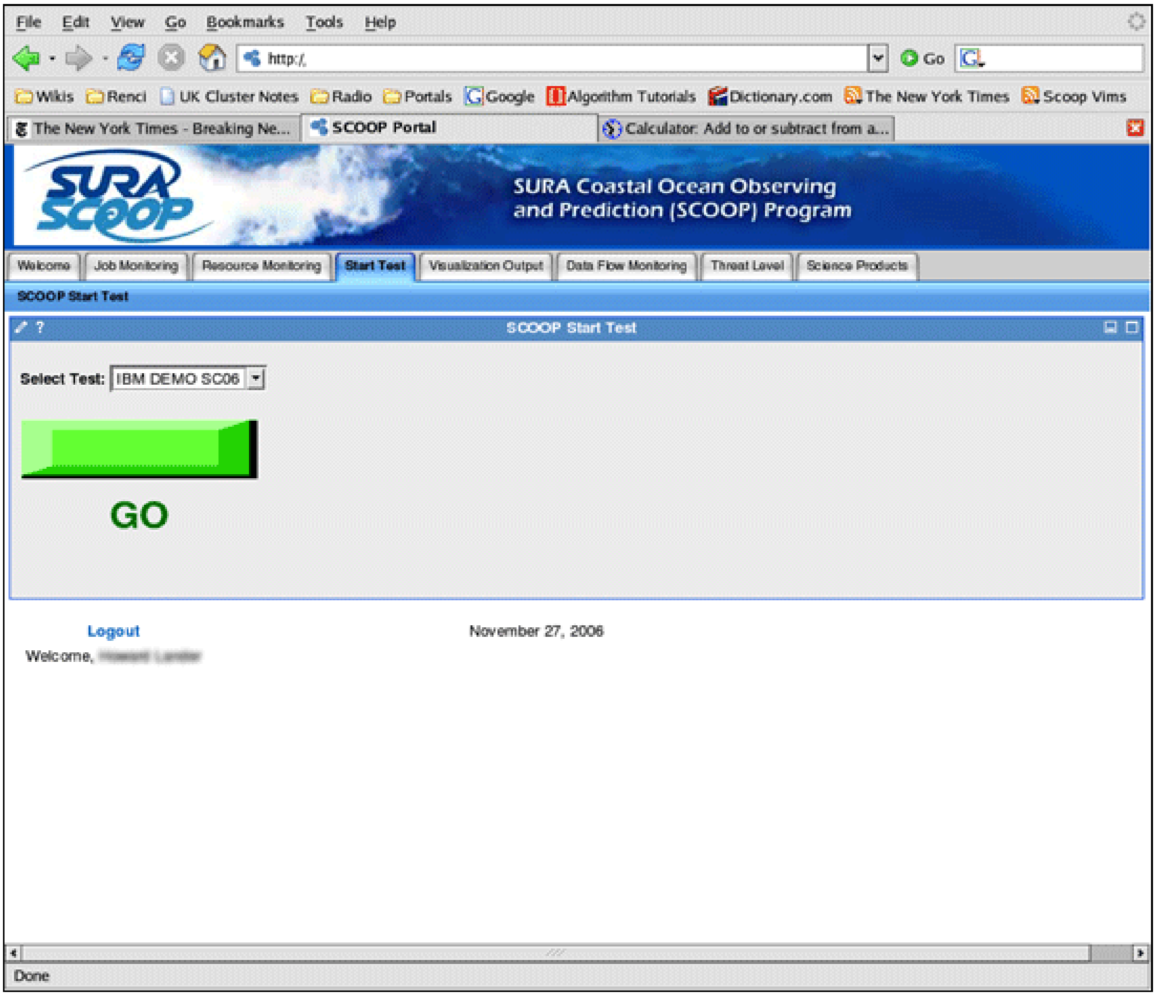
WELCOME TO THE SCOOP PORTAL

The SURA Coastal Ocean Observing and Prediction (SCOOP) program is working towards an integrated and coordinated coastal ocean observing and prediction system, leveraging emerging regional efforts and cutting edge information technologies. SCOOP ' s immediate goal is to provide a prototype of a distributed national laboratory for coastal research and operations, by implementing key elements of a distributed system for assessing and predicting environmental response to extreme events in the eastern U. S. coastal zone. The program is focused on numerical modeling, real-time data exchange and 24/7 operational prediction and visualization for storm surge, wind waves and surface currents, with special attention to predicting and visualizing phenomena that cause damage and inundation of coastal regions during severe storms and hurricanes.
SCOOP partners include Louisiana State University, University of Alabama at Huntsville, Texas A&M University, GoMOOS, University of Florida, University of North Carolina at Chapel Hill, NOAA, University of Maryland, University of Miami, and Virginia Institute of Marine Science.

For more information about SCOOP, please contact the Project Coordinator, Philip Bogden, at bogden@gomoos.org.

Figure EX-10. Login window.

Once we log in, we're presented with a number of tools across the menu bar. Under the Start Test tab we are presented with various models and tests that we can run on the grid. Here we get ready to rerun the demo that was constructed for the SC06 conference.



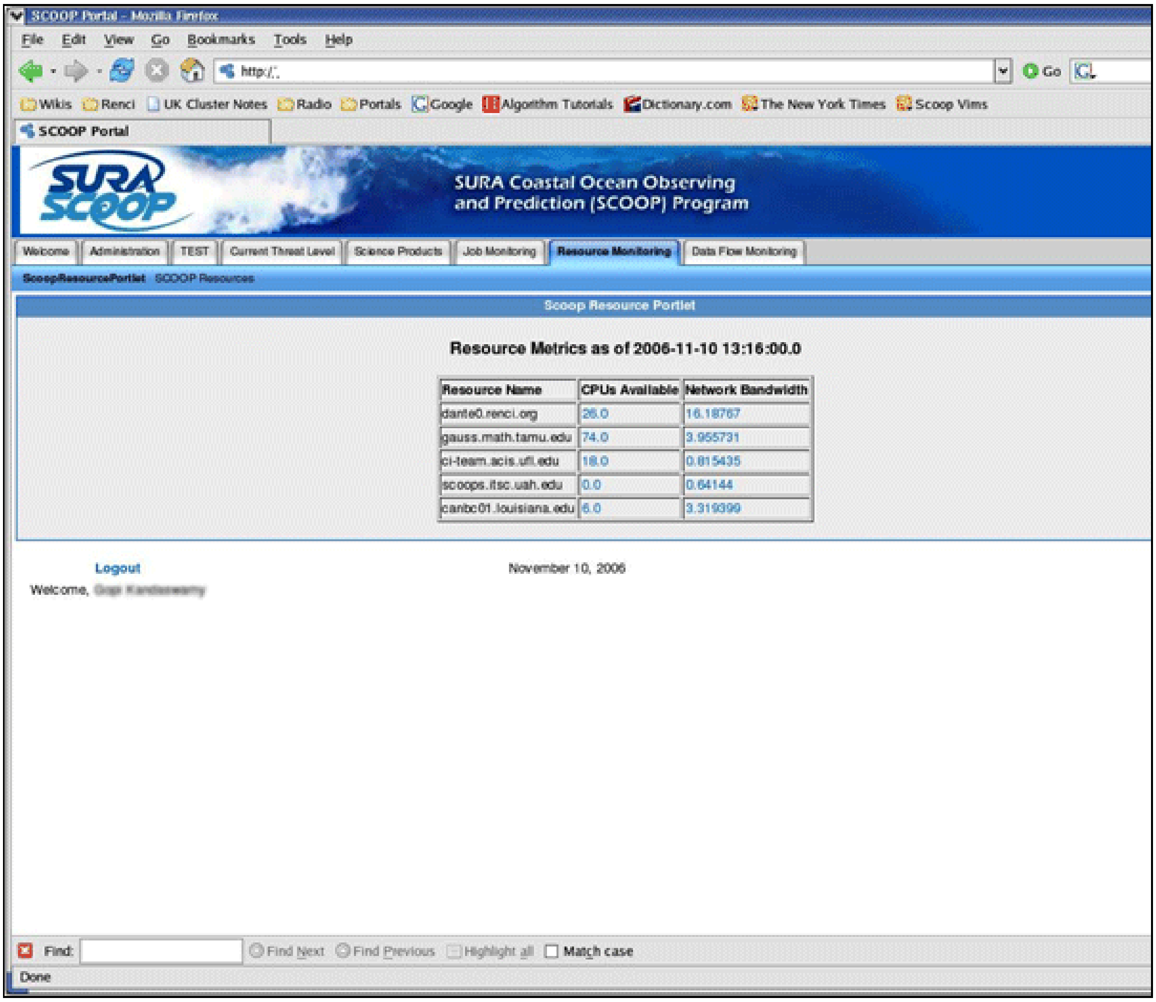
Select Test: IBM DEMO SC06

November 27, 2006

Welcome, [redacted]

Figure EX-11. Running a demo.

From here we move to the Resource Monitoring page and check to see which resources are available for our use. We see that 5 machines are in the grid and up to 124 CPUs are available. Network bandwidth information is also available.



Resource Metrics as of 2006-11-10 13:16:00.0

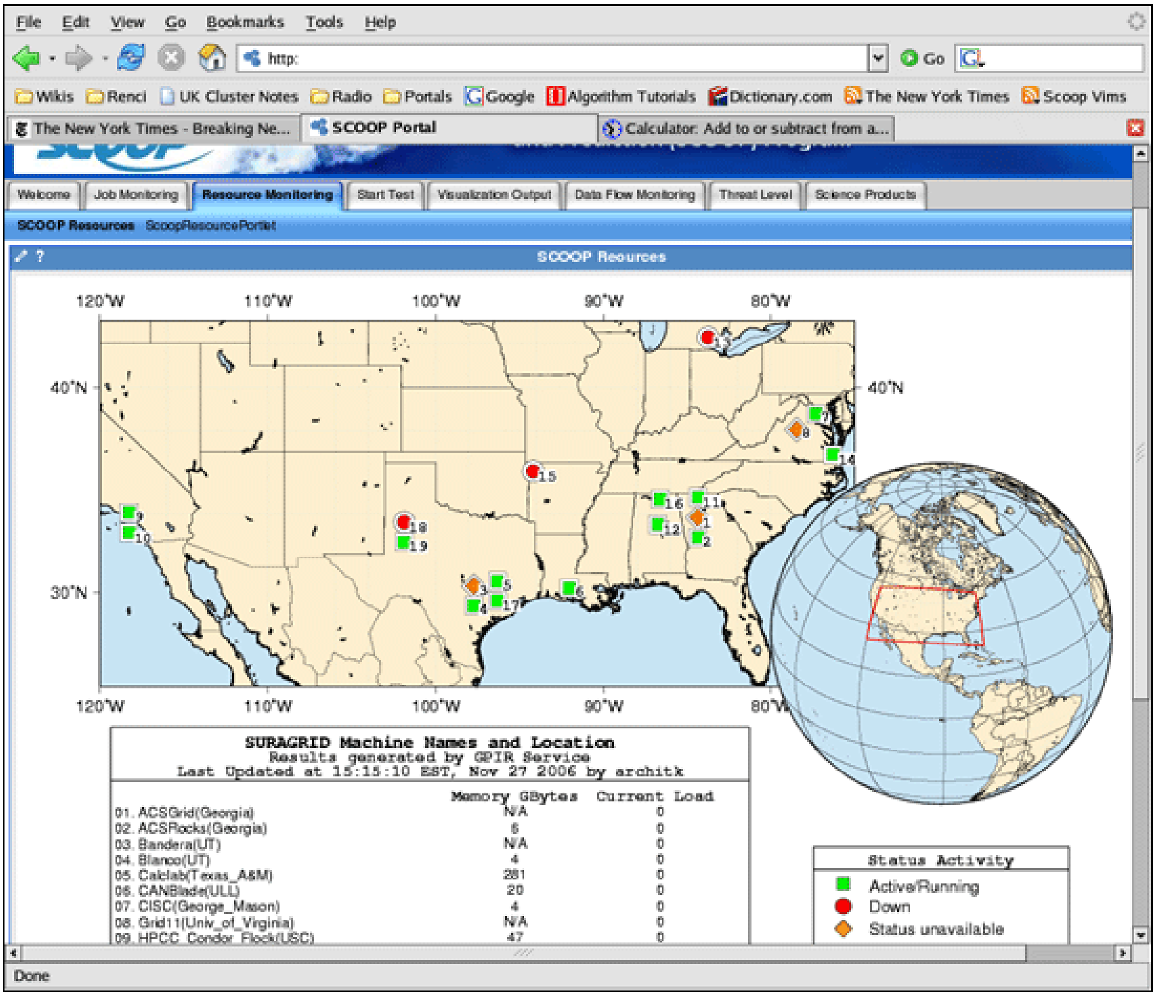
| Resource Name | CPUs Available | Network Bandwidth |
|---|---|---|
| dante0.renci.org | 26.0 | 16.18767 |
| gauss.math.tamu.edu | 74.0 | 3.955731 |
| ci-team.acis.ufl.edu | 18.0 | 0.815435 |
| scoops.itsc.uah.edu | 0.0 | 0.64144 |
| canbc01.louisiana.edu | 6.0 | 3.319399 |

November 10, 2006

Welcome, [redacted]

Figure EX-12. Resource information.

We can also get a graphical image of this information using the Resource Portlet. Note the color/shape coded status information available in this view.



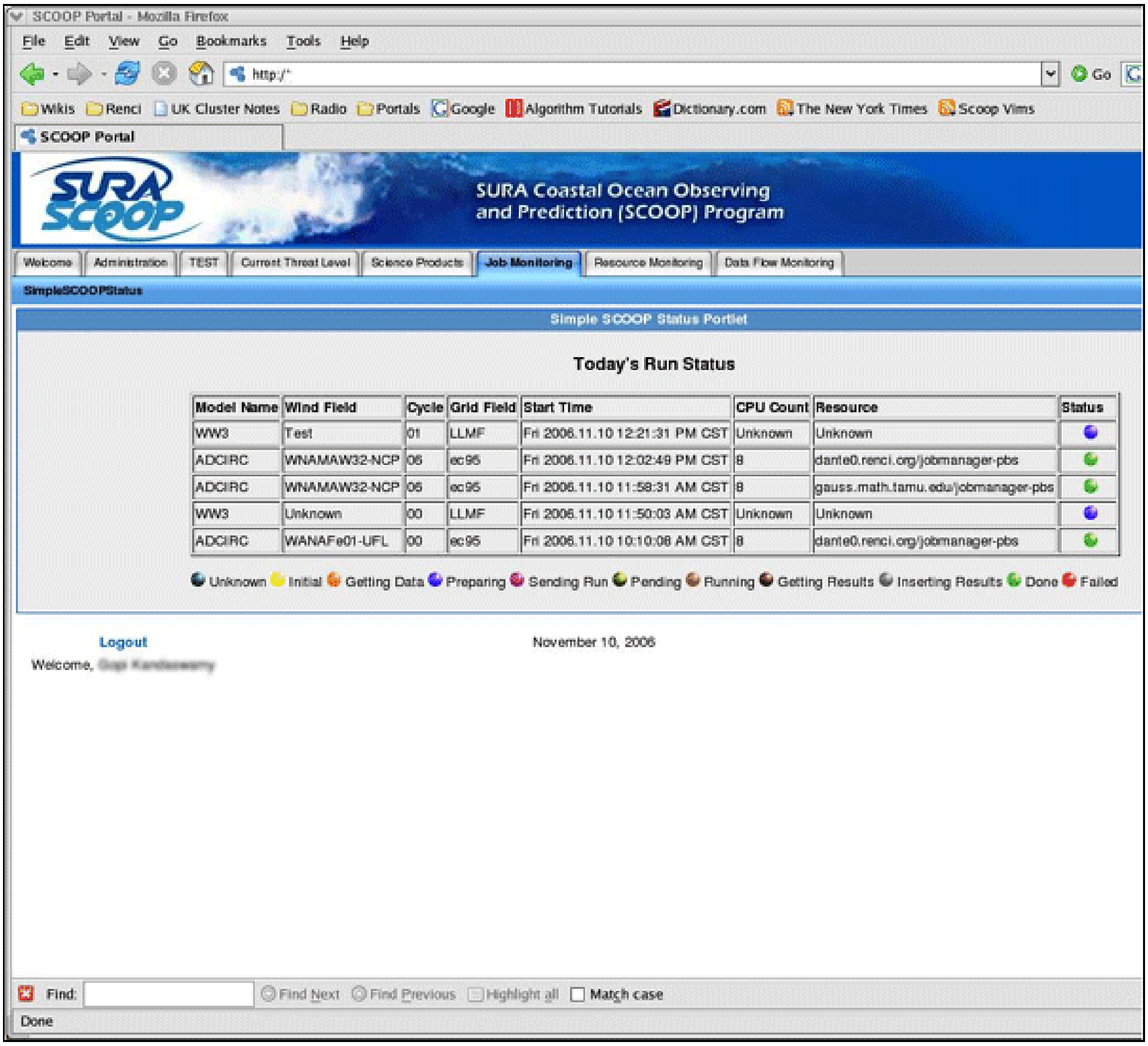
SURAGRID Machine Names and Location
Results generated by GPIR Service
Last Updated at 15:15:10 EST, Nov 27 2006 by architk

| | Memory GBytes | Current Load |
|---|---|---|
| 01. ACSGrid(Georgia) | N/A | 0 |
| 02. ACSRocks(Georgia) | 6 | 0 |
| 03. Bandera(UT) | N/A | 0 |
| 04. Blanco(UT) | 4 | 0 |
| 05. Calclab(Texas_A&M) | 281 | 0 |
| 06. CANBlade(ULL) | 20 | 0 |
| 07. CISC(George_Mason) | 4 | 0 |
| 08. Grid11(Univ_of_Virginia) | N/A | 0 |
| 09. HPCC_Condor_Flock(USC) | 47 | 0 |

| Status Activity | |
|---|---|
| 🟩 | Active/Running |
| ● | Down |
| ◆ | Status unavailable |

Figure Ex 13. SCOOP resource portlet.

We can monitor our job's progress and compare that progress with other activities on the grid through the Job Monitoring tab. We can quickly see status via the color coded Status ball. Start time and CPU usage of other jobs can also assist us in determining when our job is likely to complete.


SCOOP Portal - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://
Go
Wikis Renci UK Cluster Notes Radio Portals Google Algorithm Tutorials Dictionary.com The New York Times Scoop Vims
SCOOP Portal
SURA SCOOP
SURA Coastal Ocean Observing and Prediction (SCOOP) Program
Welcome Administration TEST Current Threat Level Science Products Job Monitoring Resource Monitoring Data Flow Monitoring
SimpleSCOOPStatus
Simple SCOOP Status Portlet
Find: Find Next Find Previous Highlight all Match case
Done

Today's Run Status

| Model Name | Wind Field | Cycle | Grid Field | Start Time | CPU Count | Resource | Status |
|---|---|---|---|---|---|---|---|
| WW3 | Test | 01 | LLMF | Fri 2006.11.10 12:21:31 PM CST | Unknown | Unknown | ● |
| ADCIRC | WNAMAW32-NCP | 06 | ec95 | Fri 2006.11.10 12:02:49 PM CST | 8 | dante0.renci.org/jobmanager-pbs | ● |
| ADCIRC | WNAMAW32-NCP | 06 | ec95 | Fri 2006.11.10 11:58:31 AM CST | 8 | gauss.math.tamu.edu/jobmanager-pbs | ● |
| WW3 | Unknown | 00 | LLMF | Fri 2006.11.10 11:50:03 AM CST | Unknown | Unknown | ● |
| ADCIRC | WANAFe01-UFL | 00 | ec95 | Fri 2006.11.10 10:10:08 AM CST | 8 | dante0.renci.org/jobmanager-pbs | ● |

● Unknown ● Initial ● Getting Data ● Preparing ● Sending Run ● Pending ● Running ● Getting Results ● Inserting Results ● Done ● Failed

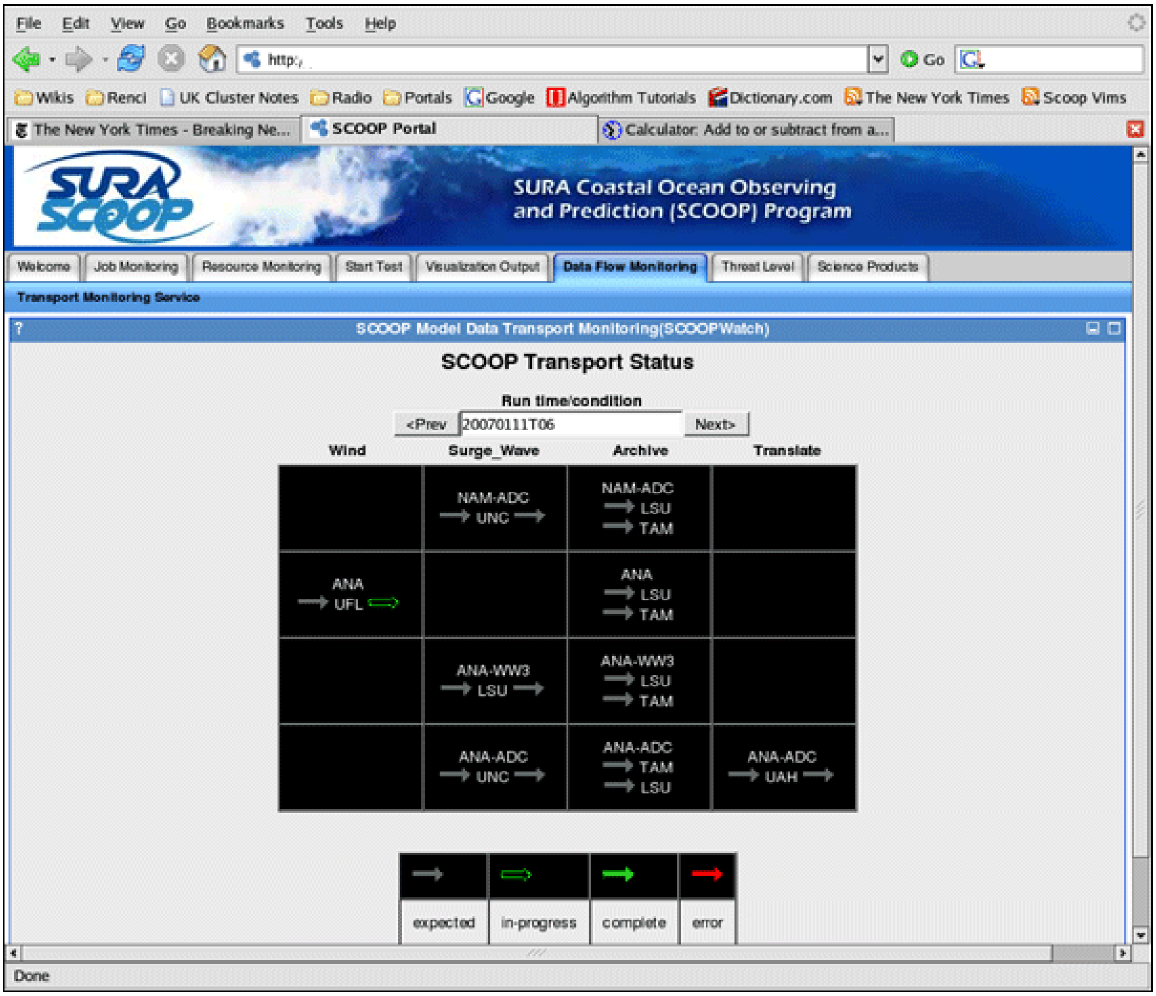
Logout

November 10, 2006

Welcome, [redacted]

Figure Ex 14. Job status window.

We can monitor the flow of various data types within the models and across the computer systems within the grid.



SCOOP Transport Status

Run time/condition

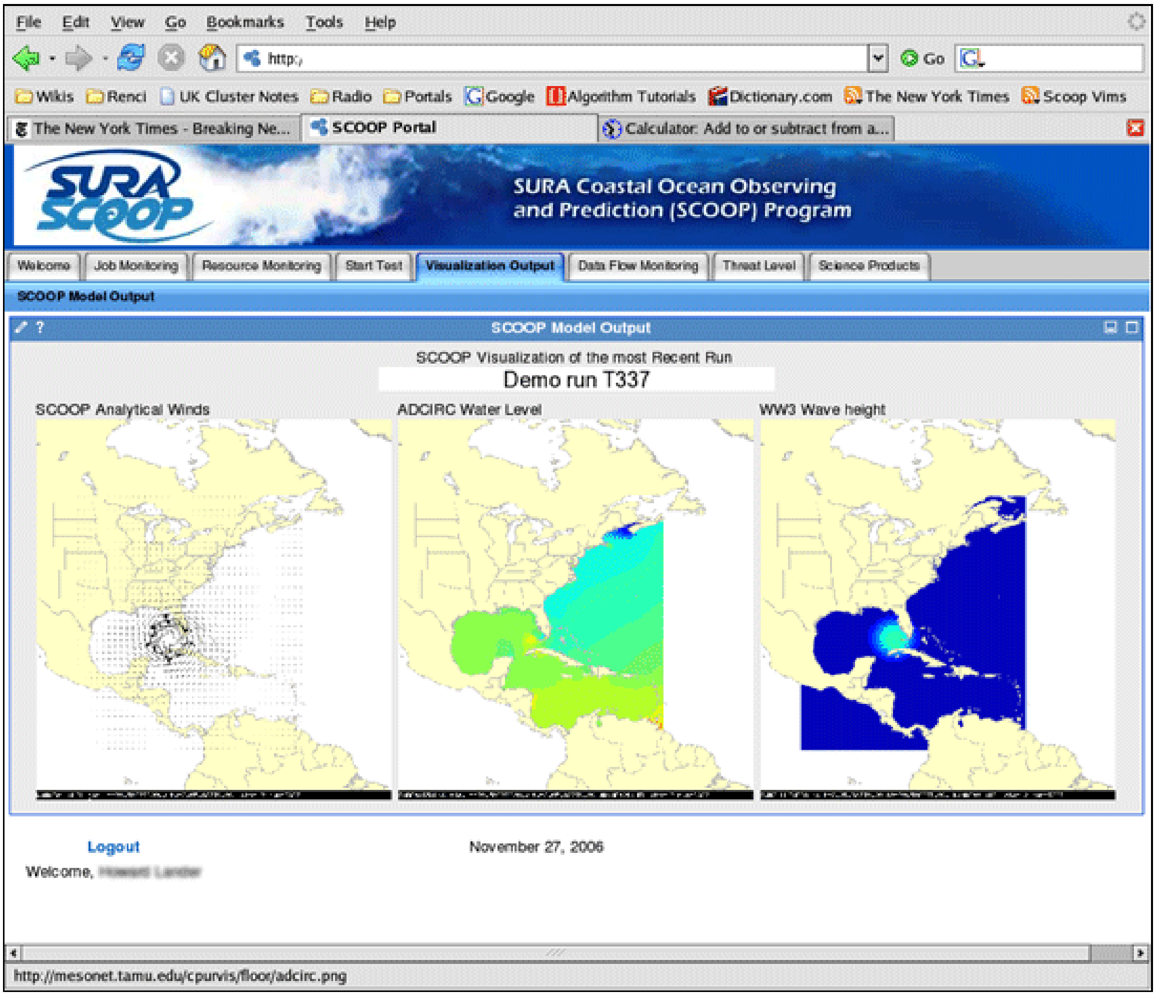
<Prev 20070111T06 Next>

| Wind | Surge_Wave | Archive | Translate |
|---|---|---|---|
| | NAM-ADC<br>→ UNC → | NAM-ADC<br>→ LSU<br>→ TAM | |
| ANA<br>→ UFL ⇨ | | ANA<br>→ LSU<br>→ TAM | |
| | ANA-WW3<br>→ LSU → | ANA-WW3<br>→ LSU<br>→ TAM | |
| | ANA-ADC<br>→ UNC → | ANA-ADC<br>→ TAM<br>→ LSU | ANA-ADC<br>→ UAH → |

| → | ⇨ | → | → |
|---|---|---|---|
| expected | in-progress | complete | error |

Figure EX-15. Data transport information.

Finally, we can use the portal's visualization tools to help us understand the results.



SCOOP Visualization of the most Recent Run

Demo run T337

November 27, 2006

Welcome, [redacted]

http://mesonet.tamu.edu/cpurvis/floor/adcirc.png

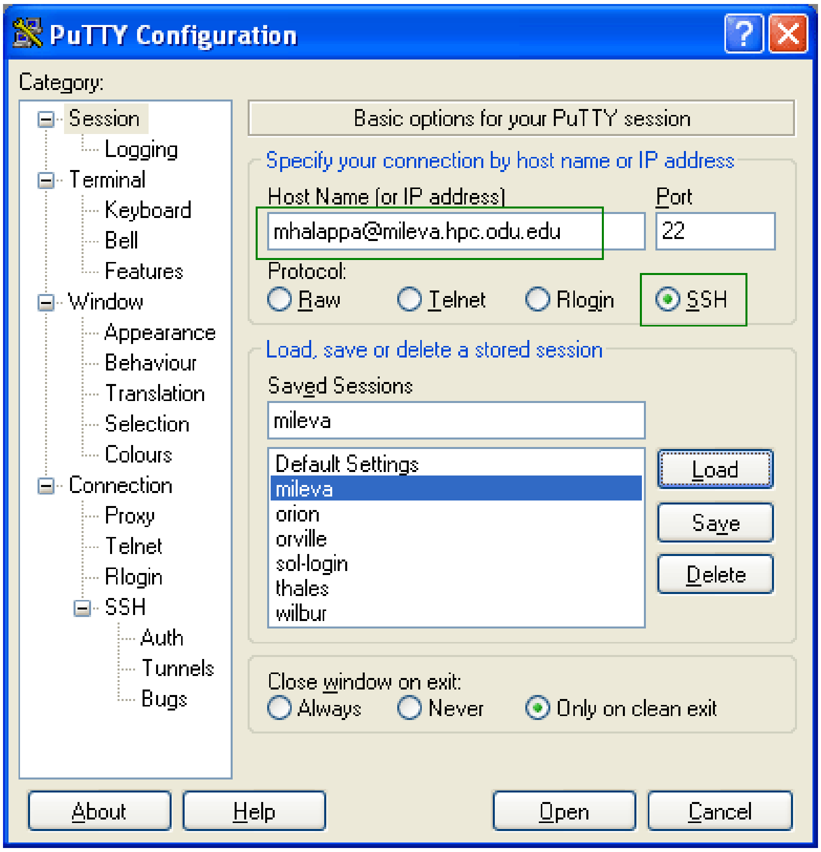
Figure EX-16. Visualization of results.

For more information, see the SURA Coastal Ocean Observing and Prediction (SCOOP) Program [9] web page.

# Job Submission: Bio-electric Simulator for Whole Body Tissue

The following application, run on SURAgrid, from Old Dominion University is designed to simulate the response of a "whole body tissue" model to potential/current stimulus through direct electrode contact and uses a command line interface for access. While web API's are rapidly developing (requiring more work but with the potential for much improved job management), some people still prefer command line interfaces. Those who are familiar with using command line options to submit jobs to a cluster will notice that the differences are not significant. Command line access is also often useful in development and initial debugging of grid applications. The ODU project team sees their current version of BioSim as their first stage of running on a grid, which has already enabled them to make significant headway on their initial goal of testing the scalability of BioSim on much larger compute clusters then would be available locally. In the next stage, the research team will modify BioSim to use the SURAgrid portal for data and job management, eventually adding automatic selection of the best available resources and dynamic job control.

The first step is to log in to a local resource and generate proxy credentials. This is done with a call to *grid-proxy-init*. Further information is provided by the *grid-proxy-info* command.



Host Name (or IP address): mhalappa@mileva.hpc.odu.edu

Port: 22

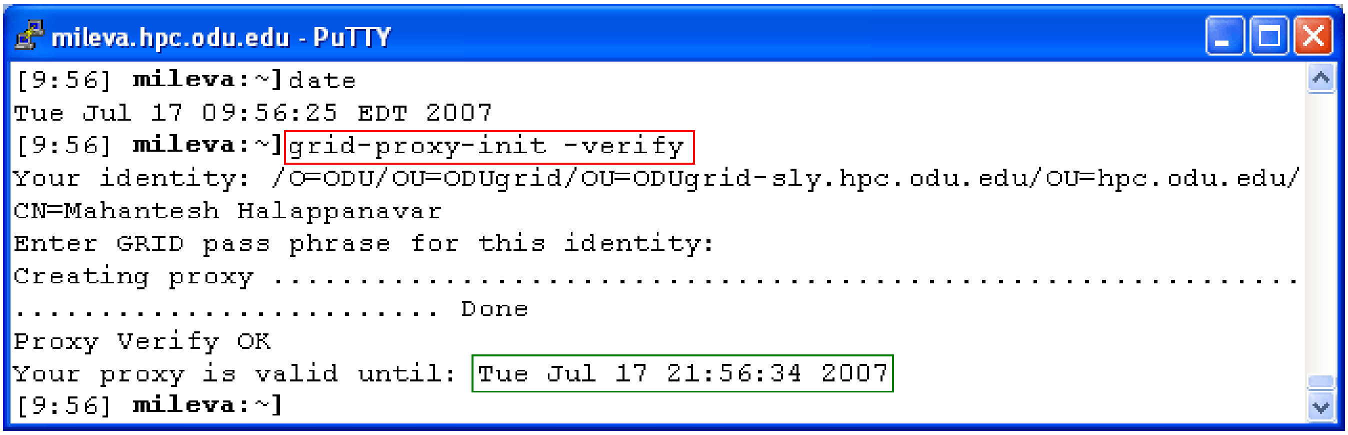
Protocol: ○ Raw ○ Telnet ○ Rlogin ● SSH

Saved Sessions: mileva

Default Settings<br>
mileva<br>
orion<br>
orville<br>
sol-login<br>
thales<br>
wilbur

Close window on exit: ○ Always ○ Never ● Only on clean exit

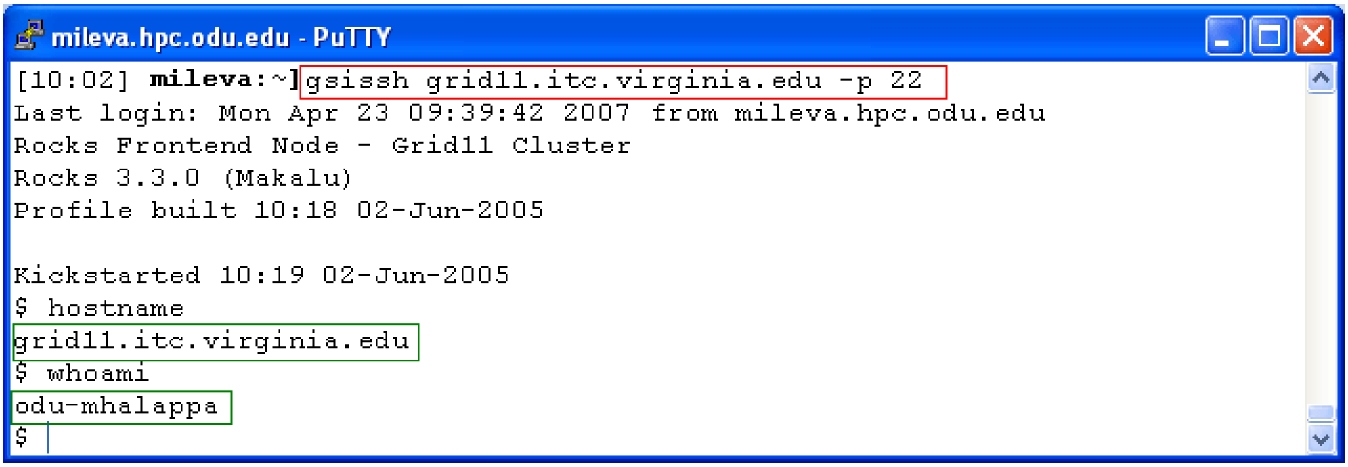
Figure EX-17. Local login window (using Putty terminal application).



```
[9:56] mileva:~]date
Tue Jul 17 09:56:25 EDT 2007
[9:56] mileva:~]grid-proxy-init -verify
Your identity: /O=ODU/OU=ODUgrid/OU=ODUgrid-sly.hpc.odu.edu/OU=hpc.odu.edu/
CN=Mahantesh Halappanavar
Enter GRID pass phrase for this identity:
Creating proxy ...............................................................
......................... Done
Proxy Verify OK
Your proxy is valid until: Tue Jul 17 21:56:34 2007
[9:56] mileva:~]
```

Figure EX-18. Proxy credential creation.

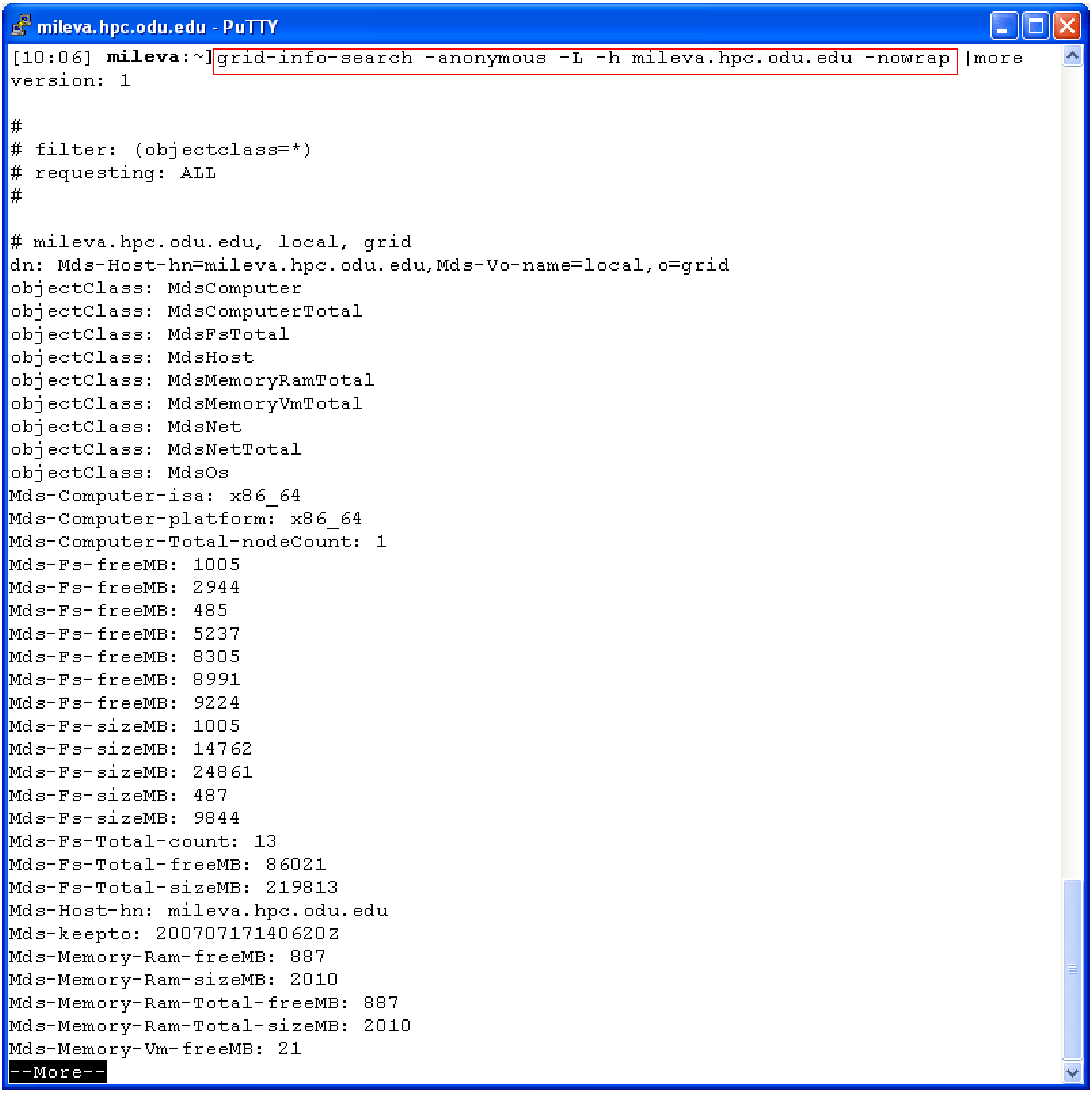
The system returns with a proxy that is valid for 12 hours. A call to *gsissh* completes the connection. (GSI-OpenSSH is a modified version of OpenSSH that adds support for GSI authentication and credential forwarding [delegation], providing a single sign-on remote login and file transfer service.)



```
[10:02] mileva:~]gsissh grid11.itc.virginia.edu -p 22
Last login: Mon Apr 23 09:39:42 2007 from mileva.hpc.odu.edu
Rocks Frontend Node - Grid11 Cluster
Rocks 3.3.0 (Makalu)
Profile built 10:18 02-Jun-2005

Kickstarted 10:19 02-Jun-2005
$ hostname
grid11.itc.virginia.edu
$ whoami
odu-mhalappa
$
```

Figure EX-19. GSI-OpenSSH login window.

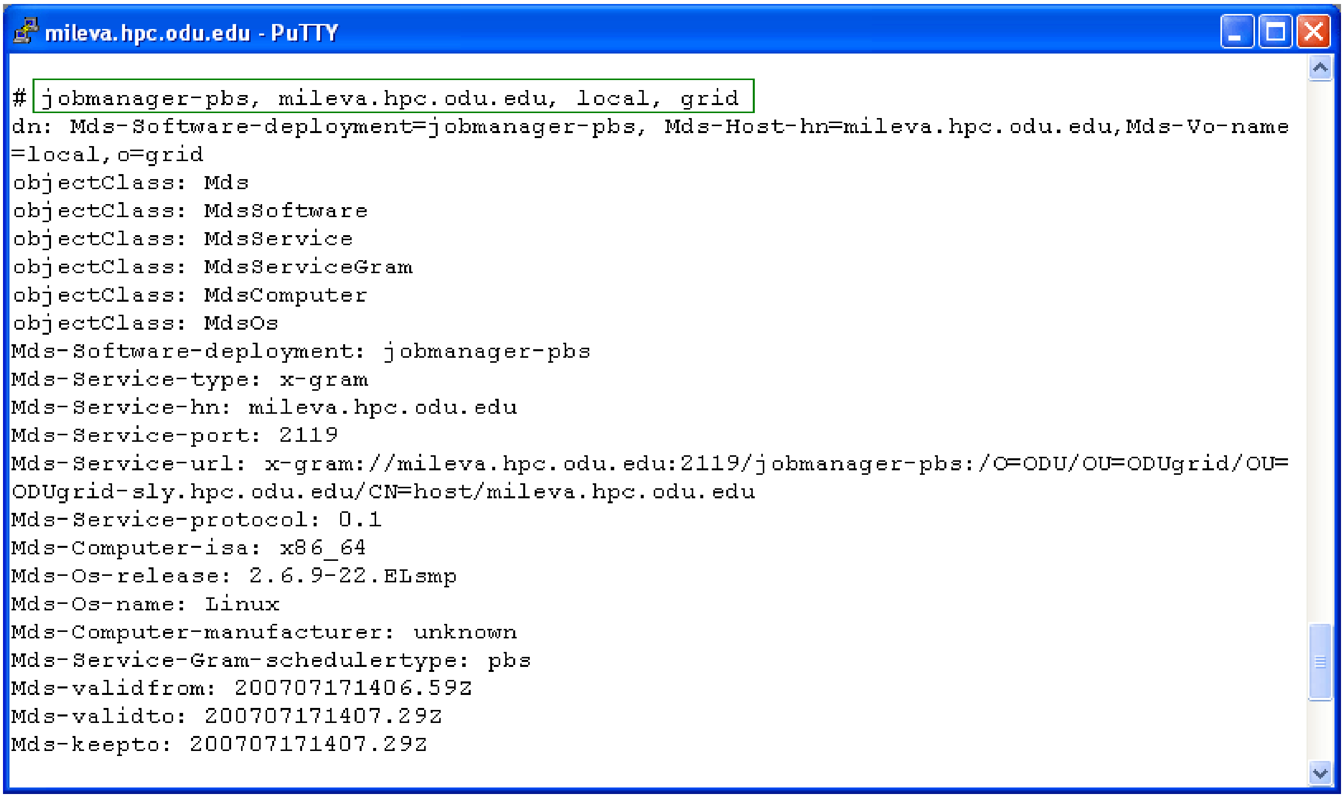
A Grid Information Service (GIS — not to be confused with GSI!) provides resource discovery and monitoring services for a grid. The shell command *grid-info-search* performs searches on a GIS server based on search filters that conform to LDAP searches, returning information for compute resources based on search criteria provided on the command line.



```
[10:06] mileva:~]grid-info-search -anonymous -L -h mileva.hpc.odu.edu -nowrap |more
version: 1

#
# filter: (objectclass=*)
# requesting: ALL
#

# mileva.hpc.odu.edu, local, grid
dn: Mds-Host-hn=mileva.hpc.odu.edu,Mds-Vo-name=local,o=grid
objectClass: MdsComputer
objectClass: MdsComputerTotal
objectClass: MdsFsTotal
objectClass: MdsHost
objectClass: MdsMemoryRamTotal
objectClass: MdsMemoryVmTotal
objectClass: MdsNet
objectClass: MdsNetTotal
objectClass: MdsOs
Mds-Computer-isa: x86_64
Mds-Computer-platform: x86_64
Mds-Computer-Total-nodeCount: 1
Mds-Fs-freeMB: 1005
Mds-Fs-freeMB: 2944
Mds-Fs-freeMB: 485
Mds-Fs-freeMB: 5237
Mds-Fs-freeMB: 8305
Mds-Fs-freeMB: 8991
Mds-Fs-freeMB: 9224
Mds-Fs-sizeMB: 1005
Mds-Fs-sizeMB: 14762
Mds-Fs-sizeMB: 24861
Mds-Fs-sizeMB: 487
Mds-Fs-sizeMB: 9844
Mds-Fs-Total-count: 13
Mds-Fs-Total-freeMB: 86021
Mds-Fs-Total-sizeMB: 219813
Mds-Host-hn: mileva.hpc.odu.edu
Mds-keepto: 20070717140620Z
Mds-Memory-Ram-freeMB: 887
Mds-Memory-Ram-sizeMB: 2010
Mds-Memory-Ram-Total-freeMB: 887
Mds-Memory-Ram-Total-sizeMB: 2010
Mds-Memory-Vm-freeMB: 21
--More--
```

Figure EX-20. Compute resource information via grid-info-search.

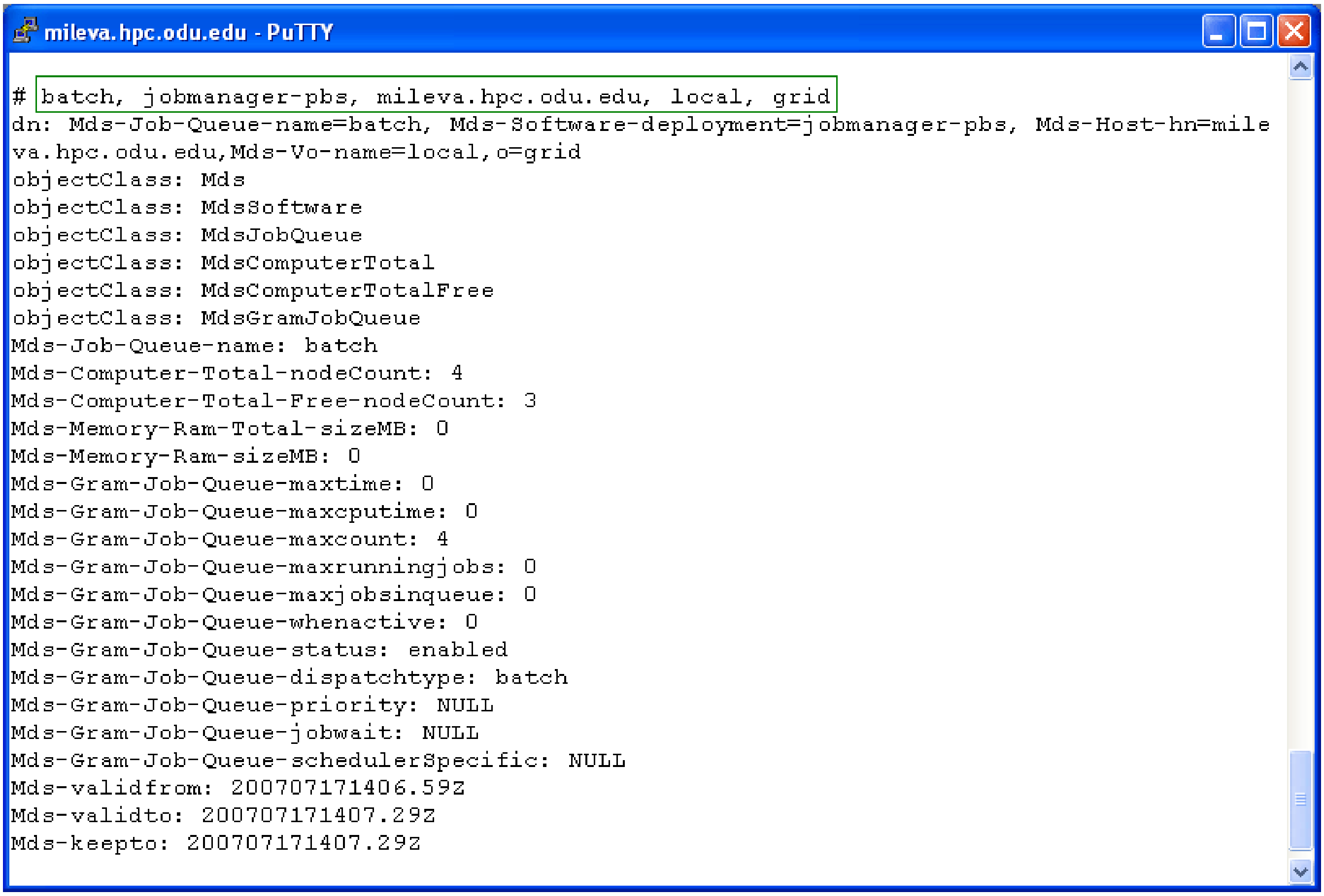
A fairly broad search includes job manager information



```
# jobmanager-pbs, mileva.hpc.odu.edu, local, grid
dn: Mds-Software-deployment=jobmanager-pbs, Mds-Host-hn=mileva.hpc.odu.edu,Mds-Vo-name
=local,o=grid
objectClass: Mds
objectClass: MdsSoftware
objectClass: MdsService
objectClass: MdsServiceGram
objectClass: MdsComputer
objectClass: MdsOs
Mds-Software-deployment: jobmanager-pbs
Mds-Service-type: x-gram
Mds-Service-hn: mileva.hpc.odu.edu
Mds-Service-port: 2119
Mds-Service-url: x-gram://mileva.hpc.odu.edu:2119/jobmanager-pbs:/O=ODU/OU=ODUgrid/OU=
ODUgrid-sly.hpc.odu.edu/CN=host/mileva.hpc.odu.edu
Mds-Service-protocol: 0.1
Mds-Computer-isa: x86_64
Mds-Os-release: 2.6.9-22.ELsmp
Mds-Os-name: Linux
Mds-Computer-manufacturer: unknown
Mds-Service-Gram-schedulertype: pbs
Mds-validfrom: 200707171406.59Z
Mds-validto: 200707171407.29Z
Mds-keepto: 200707171407.29Z
```

Figure EX-21. Job manager information.

and queue information.

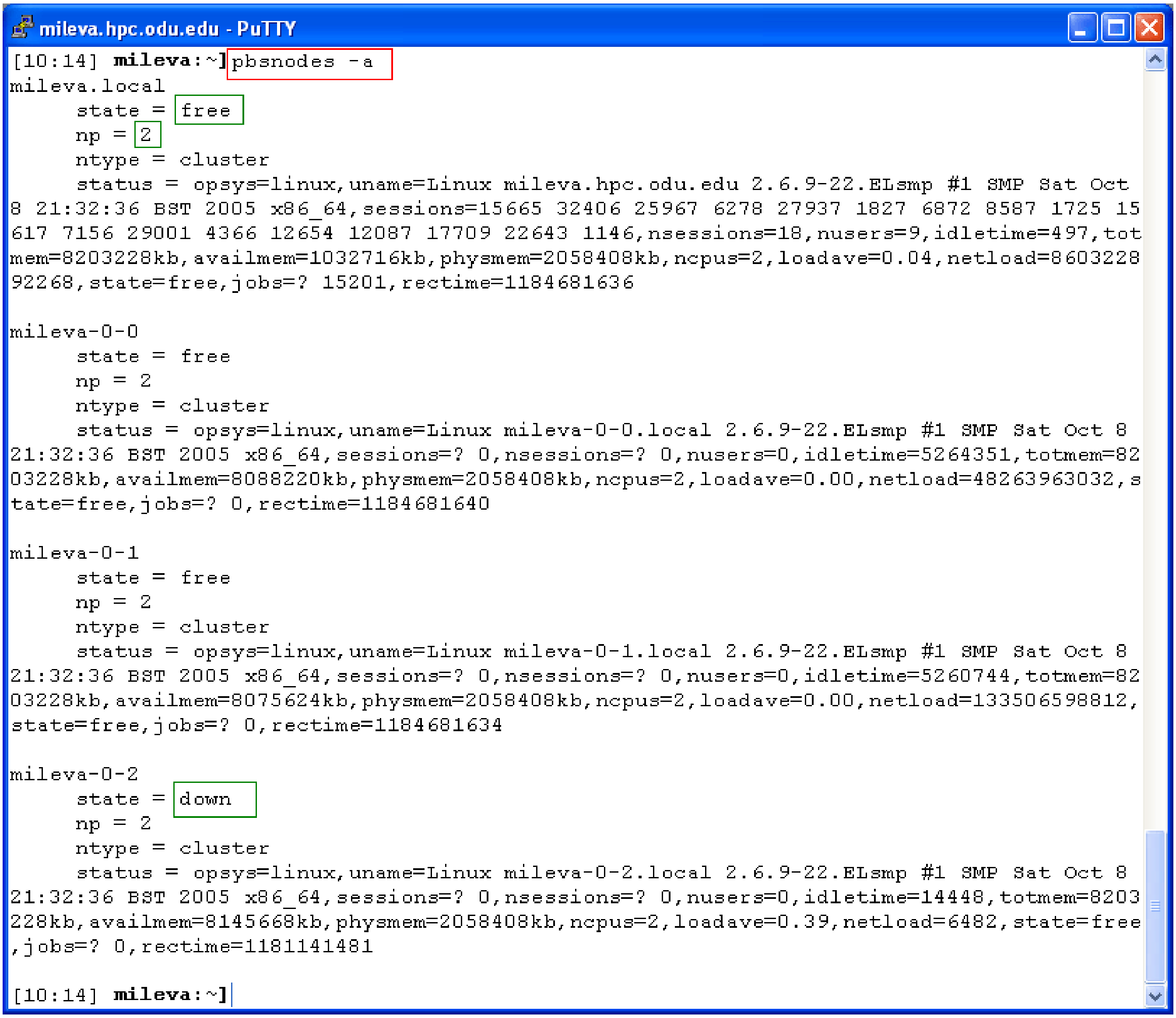


```
# batch, jobmanager-pbs, mileva.hpc.odu.edu, local, grid
dn: Mds-Job-Queue-name=batch, Mds-Software-deployment=jobmanager-pbs, Mds-Host-hn=mile
va.hpc.odu.edu,Mds-Vo-name=local,o=grid
objectClass: Mds
objectClass: MdsSoftware
objectClass: MdsJobQueue
objectClass: MdsComputerTotal
objectClass: MdsComputerTotalFree
objectClass: MdsGramJobQueue
Mds-Job-Queue-name: batch
Mds-Computer-Total-nodeCount: 4
Mds-Computer-Total-Free-nodeCount: 3
Mds-Memory-Ram-Total-sizeMB: 0
Mds-Memory-Ram-sizeMB: 0
Mds-Gram-Job-Queue-maxtime: 0
Mds-Gram-Job-Queue-maxcputime: 0
Mds-Gram-Job-Queue-maxcount: 4
Mds-Gram-Job-Queue-maxrunningjobs: 0
Mds-Gram-Job-Queue-maxjobsinqueue: 0
Mds-Gram-Job-Queue-whenactive: 0
Mds-Gram-Job-Queue-status: enabled
Mds-Gram-Job-Queue-dispatchtype: batch
Mds-Gram-Job-Queue-priority: NULL
Mds-Gram-Job-Queue-jobwait: NULL
Mds-Gram-Job-Queue-schedulerSpecific: NULL
Mds-validfrom: 200707171406.59Z
Mds-validto: 200707171407.29Z
Mds-keepto: 200707171407.29Z
```

Figure Ex 22. Queue information.

The Globus Monitoring and Discovery Service (MDS) shows us a PBS job manager on host milewa.hpc.odu.edu. The queue name is "batch" which includes 4 nodes, no maximum CPU time, and so forth.

We can take a further look at the nodes

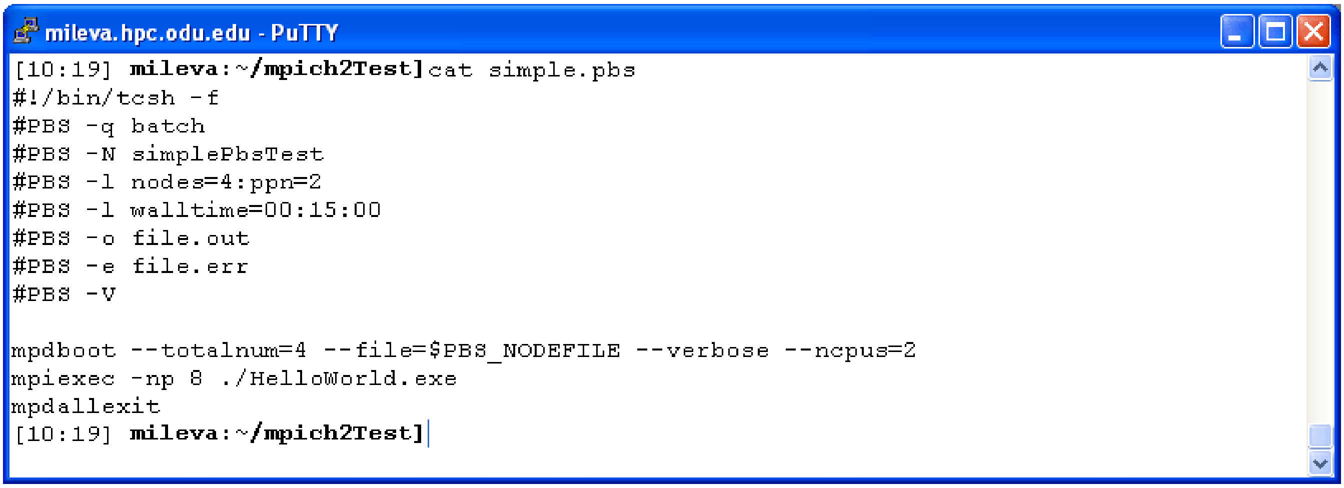


```
[10:14] mileva:~]pbsnodes -a
mileva.local
     state = free
     np = 2
     ntype = cluster
     status = opsys=linux,uname=Linux mileva.hpc.odu.edu 2.6.9-22.ELsmp #1 SMP Sat Oct
8 21:32:36 BST 2005 x86_64,sessions=15665 32406 25967 6278 27937 1827 6872 8587 1725 15
617 7156 29001 4366 12654 12087 17709 22643 1146,nsessions=18,nusers=9,idletime=497,tot
mem=8203228kb,availmem=1032716kb,physmem=2058408kb,ncpus=2,loadave=0.04,netload=8603228
92268,state=free,jobs=? 15201,rectime=1184681636

mileva-0-0
     state = free
     np = 2
     ntype = cluster
     status = opsys=linux,uname=Linux mileva-0-0.local 2.6.9-22.ELsmp #1 SMP Sat Oct 8
21:32:36 BST 2005 x86_64,sessions=? 0,nsessions=? 0,nusers=0,idletime=5264351,totmem=82
03228kb,availmem=8088220kb,physmem=2058408kb,ncpus=2,loadave=0.00,netload=48263963032,s
tate=free,jobs=? 0,rectime=1184681640

mileva-0-1
     state = free
     np = 2
     ntype = cluster
     status = opsys=linux,uname=Linux mileva-0-1.local 2.6.9-22.ELsmp #1 SMP Sat Oct 8
21:32:36 BST 2005 x86_64,sessions=? 0,nsessions=? 0,nusers=0,idletime=5260744,totmem=82
03228kb,availmem=8075624kb,physmem=2058408kb,ncpus=2,loadave=0.00,netload=13350659881 2,
state=free,jobs=? 0,rectime=1184681634

mileva-0-2
     state = down
     np = 2
     ntype = cluster
     status = opsys=linux,uname=Linux mileva-0-2.local 2.6.9-22.ELsmp #1 SMP Sat Oct 8
21:32:36 BST 2005 x86_64,sessions=? 0,nsessions=? 0,nusers=0,idletime=14448,totmem=8203
228kb,availmem=8145668kb,physmem=2058408kb,ncpus=2,loadave=0.39,netload=6482,state=free
,jobs=? 0,rectime=1181141481

[10:14] mileva:~]
```

Figure EX-23. Compute node detail.

including their state and some fairly specific hardware information.

Now its time to submit a job. This grid is using the Portable Batch System (PBS) as its scheduler and therefore a PBS script is prepared to describe the job. This script is in file simple.pbs.

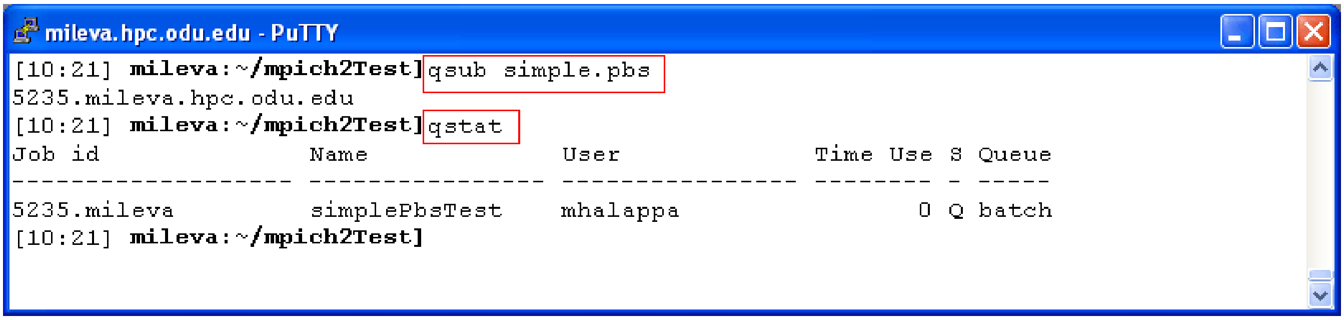


```
[10:19] mileva:~/mpich2Test]cat simple.pbs
#!/bin/tcsh -f
#PBS -q batch
#PBS -N simplePbsTest
#PBS -l nodes=4:ppn=2
#PBS -l walltime=00:15:00
#PBS -o file.out
#PBS -e file.err
#PBS -V

mpdboot --totalnum=4 --file=$PBS_NODEFILE --verbose --ncpus=2
mpiexec -np 8 ./HelloWorld.exe
mpdallexit
[10:19] mileva:~/mpich2Test]
```

Figure EX-24. PBS job submission script.

This job's name is simplePbsTest and will use 2 processors on each of 4 nodes. It will run 15 minutes and standard output will go to simplePbsTest.o<JobID> and standard error will go to simplePbsTest.e<JobID>. The executable is in the current directory and is named HelloWorld.exe. Based on the use of mpiexec, we know this application has been developed using the MPI library, which handles the program and data distribution and communication across the nodes.

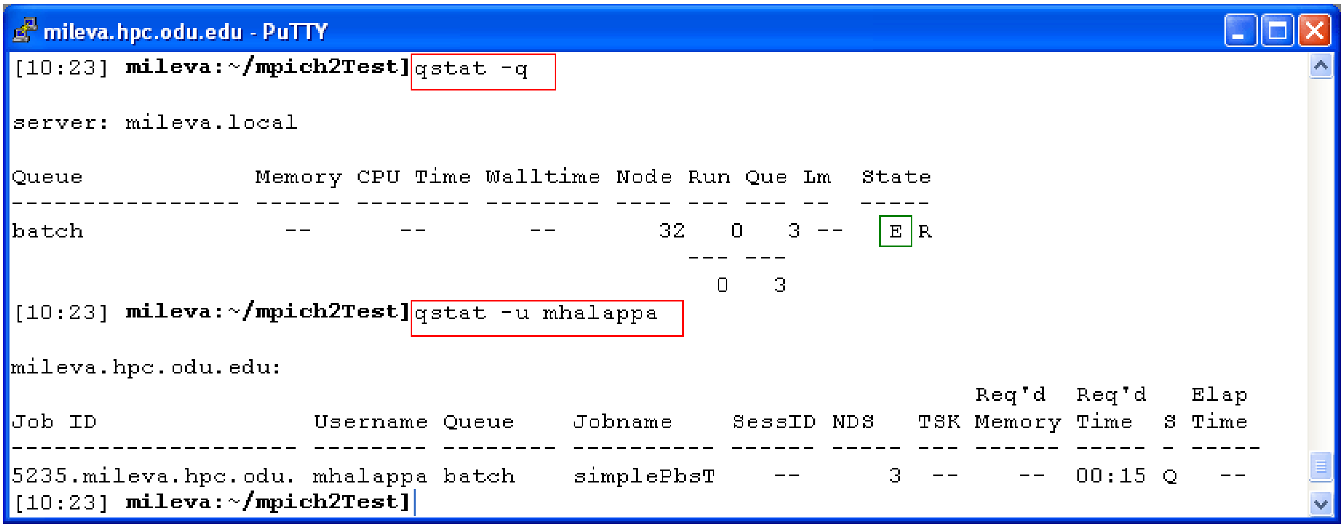
The PBS *qsub* command submits the job to the batch queue.



```
[10:21] mileva:~/mpich2Test]qsub simple.pbs
5235.mileva.hpc.odu.edu
[10:21] mileva:~/mpich2Test]qstat
Job id              Name             User             Time Use S Queue
------------------- ---------------- ---------------- -------- - -----
5235.mileva         simplePbsTest    mhalappa                0 Q batch
[10:21] mileva:~/mpich2Test]
```

Figure EX-25. Job submission information.

The Job ID "5235" can be used to track the job. For example, the PBS *qstat* command will show us the status of the job as it progresses through the machine and/or grid.

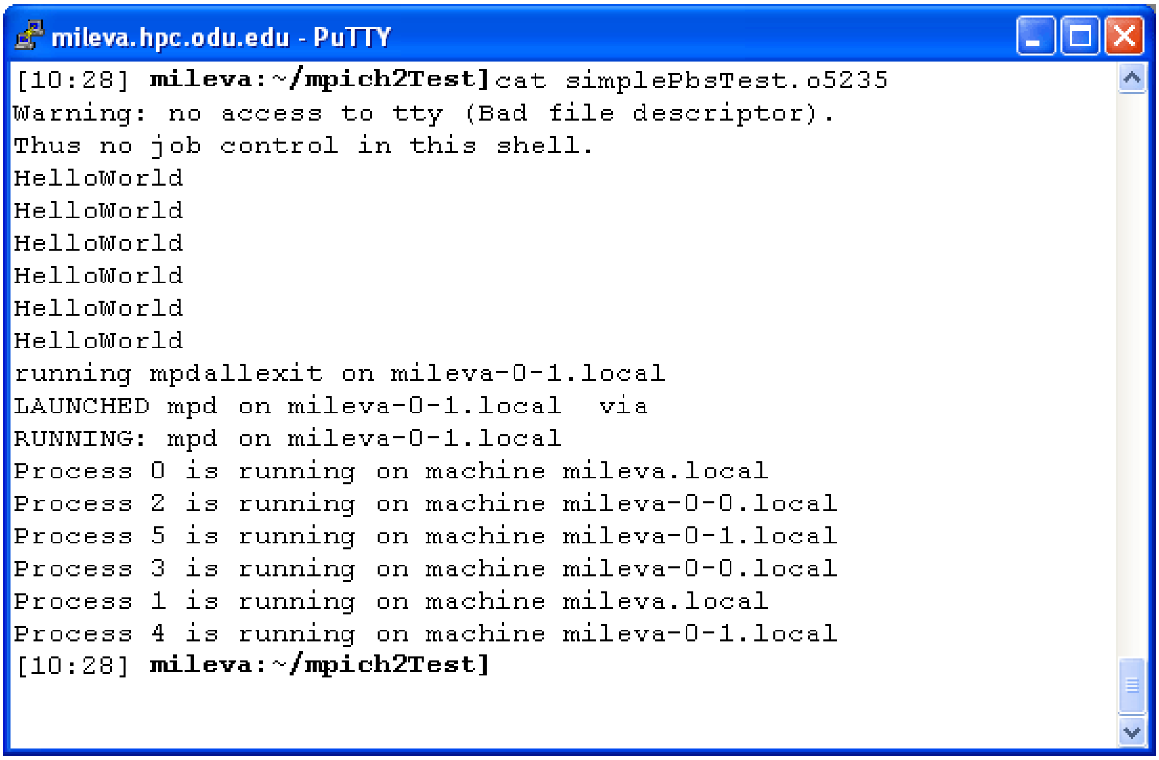


```
[10:23] mileva:~/mpich2Test]qstat -q

server: mileva.local

Queue            Memory CPU Time Walltime Node Run Que Lm  State
---------------- ------ -------- -------- ---- --- --- --  -----
batch              --      --       --      32   0   3 --    E R
                                               --- ---
                                                 0   3
[10:23] mileva:~/mpich2Test]qstat -u mhalappa

mileva.hpc.odu.edu:
                                                                   Req'd  Req'd   Elap
Job ID               Username Queue    Jobname    SessID NDS   TSK Memory Time  S Time
-------------------- -------- -------- ---------- ------ ----- --- ------ ----- - -----
5235.mileva.hpc.odu. mhalappa batch    simplePbsT     --     3  --     -- 00:15 Q   --
[10:23] mileva:~/mpich2Test]
```

Figure EX-26. Job status information.

At this point the job has not yet started but is queued to run.

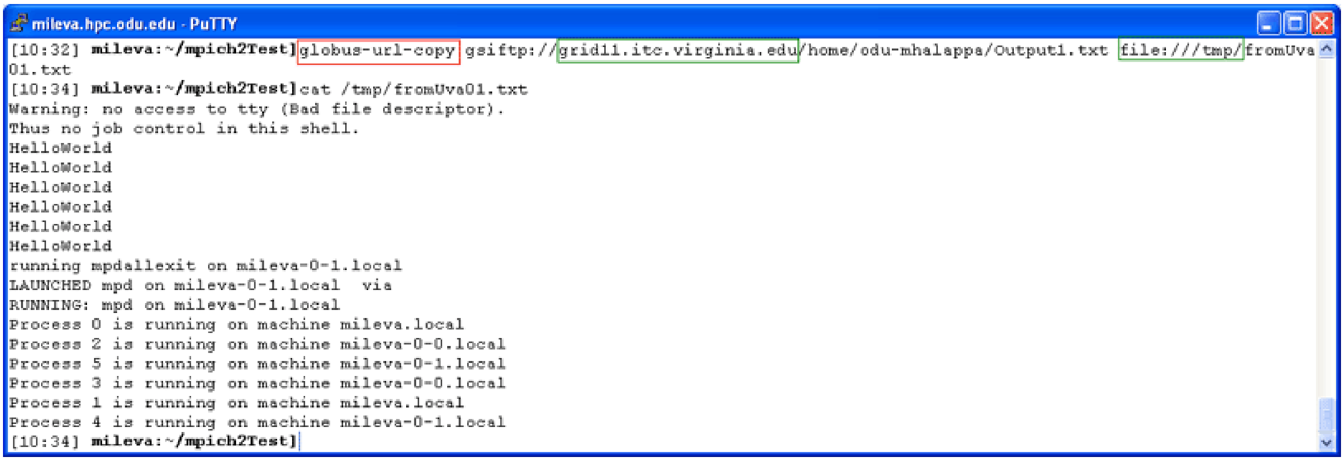
Once the job runs, we can look at the output file, simplePbsTest.o5235.



```
[10:28] mileva:~/mpich2Test]cat simplePbsTest.o5235
Warning: no access to tty (Bad file descriptor).
Thus no job control in this shell.
HelloWorld
HelloWorld
HelloWorld
HelloWorld
HelloWorld
HelloWorld
running mpdallexit on mileva-0-1.local
LAUNCHED mpd on mileva-0-1.local  via
RUNNING: mpd on mileva-0-1.local
Process 0 is running on machine mileva.local
Process 2 is running on machine mileva-0-0.local
Process 5 is running on machine mileva-0-1.local
Process 3 is running on machine mileva-0-0.local
Process 1 is running on machine mileva.local
Process 4 is running on machine mileva-0-1.local
[10:28] mileva:~/mpich2Test]
```

Figure EX-27. Job output file.

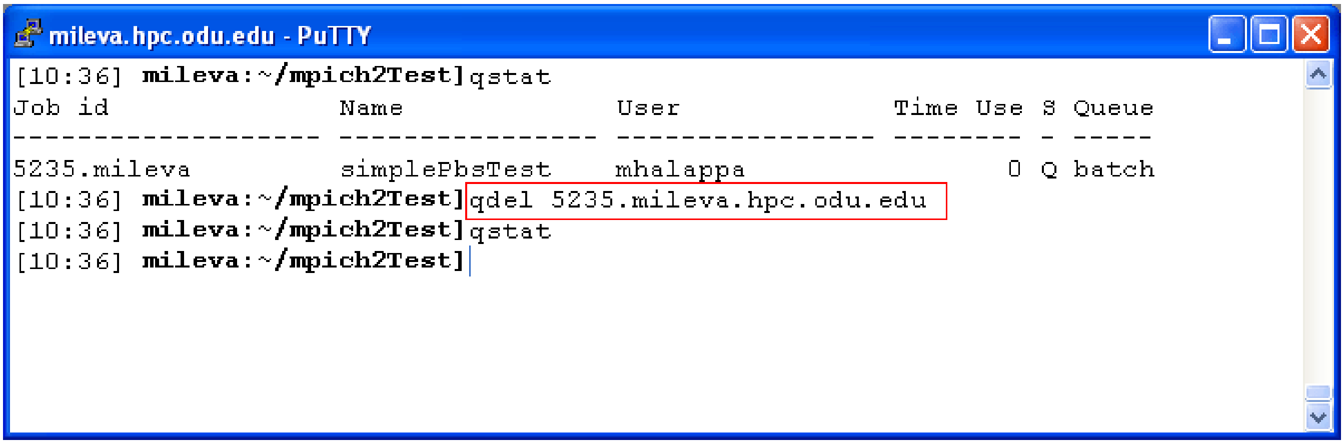
(Aha! Yet another use of the classic "Hello World" example!) At this point the output file is still on the grid's file (gridftp) server. The *globus-url-copy* command provides the ability to transfer files to, from, or between gridftp servers. In this example, the output file has been transferred from mileva to a file BioSimTestRun1.txt in directory /tmp on the local workstation.



```
[10:32] mileva:~/mpich2Test]globus-url-copy gsiftp://grid11.itc.virginia.edu/home/odu-mhalappa/Output1.txt file:///tmp/fromUva
01.txt
[10:34] mileva:~/mpich2Test]cat /tmp/fromUva01.txt
Warning: no access to tty (Bad file descriptor).
Thus no job control in this shell.
HelloWorld
HelloWorld
HelloWorld
HelloWorld
HelloWorld
HelloWorld
running mpdallexit on mileva-0-1.local
LAUNCHED mpd on mileva-0-1.local  via
RUNNING: mpd on mileva-0-1.local
Process 0 is running on machine mileva.local
Process 2 is running on machine mileva-0-0.local
Process 5 is running on machine mileva-0-1.local
Process 3 is running on machine mileva-0-0.local
Process 1 is running on machine mileva.local
Process 4 is running on machine mileva-0-1.local
[10:34] mileva:~/mpich2Test]
```

Figure EX-28. Transferring files back home.

Lastly, what if we realize a job we submitted includes errors and we want to delete it? The PBS *qdel* command will take care of that.



```
[10:36] mileva:~/mpich2Test]qstat
Job id                    Name             User             Time Use S Queue
------------------------- ---------------- ---------------- -------- - -----
5235.mileva               simplePbsTest    mhalappa                0 Q batch
[10:36] mileva:~/mpich2Test]qdel 5235.mileva.hpc.odu.edu
[10:36] mileva:~/mpich2Test]qstat
[10:36] mileva:~/mpich2Test]
```

Figure EX-29. Deleting a job.

A followup *qstat* shows there are no more jobs running or queued for our user.

# Bibliography

[1] Alfred E Neuman (http://www.answers.com/topic/alfred-e-neuman)
[2] About SURAGrid (http://www.sura.org/programs/sura_grid.html)
[3] SCOOP institutions (http://violet.itsc.uah.edu:8080/gridsphere/gridsphere?cid=partners)
[4] Office of Naval Research/ (http://www.onr.navy.mil/)
[5] NOAA's Coastal Services Center (http://www.csc.noaa.gov/)
[6] U.S. Ocean Action Plan/ (http://ocean.ceq.gov/actionplan.pdf)
[7] Global Earth Observation System of Systems (http://www.epa.gov/geoss/)
[8] Integrated Earth Observing System (http://www.noaa.gov/lautenbacher/oceanology.htm)
[9] SURA Coastal Ocean Observing and Prediction (SCOOP) Program< (http://scoop.sura.org/)

# Related Topics

Over the course of cookbook development, we collected material that is connected to the grid topic though not necessarily in the context of version one of this cookbook. We present it here for your reading as time permits.

# Networks and grids

Grids are predicated on the existence of persistent network connections between grid nodes. The grid concept of sharing resources, like storage and computing cycles, via the network is analogous to the idea of sharing information via the Web. Neither the Grid nor the Web would be developing without a broadly deployed, reliable, worldwide network. Ubiquitous high performance networks are often a requirement for high performance computing on the grid, but slower network connections can accommodate some types of applications, particularly if network parameters are considered in resource selection and scheduling.

In a grid, networks serve as the virtual bus for the distributed colleciotn of resources that are orchestrated through grid middleware and are central to the efficient, effective operation of the entity as a whole.. Because of this, understanding networks and how they interact with grid systems is an important part of developing, deploying and managing a grid infrastructure.

## General concepts

***Network Reference Model and Terminology***

Networks have their own terminology and we first introduce some of the important concepts and terms.

The Open Systems Interconnection (OSI [1]) reference model provides a layered abstract description of the computer/network communication model. TCP/IP is the network protocol that enables today's Internet. Although TCP/IP is not a strict implementation of the OSI model (for instance, some applications can extend their functionality beyond the application layer), it is useful to consider network functionality in terms of the model, and its implications for grid operation and performance. Briefly the seven layers of the OSI model, from "lowest" or most fundamental to "highest" are:

1. Physical Layer (example: Optical fiber)
2. Data Link Layer (example: Ethernet)
3. Network Layer (example: Internet Protocol, or IP, in a TCP/IP network)
4. Transport Layer (example: Transmission Control Protocol, or TCP, in a TCP/IP network)
5. Session Layer (example: NetBIOS, named pipes)
6. Presentation Layer (example: ASCII or MPEG encoding)
7. Application Layer (example: GridFTP)

All layers can have impact on the end-to-end performance of applications, but layers 1-4 are typically associated most closely with the network.

***Details of the OSI Reference Model***

Basic network functionality involves the transmission of information, or data, from a source to a destination using some addressing scheme. The information is sent by some application (OSI layer 7) using some encoding (layer 6), perhaps within some session (layer 5) and delivered to the "network" (layers 1-4) which, from the application view, "transports" the information to the destination. This section focuses on layers 1 through 4 and considers details as related to IP (Internet Protocol) networks, as the underlying network for grids as covered in this Cookbook.

### Transport Layer (4)

At the transport layer a protocol is chosen to manage the delivery of the source information to the destination. There are a number of possible services that can exist at this level, although none of them are required:

- Connection oriented
- Ordered delivery
- Reliable delivery
- Flow control
- Ports

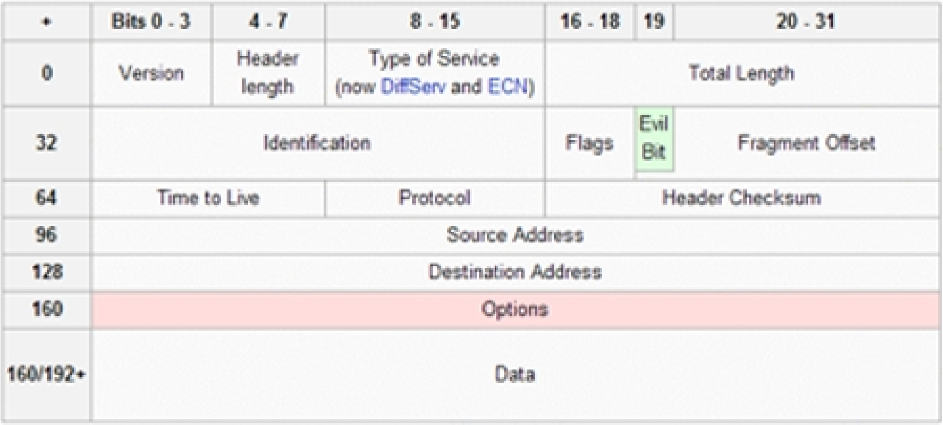
The transmitted data is broken into "packets" of potentially varying sizes by the selected transport protocol. Two typical choices are TCP [2] (Transmission Control Protocol) and UDP [3] (User Datagram Protocol). TCP guarantees delivery of packets of information in the order that they were sent. UDP is a lighter weight alternative that doesn't provide any guarantees of delivery or ordering but is faster and more efficient than TCP. Because of lack of feedback inherent in UDP, however, there is no way for UDP traffic to "fairly" share network bandwidth which is a primary concern of TCP.

There are some hardware devices that operate at layer 4. For example web server load balancing devices are used to distribute web page requests amongst many possible servers depending upon their current loading.

### Network Layer (3)

Both UDP and TCP (and other transport protocols) rely on the network level (layer 3) to address and transmit information. The network layer addresses messages and translates logical addresses into physical ones. It is responsible for the end-to-end delivery of packets.

One of the most broadly used protocols at the network layer is IP (Internet Protocol) of which IPv4 [4] is the most widespread. The packet structure of an IPv4 packet is shown in Figure NG-1.



| + | Bits 0 - 3 | 4 - 7 | 8 - 15 | 16 - 18 | 19 | 20 - 31 |
|---|---|---|---|---|---|---|
| 0 | Version | Header length | Type of Service (now DiffServ and ECN) | Total Length | | |
| 32 | Identification | | | Flags | Evil Bit | Fragment Offset |
| 64 | Time to Live | | Protocol | Header Checksum | | |
| 96 | Source Address | | | | | |
| 128 | Destination Address | | | | | |
| 160 | Options | | | | | |
| 160/192+ | Data | | | | | |

Figure NG-1. The format of an IPv4 packet (See RFC 3514 [5] for definition of the "Evil Bit")

The typical layer 3 device on networks is the router (and also "layer-3 switches"). The details of the packet structure are explained in RFC 791 [6]. Some important details about the header:

- *Version* is the version of the internet header.
- *IHL* is the length of the internet header in 32 bit words (minimum of 5 as shown in Figure NG-1 above).
- *Type of Service* is used to indicate abstract parameters of the quality of service desired or more

recently to indicate ECN (See RFC 3168).

- *Total Length* is 16 bits defining the entire packet size (including header and data) in bytes.

For further details see the Wikipedia entry on IPv4.

### Data Link Layer (2)

The data transport layer (layer 2) manages the node-to-node (or hop to hop) packet delivery. One typical example of layer 2 data transport is Ethernet. The framing of Ethernet data is shown in Figure NG-2. The higher level layers are encapsulated inside the layer 2 framing. The switch is a typical layer 2 device.

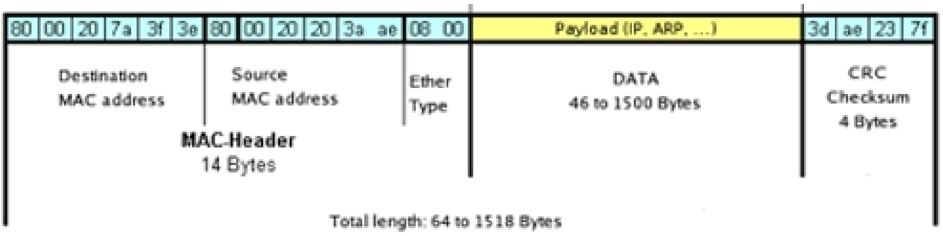


Figure NG-2. Details of an Ethernet Type II frame format.

Good general explanations of the Ethernet [7] and the Data link layer [8] can be found in Wikipedia.

### Physical Layer (1)

The physical layer denotes the actual physical cabling, wireless electromagnetic connection or optically modulated carrier that transports bits in the network.

This layer deals with contention resolution, flow control, initiation and termination of connections and conversion between digital data representations.

Typical devices are Ethernet hubs, optical transponders, wireless access points, etc.

### Summarizing

Application data destined for a remote network location undergoes a number of steps before being transmitted on the physical layer. Figure NG-3 shows how fragmented data from an application is encapsulated by various headers and trailers, and at various levels in the OSI network model.

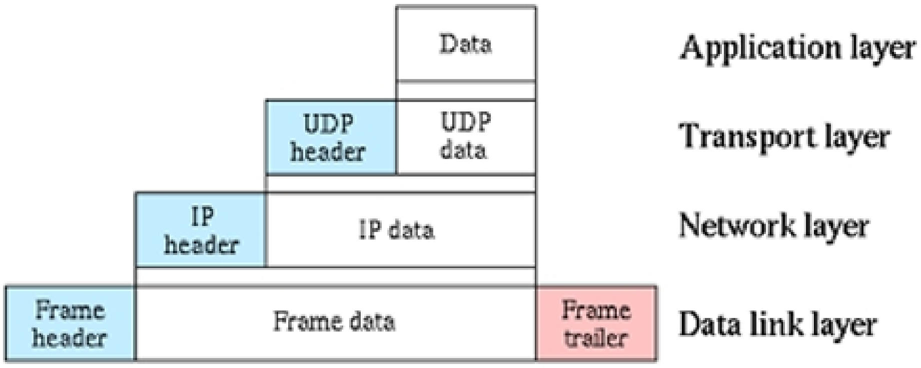


Figure NG-3. OSI data encapsulation.

The figure above shows how application data is progressively encapsulated at each level for transmission on the physical layer. This encapsulation has implications for how networks perform that we will discuss in following sections.

***Some important functional considerations***

Now that we have an overview of how networks deliver data between source and destination we can discuss some of the important functional considerations that impact how the network performs.

Typically applications have widely varying data sizes which that need to be sent across the network. However, the underlying network layers present their own constraints on the data block sizes they can transport per packet. This means that depending upon the amount of data to be sent multiple packets or frames may be required for a given application data block. One potential way to optimize traffic flow is to maximize the size of the data content of each packet, which is discussed below.

### Frame/Packet Size and Rate

Ethernet (layer 2) typically has a constraint on the maximum frame size that can be transmitted. The limit is denoted as the MTU (Maximum Transmission Unit) and is typically 1500 bytes. This has an implication for data transmission. If a large file is being sent across the network there will be many Ethernet frames required to transport the data. Each frame can require the receiving NIC (network interface card) to issue an interrupt to the local processor so it can move the received data. The rate of interrupts depends upon the data arrival speed (network bandwidth). At slower NIC speeds (Ethernet = 10 Mbits/sec or Fast Ethernet 100 Mbits/sec) this is not a daunting challenge for modern processors. However at higher speeds (Gigabit or 10 Gigabit Ethernet) the load can bring even the fastest current processors to their knees and reduce the overall network throughput achievable to well below the "wire-speed".

There exist a number of means of overcoming high-speed limitations:

1. Increase the layer-2 frame size (Jumbo Frames — Wikipedia [9] and Wareonearth [10].)
2. Use options in the NIC/drivers to coalesce multiple frames into a single interrupt to the processor (typically controlled by 'ethtool' [11], see the ethtool -C options)
3. Use NIC options to offload packet processing from the CPU (again, see ethtool options -G and -K)
4. Tune your system TCP stack for the type of connections you need to optimize (see Enabling High Performance Data Transfers [12] or TCP Tuning Guide [13] for details)
5. Purchase newer NICs that have significant improvements in reducing the load on the processor.

### Network Stacks and TCP

Another consideration is the use of TCP across long, high bandwidth networks. TCP was designed when typical high-speed network connections were Ethernet (10 Mbits/sec) and typical networks were "local" and only rarely regional or national in size. It is perhaps understandable that this protocol does not scale well to gigabit and beyond network speeds over national or international distance scales. Tuning the TCP network stack parameters can help ameliorate poor WAN performance for TCP applications and is often required to achieve any kind of reasonable WAN performance.

TCP, because of its widespread use, plays an important role in how well many networked applications function. Most operating systems "out of the box" have their TCP implementation's tuned for LANs. This is partly historical (most applications used to be "local") and partly to minimize resource consumption (tuning network stacks for WAN performance can significantly increase memory consumption because all network connections may use more resources).

Newer operating systems are doing better at having default configurations which are better optimized for networked use. Windows XP/2003 and Linux (2.6.9+ kernels) have significantly improved in their network tunings. Linux now supports "autotuning" of stacks and allows selecting different variants of TCP (2.6.12+ kernels).

### Hosts and Network Performance

A last functional consideration involves the hosts themselves. Many times users will blame "the network" for poor application performance when many times the problem does not originate with the network. Applications that use the network must be considered as an end-to-end system and problems can arise in many different areas from poorly designed applications (especially how such applications interact with the network itself), high host CPU load, low free host memory, deficient or mis-configured host storage subsystems, buggy or badly designed device drivers, out-of-date firmware, poorly tuned network stacks, badly designed or defective NICs or faulty cabling. Ruling out all these issues on the local host still doesn't necessarily implicate the network since those same issues may be affecting the remote host.

Of course there are times when the network is the problem. Congested or mis-configured local-area, campus, regional or backbone networks cause significant disruption to networked applications.

This is important when considering network tuning and monitoring for grids. To help manage and optimize networks for grid use will require careful attention on the whole "end-to-end" problem and not just the network in isolation. Being able to determine if the problem is at either end or the network in the middle is critical to quickly resolving the problem.

#### *Network components and operation*

- NICs and Hosts – Network interface cards (NICs) provide hosts with access to the network. Typically these cards encode their information via Ethernet at layer 2 (either wired or wireless) with speeds from 10-10000 Mbits/sec. The wired versions typically come in copper or fiber (optical) variants.
- Switches – These are the "layer 2" devices with 2 or more ports responsible for interconnecting multiple network devices (hosts, switches, etc). Switches "learn" the hardware addresses (MAC for Ethernet) of the devices connected to each of their ports and can switch layer 2 packets coming in one port to the "correct" destination port. Newer switches are typically "non-blocking" meaning that all their ports allow wire-speed, full duplex interconnections (each port can "talk" to a partner port full-duplex at the same times as all other port pairs are doing this).
- Routers – These are the "layer 3" devices with 2 or more ports responsible for "routing" (determining the best path for) IP packets across the network. Routers are aware of the various networks connected to their ports and can route incoming IP packets to the correct destination port.
- Optical Components – Many newer switches and routers utilize optical rather then electrical interconnects. Instead of copper cabling connecting to a port, fiber carrying modulated light is used to transmit information. Light pulses can propagate further without degradation compared to electrical signaling on copper cables. Since almost all existing switches and routers utilize electronic components internally to fulfill their roles, network information encoded in modulated light must be translated into electrical signals for processing (and then perhaps converted back to light for transmission). The process is referred to as OEO (Optical-Electrical-Optical). Gigabit speed optical components are called GBICs (GigaBit Interface Convert) and typical 10 Gigabit components are called Xenpaks. Both come in a variety of physical layer interfaces (single-mode, multi-mode, extended reach, short range, long range, etc.). Note that there are also optical "switches" which can connect light from a source fiber to a destination fiber through the use of small mirrors on the millisecond timescale.
- Monitoring – Monitoring is an often neglected but vital component of networks and their operation. Being able to track and measure various network data is critical for problem diagnosis and localization, resource planning and network management. Broadly speaking "monitoring" can include

tracking network switch/router configurations, port bandwidth utilization and errors, system logging information (syslog from switches/hosts/routers), network "flow" information and statistics and endhost network usage and errors. There is no uniform "end-to-end" monitoring system for networks that is deployed but there are a number of projects working on providing a lot of this capability: PerfSONAR, MonALISA and others. Also SNMP (Simple Network Management Protocol) is available to provide a standard way of accessing much of the information about the network and the devices which comprise it.

## Measurement and monitoring

*If you don't measure, you don't know.*

Grids consist of many hardware and software components, any of which can break or misbehave. Monitoring and measuring at least portions of the network connections between grid nodes is necessary for reliable operation and support. This section discusses some of the tools available for this.

Whether your concern is monitoring the network itself or the user experience, monitoring from diverse locations is essential to identifying problems. Layers 1 through 3 can be monitored using typical network monitoring and measurement tools, however, we also need to look into the application layer to understand what users are actually trying to do. In addition to getting information to the network manager (ideally before a user would notice the problem), monitoring that can transgress layers greatly benefits the task of root cause analysis.

We describe the family of network monitoring approaches in two categories, passive monitors and active monitors. [18]

- Passive Monitors

  Passive monitors don't add traffic to the network; they just provide a view of what goes by. The major advantage of this of course is that no extra load is generated on the network and servers from the use of the monitoring device itself. If the monitoring is done in enough detail, however, user-perceived performance for network activities such as TCP connections, DNS lookups and file transfers can be gauged. This is significant because, while many of the active approaches (described later) claim to provide a measure of the user experience, they do not involve measurement of actual user activities.

  The disadvantage of passive monitoring is that it becomes more and more difficult to monitor correctly as the volume of data on the network increases. The movement away from true broadcast networks to switching further complicates this situation in that more monitoring points are required in order to "see" all of the traffic. Another problem is the increasing use of encryption , which can hide the actual application details that we want to monitor.

  Passive monitoring generally relies on a promiscuous mode tap that can see all network traffic. This is the classic Remote MONitoring [19] (RMON) approach and can be found in commercial products like TrafficDirector as well as the current GOAT and many other publicly available tools and appliances such as NTOP. These tools are typically deployed at one or more locations on a network (e.g. border gateway, one per subnet). The data is gathered and often brought back to a central server for correlation and analysis.

  In addition to the dedicated monitoring device, there are a number of passive client-based tools that have been developed. These tools focus on the network performance experienced by a single user. A passive monitor, installed on the user's computer, watches network applications as they are being used and reports the performance to a central collection point. From the network and service manager this is ideal as all of the users become "free" network probes. Of course, nothing is ever really free and some performance degradation is likely to be obvious tothe user. The more successful attempts at this have worked to limit the pain. Some examples in this arena are a commercial product called FirstSense and NETI@home [20], an open source package from Georgia Institute of Technology.

- Active Monitors

  Unlike passive monitors, active monitors will generate traffic to perform a measurement. This includes traditional network tests like ping [22] and traceroute [23] but also application tests like file transfers and DNS lookups. The primary advantages of this approach are that it is somewhat easier to implement than the passive scanner and that it is possible for the network administrator to see a problem even before a user would see it. For instance, we can discover that the mail server went down at 4am and get it back up and running before users ever notice there is a problem. The primary disadvantages of active monitoring are the additional load on both network and servers and the fact that we don't actually observe the real user experience but something designed to look like a user. The techniques used can be divided into two groups: real tests and synthetic tests.

  A Real Test Active Monitor is a probe that sits out on the network, either in a dedicated box or on a user's computer, and performs operations with an on-line, production server. This tests not only the network performance but also the complete end-to-end service. The goal is to get as close to the real user's experience as possible. If the probe can do a DNS lookup or get a DHCP lease, then there's a good chance that the user can too. Tools available for use in real test monitoring include the publicly available Nagios and commercial tools from Micromuse.

  An Active Synthetic Test is very similar to a real test in that it performs some real application, such as a file transfer. However, this is not done to the production server but to a collection of dedicated performance testing boxes. There are several of these in the Internet today. Tools such as AMP and PingER are in this category, along with many others. The Iperf tool is often used in this manner. The Ganymede tool is a commercial offering that operates in this way.

  Most implementations of active monitors will break the test down into components. For instance, a web server measurement will include timings for DNS lookup, TCP connection and then detailed application transaction timings (complete order, process credit card, etc.)

  Active monitoring tools include some simple things like:

  - ping: The name of this tool comes from the game "ping pong" where the source site sends a packet of some size to a destination site. The source site measures how long it takes the destination site to return the packet and determines if there was any data loss in the returned packet. ping is very helpful in telling if the destination site is down, unreachable for some reason, or if the network between is causing delays or transmission issues of any kind. You can try the command *ping internet2.edu* from a Unix workstation to see how it works; other operating systems typically offer the same or a similar command. (Note: Some system administrators disable ping acknowledgements from their machines if they feel the network traffic it generates is unbearable.)
  - traceroute: The traceroute tool does just that - it traces the route between source and destination IP sites. The segment between each two sites along the path is called a "hop". Traceroute will list each hop IP address (and hostname if available) along with three sample test times. Viewing the route and the test times can be very informative. For example, you may find that the route taken is not what you expected which can indicate a network outage in your usual route. You may find serious delays (which will be noted with asterisks or test timeouts.) And you may find many more hops than you expected to see denoting possible rerouting or path problems. Try a *traceroute sura.org* to see how this cookbook material reaches you!

- iperf: iperf measures TCP and UDP bandwidth performance. NLANR's Iperf [24] tool reports bandwidth, delay jitter, and datagram loss.

Internet2 has also developed several advanced tools for network measurement:

- owamp: One-Way Active Measurement Protocol (OWAMP) [25] is a command line client application and a policy daemon used to determine one way latencies between hosts and is an implementation of the standard of the same name. The one-way measurements performed by OWAMP help to determine the direction of the congestion (note that the route there is not always the same as the route back.)
- bwctl: Bandwidth Test Controller (BWCTL) [26] is a command line client application and a scheduling and policy daemon for using Iperf. BWCTL does things like arrange Iperf tests between different servers on different systems, request and reserve specific types of tests, and streamline multipoint tests via configuration options for administrators.
- ndt: Network Diagnostic Tool (NDT) [27] is a client/server program that provides network configuration and performance testing to a users desktop or laptop computer. NDT will look for things like duplex mismatch conditions on Ethernet/FastEthernet links, incorrectly set TCP buffers in the user’s computer, or problems with the local network infrastructure. A multi-level series of plain language messages, suitable for novice users, and detailed test results, suitable for a network engineer, are generated and available to the user (test results may be easily emailed to the appropriate administrator to assist in the problem resolution phase as well.)

Of course, the real value of all of this monitoring is limited unless there is adequate work on the data gathering, correlation and reporting tools. This is where the real analysis is done to determine first whether or not a problem exists and then who to contact to get it resolved.

Other examples of active monitors include:

- Popular monitoring tools

MRTG: The Multi Router Traffic Grapher (MRTG) [28] is a tool to monitor the traffic load on network links. MRTG generates HTML pages containing PNG images which provide a LIVE visual representation of this traffic.

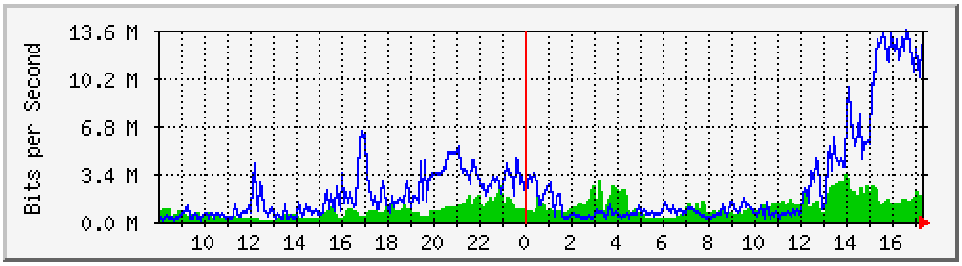
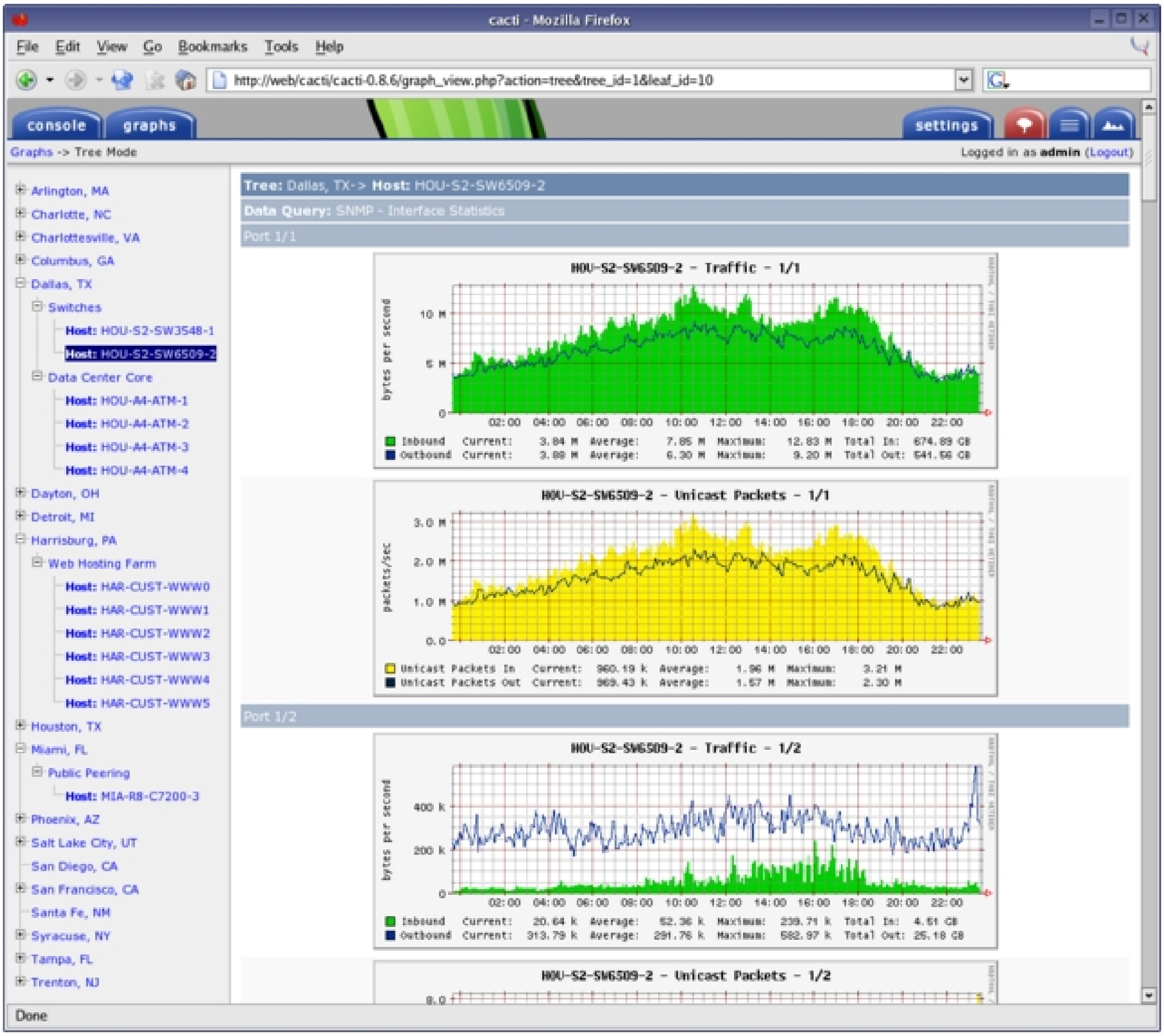


Figure NG-4. A sample MRTG output graph.

Cacti: Cacti, the Complete RRDTool-based Graphing Solution [34] is a very versatile, easy-to-manage tool designed to harness the power of RRDTool's data storage and graphing functionality. It supports a plugin architecture and has been demonstrated to scale to monitoring thousands of hosts. End-systems, network devices and many other types of information can be tracked via Cacti. There are plugins to alert based upon threshold and system events as well as the ability to gather and track MAC address locations in complicated switching environments.



```
🟩 Inbound   Current:    3.84 M  Average:    7.85 M  Maximum:   12.83 M  Total In:  674.89 GB
🟦 Outbound  Current:    3.88 M  Average:    6.30 M  Maximum:    9.20 M  Total Out: 541.56 GB
```

```
🟨 Unicast Packets In   Current:  960.19 k  Average:   1.96 M  Maximum:   3.21 M
⬛ Unicast Packets Out  Current:  969.43 k  Average:   1.57 M  Maximum:   2.30 M
```

```
🟩 Inbound   Current:   20.64 k  Average:   52.36 k  Maximum:  239.71 k  Total In:   4.51 GB
🟦 Outbound  Current:  313.79 k  Average:  291.76 k  Maximum:  582.97 k  Total Out: 25.18 GB
```

Figure NG-5. Cacti Dual Pane Tree View.

OpenView: HP OpenView Network Node Manager Smart Plug-in for IP Multicast [29] is designed specifically to manage the multicast environment. OpenView will:

- ♦ Automatically discover IP multicast routing topology relationships
- ♦ Proactively monitor device health and measure IP multicast traffic flow
- ♦ Rapidly generate alarms based on multicast activity
- ♦ Quickly isolate and fix multicast faults through built-in diagnostic capability.

- Tools for monitoring clusters and servers

ganglia: The Ganglia Monitoring System [30] is a scalable distributed monitoring system for high-performance computing systems such as clusters and Grids. It is based on a hierarchical design targeted at federations of clusters. It leverages widely used technologies such as XML for data representation, XDR for compact, portable data transport, and RRDtool for data storage and visualization.

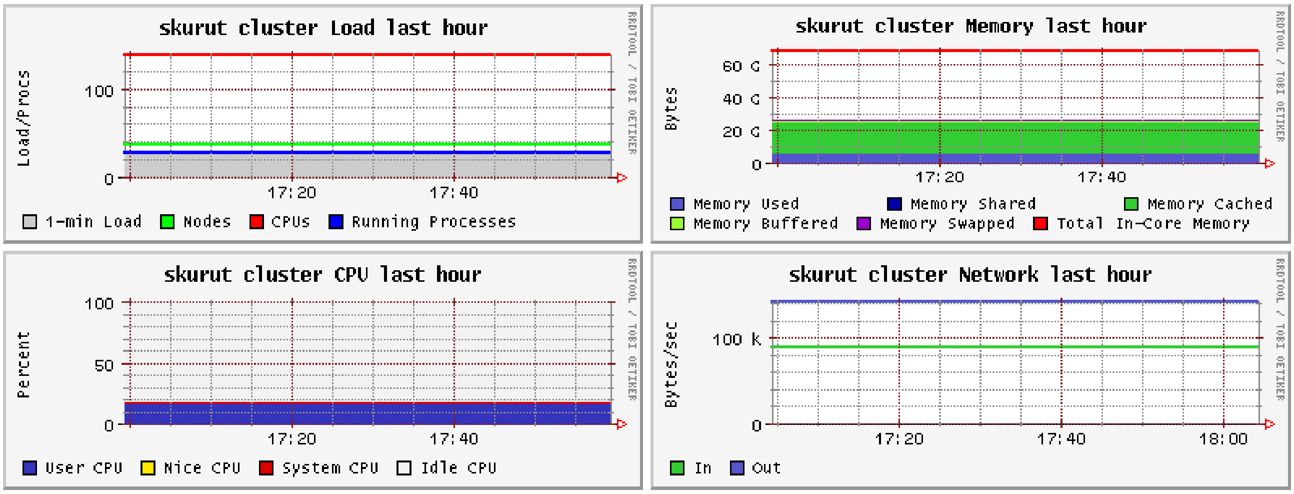


Figure NG-6. Several real-time graphical reports from Ganglia.
(http://lindir.ics.muni.cz/ganglia/?c=skurut%20cluster&m=&r=hour&s=descending&hc=4).

- Comprehensive tools

MonALISA: MONitoring Agents using a Large Integrated Services Architecture (MonALISA) [31] has been developed by Caltech and its partners with the support of the U.S. CMS software and computing program. The framework is based on Dynamic Distributed Service Architecture and is able to provide complete monitoring, control and global optimization services for complex systems. MonALISA provides:

- Distributed Registration and Discovery for Services and Applications.
- Monitoring all aspects of complex systems.
- System information for computer nodes and clusters.
- Network information (traffic, flows, connectivity, topology) for WAN and LAN.
- Monitoring the performance of Applications, Jobs or services.
- End User Systems, and End-To-End performance measurements.
- Can interact with any other services to provide in near real-time customized information based on monitoring information.
- Secure, remote administration for services and applications.
- Agents to supervise applications, to restart or reconfigure them, and to notify other services when certain conditions are detected.
- The Agent system can be used to develop higher level decision services, implemented as a distributed network of communicating agents, to perform global optimization tasks.
- Graphical User Interfaces to visualize complex information.
- Global monitoring repositories for distributed Virtual Organizations.

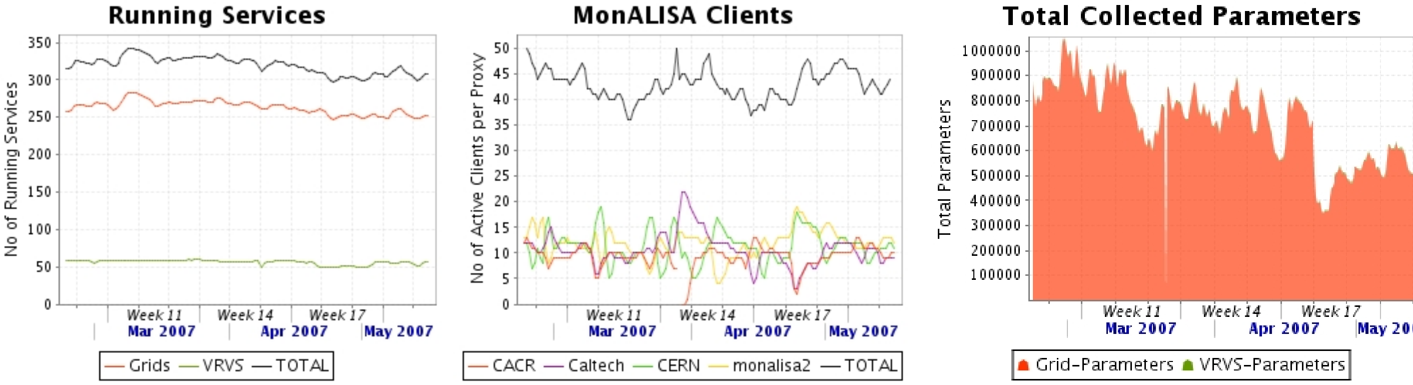


Figure NG-7. Several real-time graph examples from MonALISA global statistics (http://monalisa.cacr.caltech.edu/monalisa__Looking_Glass.htm)

PerfSONAR: PERFormance Service Oriented Network monitoring ARchitecture (PerfSONAR) [32] has three contexts: it is a consortium, a protocol, and a set of code. For our purposes, the last item is most interesting to us in terms of the services developed to act as an intermediate layer, between the performance measurement tools and the diagnostic or visualization applications. Major perfSONAR services include:

- Measurement Point Service: Creates and/or publishes monitoring information related to active and passive measurements.
- Measurement Archive Service: Stores and publishes monitoring information retrieved from Measurement Point Services.
- Lookup Service: Registers all participating services and their capabilities.
- Authentication Service: Manages domain-level access to services via tokens.
- Transformation Service: Offers custom data manipulation of existing archived measurements.
- Resource Protector Service: Manages granular details regarding system resource consumption.
- Toplogy Service: Offers topological information on networks.

An example of PerfSONAR use can be seen at the ESnet PerfSONAR Traceroute Visualizer [33].

# Manpower requirements

### Grid system administration and manpower requirements of a campus-wide grid (Texas Tech University example)

Clear definition of operational policies and procedures provides a foundation for the successful support of a production grid. The examples below from TechGrid, the campus grid of Texas Tech University, illustrate the level of detail that is considered in some key areas of policy and administration, and evolving as needed to support increasing usage and infrastructure development.

This section shall outline the administration requirements of a campus-wide grid, how a grid works, and policies/procedures of TechGrid with respect to events that are grid related in nature such as grid infrastructure failures, planned maintenance, and planned reimaging of given 'zones'.

### Zone Administrators duties (2 hours a week):

Responsibilities:

1. Installing nodes: A script has been provided.

2. Uninstalling nodes: A script has been provided.

3. Configuring nodes: A script has been provided.

4. Testing nodes: A script has been provided.

5. Reimaging: This is a standard ATLC function; however nodes need to be uninstalled before reimaging takes place.

**Campus Grid Administrators duties (20 hours a week):**

Responsibilities:

1. Maintain the Bootstrap server: create scripts to monitor Grid usage and failures.

2. Train Zone Administrators.

3. Write scripts to add functionality to the Grid.

4. Train users.

5. Find more resources to add to the Grid.

6. Help Zone Administrators with Grid related issues.

7. Help students/researchers Grid-enable their code.

6. Installing nodes: Install nodes into new Grid zones.

7. Uninstalling nodes: Help new Grid zones uninstall at first reimaging.

8. Configuring nodes: Help new Grid zones configure their nodes.

9. Testing nodes: Create scripts that test the availability of a node.

**Emergency procedures:**

1. Grid Maintenance: If the Campus Administrator knows when the Grid will go down with enough warning, then the Grid can be gracefully unmounted using an elegant shutdown script that will unmount each individual node in the Grid without affecting quality of services for the end users. Zone Administrators will be told of the Grid shutdown in advance. Grid Zone Administrators will be asked to reset their compute nodes at the end of the day to reactivate the Grid on those nodes.

2. If an emergency shutdown is required, then the Grid will be shutdown without dismounting worker nodes. This case is rare since this type of failure is caused by circumstances beyond Grid Administration control such as power or chiller failure at Reese Center. Emails will be sent to Zone Administrators.

3. Grid Failures: If the Grid goes down without warning, then the next step will be to disable Grid system services on each machine (This is a rare occurrence). It is the zone administrator's duty to inform the Campus Grid Administrator when issues like this arise so that a remedy can be applied immediately.

**In review, the current policies that are in effect:**

1. Grid nodes cannot be used during the day.

2. Grid nodes cannot be used at anytime if processing load is higher than 50%.

3. Grid nodes cannot be used at anytime if anyone is logged into it locally or remotely.

4. Jobs will cease automatically if a user logs in or if the wall clock time of the Grid node displays any time between 7:00AM and 8:00PM

**Contact**

Jerry Perez, Texas Tech University.

URL: http://www.hpcc.ttu.edu/techgrid.html [40]

# Bibliography

[1] OSI Model — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/OSI_model)
[2] Transmission Control Protocol — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Transmission_Control_Protocol)
[3] User Datagram Protocol — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/User_Datagram_Protocol)
[4] IPv4 — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/IPv4)
[5] The Security Flag in the IPv4 Header (http://www.ietf.org/rfc/rfc3514.txt)
[6] RFC 791 (http://tools.ietf.org/html/rfc791)
[7] Ethernet — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Ethernet)
[8] Data_link_layer — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Data_link_layer)
[9] Jumbo Frames — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Jumbo_Frames)
[10] Gigabit Ethernet Jumbo Frames (http://sd.wareonearth.com/~phil/jumbo.html)
[11] ethtool (http://sourceforge.net/projects/gkernel)
[12] Enabling High Performance Data Transfers (http://www.psc.edu/networking/projects/tcptune/)
[13] TCP Tuning Guide (http://dsd.lbl.gov/TCP-tuning/TCP-tuning.html)
[18] Taxonomy of Network and Service Monitoring Approaches (http://www.rnoc.gatech.edu/cpr/taxonomy.html)
[22] ping, From Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Ping)
[23] traceroute, From Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Traceroute)
[24] Iperf — The TCP/UDP Bandwidth Measurement Tool (http://dast.nlanr.net/Projects/Iperf/)
[25] One-Way-Ping (OWAMP) (http://e2epi.internet2.edu/owamp/)
[26] Bandwidth Test Controller (BWCTL) (http://e2epi.internet2.edu/bwctl/)
[27] Network Diagnostic Tool (NDT) (http://e2epi.internet2.edu/ndt/)
[28] Tobi Oetiker's MRTG — The Multi Router Traffic Grapher (http://oss.oetiker.ch/mrtg/)
[29] HP OpenView Network Node Manager Smart Plug-in for IP Multicast (http://www.openview.hp.com/products/mcast/)
[30] Ganglia Monitoring System (http://ganglia.sourceforge.net/)
[31] Monalisa — Monitoring the Grid since 2001 (http://monalisa.cacr.caltech.edu/monalisa.htm)
[32] PERFormance Service Oriented Network monitoring ARchitecture (http://wiki.perfsonar.net/jra1-wiki/index.php/PerfSONAR_About)
[33] ESnet PerfSONAR Traceroute Visualizer (https://performance.es.net/cgi-bin/level0/perfsonar-trace.cgi)
[34] Cacti, the Complete RRDTool-based Graphing Solution (http://cacti.net/)
[40] Texas Tech TechGrid (http://www.hpcc.ttu.edu/techgrid.html)

# Glossary

As you have probably discovered, while there is some crossover with other areas of computing, grid computing has its own set of terms. And there are plenty of them! Thankfully several excellent glossaries exist. Therefore instead of creating another one here, we will provide you with a list to choose from.

- A grid glossary: Top 30 terms
- Grid computing — Wikipedia, the free encyclopedia
- Open Science Grid Glossary and Grid Computing Terms (as used in OSG)
- Glossary — AustrianGrid Wiki
- WorkBookGlossary < Atlas < Twiki
- ATLAS Glossary
- UK Computing for Particle Physics (GridPP) Grid Acronym Soup (GAS)
- BIRN Coordinating Center — Glossary
- Oracle Grid Computing Glossary
- e-Infrastructure Glossary
- Information Society Technologies Glossary of Grid related terms
- GGF OGSA Glossary
- nmi-edit Glossary

Most of these glossaries are listed in alphabetical order. But many of the terms are related in various other ways. Future versions of this cookbook may provide a glossary with terms related, instead, by category, hierarchy, area of science, project, and so forth.

# Appendices

## Related links

Our intention is to keep this Cookbook current and to broaden it in future versions. For example, one goal we have is to show more international activities. (While we spoke with many international groups, to our own disappointment, this turned out to be something we had to cut short as we pushed to get Version 1 out.) In the meantime, we will include these and other important and interesting links here.

### Grid resources

*International Science Grid This Week is a weekly newsletter that promotes grid computing around the world by sharing stories of grid-empowered science and scientific discoveries. iGSTW also includes an extensive links section.*
*Grid Café (http://gridcafe.web.cern.ch/gridcafe/gridatwork/gridatwork.html)*

### Grid projects

*EGEE, Enabling Grids for E-SciencE (Europe)*
*The White Rose Grid (UK, York)*
*UK Computing for Particle Physics (United Kingdom)*
*UK Astrogrid (United Kingdom)*
*NorduGrid (Denmark, Finland, Norway, Sweden)*
*Grid-enabled Know-how Sharing Technology Based on ARC Services and Open Standards (KnowARC) (Europe)*
*Nordic DataGrid Facility (Denmark, Finland, Norway, Sweden)*
*Grid'5000 (France)*
*RINGrid, Remote Instrumentation in Next-Generation Grids (Europe)*
*HellasGrid (Greece)*
*K*Grid, TIGRIS, Tera-scale Infrastructure for K*GRId Service (Korea)*
*VizGrid (Japan)*
*National Grid of Singapore*
*Grid3(United States)*
*SURAgrid(United States)*
*Open Science Grid(United States)*
*Texas Tech TechGrid(United States)*

### Grid applications and software

*SCOOP Storm Surge Model, detailed description, Lavanya Ramakrishnan, Renaissance Computing Institute*
*RUNTIME Research Team (Efficient runtime systems for grids and parallel architectures)*
*XtreemOS, Enabling Linux for the Grid*
*gridMathematica, Grid-enabled Mathematica*
*K*Grid, KMI, K*GRId Middleware Initiative (Korea)*

### Research organizations

*UK Particle Physics and Astronomy Research Council (PPARC)*
*UK National e-Science Centre Japan Research Organization for Information Science & Technology*
*Taiwan National Center for High Performance Computing (EcoGrid, BioGrid, MedicalGrid, Flood Mitigation Grid)*

*Japan Advanced Industrial Science and Technology, Grid Technology Research Center*

**Corporate and industry**

*The Israeli Association of Grid Technologies (IGT)*

(Note: This list is a growing compilation and includes many interesting things that we collected at SC07. Please send more and we will include!)

# Grid mailing and discussion lists, twikis

The grid community, primarily at the project level, collaborate through many ways including online options such as mailing lists and newer collaborative tools such as twikis. These sources develop and change rapidly (as does the activity they track), so while we offer the following list, we can only hope they remain somewhat current between Cookbook revisions.

*SURAgrid communications, http://www.sura.org/programs/sura_grid_communications.html*
*Open Science Grid email lists and web-based collaboration, https://twiki.grid.iu.edu/twiki/bin/viewauth/DocsComm/WebHome*
*EGEE User Forum, http://www.eu-egee.org/uf2*

# Benchmarks and performance

So just how much faster (or slower) might your scientific results be computed, data be transfered, visualizations be rendered on a grid? This is an important question and the answers cannot be assumed (or even easily estimated.) So various research teams are formally measuring and presenting these results.

We hope to expand this topic in future versions of the cookbook, enabling you to better estimate the performance you will see for your application and determine what measures to take to achieve better performance. In the meantime, we will provide some links to research results here.

*Data Grids and Data Grid Performance Issues*
*On Grid Performance Evaluation using Synthetic Workloads*
*Agenda and Talks: Grid Performance Workshop 2004*
*Gelato, University of Houston, Grid Performance Modeling Project*
*APART-2 workshop on Grid Monitoring and Performance Analysis*
*Report of the International Grid Performance Workshop 2005*

# Full Bibliography

This appendix is simply a compilation of all other bibliographies in the Cookbook. It is in order by cookbook section and, as with the separate bibliographies, includes the URL of each item. While all links are clickable in html and pdf versions, print versions suffer if URL's aren't included.

### Introduction

[1] George E. Brown, Jr. Network for Earthquake Engineering Simulation (http://www.nees.org)
[2] Grid Enabled Remote Instrumentation with Distributed Control and Computation (GRIDCC) (http://www.gridcc.org/)
[3] Laser Interferometer Gravitational-Wave Observatory (LIGO) (http://www.ligo.caltech.edu)
[4] Earth System Grid (http://www.earthsystemgrid.org/)
[5] cancer Biomedical Informatics Grid (caBIG) (http://cabig.cancer.gov/index.asp)
[6] Instrument Middleware Project (http://www.instrumentmiddleware.org/metadot/index.pl)

[7] Grid Café (http://gridcafe.web.cern.ch/gridcafe/gridatwork/gridatwork.html)
[11] Foster, The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, 2002
[12] nmi-edit Glossary (http://www.nmi-edit.org/glossary/index.cfm)
[13] GFD Authorization Glossary (http://www.gridforum.org/documents/GFD.42.pdf)
[14] Internet2 Authentication WebISO (http://middleware.internet2.edu/core/authentication.html)
[17] SURA's NMI Case Study Series (http://www.sura.org/programs/nmi_testbed.html#NMI)
[18] Adiga, Henderson, Jokl, et al. "Building a Campus Grid: Concepts and Technologies" (September 2005) (http://www1.sura.org/3000/SURA-AuthNauthZ.pdf)
[19] Adiga, Barzee, Bolet, et al. "Authentication & Authorization in SURAgrid: Concepts and Technologies", (May 2005) (http://www1.sura.org/3000/BldgCampusGrids.pdf)
[20] NEEScentral website (https://central.nees.org/?action=DisplayFacilities)
[21] caBIG Tools, Infrastructure, Datasets (https://cabig.nci.nih.gov/inventory/)

## History, Standards & Directions

[1] J. C. R. Licklider, "Man-Computer Symbiosis," IRE Trans. on Human Factors in Electronics, v. HFE-1, pp. 4--11, Mar. 1960
[2] http://tools.ietf.org/html/rfc707 (http://tools.ietf.org/html/rfc707)
[3] M. J. Litzkow, "Remote UNIX — Turning Idle Workstations into Cycle Servers," Proc. of USENIX, pp. 381--384, Sum. 1987
[4] M. Litzkow, M. Livny, M. Mutka, "Condor — A Hunter of Idle Workstations," Proc. of 8th Int. Conf. of Dist. Comp. Sys., pp. 104--111, Jun. 1988
[5] V. S. Sunderam, "PVM: A Framework for Parallel Distributed Computing," Concurrency: Prac. and Exp., v. 2(4), pp. 315--339, Dec. 1990
[6] Gigabit Testbed Initiative Final Report, 1996. (http://www.cnri.reston.va.us/gigafr/)
[7] I. Foster, J. Geisler, W. Nickless, W. Smith, S. Tuecke, "Software Infrastructure for the I-WAY High Performance Distributed Computing Experiment," Proc. 5th IEEE Symposium on High Performance Distributed Computing, pp. 562--571, 1997.
[8] The Globus Project (http://www.globus.org/)
[9] The Globus Alliance (http://www.globus.org/alliance/)
[10] I. Foster, C. Kesselman, S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," Lecture Notes in Computer Science, v. 2150, 2001.
[11] I. Foster, C. Kesselman, J. Nick, S. Tuecke "The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration, 2002.
[12] Legion (http://legion.virginia.edu/)
[13] Grimshaw, A. S., Wulf, W. A., "The Legion Vision of a Worldwide Virtual Computer," Comm. of the ACM, v. 40(1), January 1997.
[14] The Mentat project (http://www.cs.virginia.edu/~mentat/)
[15] Sybase Avaki EII (http://www.sybase.com/products/developmentintegration/avakieii)
[16] UNICORE (http://www.unicore.org/)
[17] Grid Engine open source project website (http://www.sun.com/software/gridware/)
[18] UNICORE Plus (http://www.fz-juelich.de/unicoreplus/)
[19] GRIP (http://www.fz-juelich.de/zam/cooperations/grip)
[20] UniGrids (http://www.unigrids.org/)
[21] Enterprise Grid Alliance, "EGA Reference Model and Use Cases v1.5" (http://www.gridalliance.org/en/WorkGroups/ReferenceModel.asp)
[22] Enterprise Grid Alliance,"EGA Grid Security Requirements v1.0" (http://www.gridalliance.org/en/WorkGroups/GridSecurity.asp)
[23] Enterprise Grid Alliance, "Enterprise Data and Storage Provisioning Problem Statement and Approach," (http://www.gridalliance.org/en/WorkGroups/DataandStorageProvisioningRequirements.asp)
[24] Jeffrey Hutzelman, Jospeh Salowey, Joseph Galbraith, and Von Welch, "RFC4462: Generic Security Service Application Program Interface (GSS-API) Authentication and Key Exchange for the Secure Shell

(SSH) Protocol," In RFC4462, Internet Engineering Task Force, 2006.
[25] S. Tuecke, V. Welch, D. Engert, L. Perlman, M. Thompson, "RFC3820: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile," In RFC3820, Internet Engineering Task Force, 2004.
[26] Web Services (http://www.w3.org/2002/ws/)
[27] SOAP version 1.2 (http://www.w3.org/TR/2002/WD-soap12-part0-20020626/)
[28] WSDL (http://www.w3.org/TR/wsdl)
[29] UDDI (http://uddi.org/pubs/uddi_v3.htm#_Toc85907967)
[30] WS-RF Primer, (http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf)
[31] WS-ResourceProperties (WS-RP) (http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-06.pdf)
[32] WS-ResourceLifetime (WS-RL) (http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-03.pdf)
[33] WS-ServiceGroup (WS-SG) (http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ServiceGroup-1.2-draft-02.pdf)
[34] WS-BaseFaults (WS-BF) (http://docs.oasis-open.org/wsrf/wsrf-ws_base_faults-1.2-spec-pr-01.pdf)
[35] WS-Addressing (http://www.w3.org/Submission/ws-addressing/)
[36] WS-BaseNotification, March 2004 (ftp://www6.software.ibm.com/software/developer/library/ws-notification/WS-BaseN.pdf)
[37] WS-BrokeredNotification, March 2004 (ftp://www6.software.ibm.com/software/developer/library/ws-notification/WS-BrokeredN.pdf)
[38] WS-Topics, March 2004 (ftp://www6.software.ibm.com/software/developer/library/ws-notification/WS-Topics.pdf)
[39] Web Services Resource Transfer (WS-RT) (http://devresource.hp.com/drc/specifications/wsrt/WS-ResourceTransfer-v1.pdf)
[40] I. Foster, C. Kesselman, J. M. Nick, S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration" (http://www.globus.org/alliance/publications/papers/ogsa.pdf)
[41] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich, "The Open Grid Services Architecture, Version 1.0" (http://www.gridforum.org/documents/GWD-I-E/GFD-I.030.pdf)
[42] I. Foster, D. Gannon, H. Kishimoto, J. J. Von Reich, "Open Grid Services Architecture Use Cases" (ttp://www.gridforum.org/documents/GWD-I-E/GFD-I.029v2.pdf)
[43] H.Kishimoto, J. Treadwell,"Defining the Grid: A Roadmap for OGSA\texttrademark\ Standards: Version 1.0" (http://www.ogf.org/documents/GFD.53.pdf)
[44] S. Tuecke, K. Czajkoski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, D. Snelling, P. Vanderbilt, "Open Grid Services Infrastructure (OGSI): Version 1.0" (http://www.ogf.org/documents/GFD.15.pdf)
[45] Open Grid Service Infrastructure Primer (http://tinyurl.com/yss7tp)
[46] I. Foster, T. Maguire, D. Snelling, "OGSA WS-RF Basic Profile 1.0" (http://www.ogf.org/documents/GFD.72.pdf)
[47] T. Maguire, D. Snelling, "OGSA Profile Definition Version 1.0" (http://www.ogf.org/documents/GFD.59.pdf)
[48] M. Roehrig, M. Ziegler, "Grid Scheduling Dictionary of Terms and Keywords" (http://www.ogf.org/documents/GFD.11.pdf)
[49] R. Yahyapour, P. Wieder,"Grid Scheduling Use Cases" (http://www.ogf.org/documents/GFD.64.pdf)
[50] F. B. Maciel, "Resource Management in OGSA" (http://www.ogf.org/documents/GFD.45.pdf)
[51] D. Bell, T. Kojo, P. Goldsack, S. Loughran, D. Milojicic, S. Schaefer, J. Tatemura, P. Toft, "Configuration Description, Deployment, and Lifecycle Management (CDDLM) Foundation Document" (http://www.ogf.org/documents/GFD.50.pdf)
[52] P. Goldsack, "Configuration Description, Deployment, and Lifecycle Management: SmartFrog-Based Language Specification" (http://www.ogf.org/documents/GFD.51.pdf)
[53] S. Schaefer, "Configuration Description, Deployment, and Lifecycle Management: Component Model: Version 1.0" (http://www.ogf.org/documents/GFD.65.pdf)

[54] S. Loughran, "Configuration Description, Deployment, and Lifecycle Management: CDDLM Deployment API" (http://www.ogf.org/documents/GFD.69.pdf)
[55] M. Antonioletti, M. Atkinson, A. Krause, S. Laws, S. Malaika, N. W. Paton, D. Pearson, G. Riccardi, "Web Services Data Access and Integration — The Core (WS-DAI) Specification, Version 1.0" (http://www.ogf.org/documents/GFD.74.pdf)
[56] M. Antonioletti, B. Collins, A. Krause, S. Laws, J. Magowan, S. Malaika, N. W. Paton, "Web Services Data Access and Integration â – The Relational Realisation (WS-DAIR) Specification, Version 1.0" (http://www.ogf.org/documents/GFD.76.pdf)
[57] M. Antonioletti, S. Hastings, A. Krause, S. Langella, S. Lynden, S. Laws, S. Malaika, N. W. Paton, "Web Services Data Access and Integration â – The XML Realization (WS-DAIX) Specification, Version 1.0" (http://www.ogf.org/documents/GFD.75.pdf)
[58] W. Allcock, J. Bester, J. Bresnahan, S. Meder, P. Plaszczak, S. Tuecke,"GridFTP: Protocol Extensions to FTP for the Grid" (http://www.ogf.org/documents/GFD.20.pdf)
[59] I. Mandrichenko, W. Allcock, T. Perelmutov,"GridFTP v2 Protocol Description" (http://www.ogf.org/documents/GFD.47.pdf)
[60] R. Siebenlist, V. Welch, S. Tuecke, I. Foster N. Nagaratnam, P. Janson, J. Dayka, A. Nadalin, "OGSA Security Roadmap (Draft)" (http://www.cs.virginia.edu/~humphrey/ogsa-sec-wg/ogsa-sec-roadmap-v13.pdf)
[61] V. Welch, F. Siebenlist, D. Chadwick, S. Meder, L. Pearlman, "OGSA Authorization Requirement" (http://www.ogf.org/documents/GFD.67.pdf)
[62] GSI Working Group (https://forge.gridforum.org/projects/gsi-wg)
[63] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke, "A Security Architecture for Computational Grids," Proc. 5th ACM Conference on Computer and Communications Security Conference, pp. 83--92, 1998.
[64] Grimoires (http://www.ecs.soton.ac.uk/research/projects/grimoires)
[65] ebXML Registry Services Specification v2.5 (http://www.oasis-open.org/committees/regrep/documents/2.5/specs/ebrs-2.5.pdf)
[66] ebXMLsoft Registry and Repository (http://www.ebxmlsoft.com/)
[67] REST (http://en.wikipedia.org/wiki/REST)
[68] JDSL (https://forge.gridforum.org/projects/jsdl-wg/)
[69] JSDL-doc (http://www.gridforum.org/documents/GFD.56.pdf)
[70] DRMAA (http://www.drmaa.org)
[71] SAGA (http://www.ogf.org/gf/group_info/view.php?group=saga-rg)
[72] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludaescher, S. Mock,"Kepler: An Extensible System for Design and Execution of Scientific Workflows," Proc. of 16th Int. Conf. on Sci. and Statistical Database Management (SSDBMÃ 04), pp. 423--424, 2004
[73] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, P. Li, "Taverna: A Tool for the Composition and Enactment of Bioinformatics Workflows," Bioinformatics J., v. 20(17), pp. 3045--3054, 2004
[74] K. Amin, G. vonLaszewski, "GridAnt: A Grid Workflow System,"Argonne National Laboratory, Feb 2003
[75] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, S.Patil, M. Su, K. Vahi, M. Livny, "Pegasus: Mapping Scientific Workflows onto the Grid," Across Grids Conference 2004
[76] DAIS-WG (https://forge.gridforum.org/projects/dais-wg)
[77] OGSA-DAI (http://www.ogsadai.org.uk)
[78] OGSA-DQP (http://www.ogsadai.org.uk/about/ogsa-dqp/)
[79] K. Cline, J. Cohen, D. Davis, D. F. Ferguson, H. Kreger, R. McCollum, B. Murray, I. Robinson, J. Schlimmer, J. Shewchuk, V. Tewari, W. Vambenepe, "Toward Converging Web Service Standards for Resources, Events, and Management" (http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/Harmonization_Roadmap.pdf)
[80] ISO standard 7498-1, 1994 (http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip)
[81] High Performance Fortran standards (http://hpff.rice.edu/versions/)
[82] Globus Toolkit 3 Programmer's Tutorial, Key Concepts: WSRF & GT4 (http://gdp.globus.org/gt3-tutorial/multiplehtml/ch01s05.html)

[83] Globus Toolkit 3 Programmer's Tutorial, Key Concepts: OGSA, WSRF, and GT4 (http://gdp.globus.org/gt4-tutorial/multiplehtml/ch01s01.html)
[84] Web Services Standards as of Q1 2007 (http://www.innoq.com/resources/ws-standards-poster/)

### What Grids Can Do For You

[1] Public Key Infrastructure (http://tinyurl.com/39kx4a)
[2] Community of interest (http://en.wikipedia.org/wiki/Community_of_interest)
[3] Geodise project (http://www.geodise.org/)
[4] Engineering and Physical Sciences Research Council (http://www.epsrc.ac.uk/default.htm)
[5] The Geodise Toolboxes, A User's Guide (http://www.geodise.org/documentation/html/index.htm)
[6] The Geodise Project: Making the Grid Usable Through Matlab (http://www.gridtoday.com/grid/343938.html)
[7] Grid Today (http://www.gridtoday.com/gridtoday.html)
[8] SURAgrid (http://www.sura.org/programs/sura_grid.html)
[9] Amazon Web Services (http://tinyurl.com/2sbgmv)
[10] [Amazon's] Solutions catalog (http://solutions.amazonwebservices.com/connect/index.jspa)
[11] [Amazon's] Elastic Compute Cloud (http://www.amazon.com/gp/browse.html?node=201590011)
[12] Infoworld (http://www.infoworld.com/)
[13] Amazon.com's rent-a-grid (http://www.infoworld.com/article/06/08/30/36OPstrategic_1.html)
[14] 3Tera (http://www.3tera.com/index.html)
[15] AppLogic grid system (http://www.infoworld.com/4449)
[16] International Virtual Data Grid Laboratory (http://www.ivdgl.org/)
[17] Compact Muon Solenoid (CMS) (http://cms.cern.ch/)
[18] Large Hadron Collider (LHS) (http://public.web.cern.ch/Public/Content/Chapters/AboutCERN/CERNFuture/WhatLHC/WhatLHC-en.html)
[19] CERN (http://public.web.cern.ch/Public/Welcome.html)
[20] U. S. CMS (http://www.uscms.org/)
[21] Fermi National Accelerator Laboratory (http://www.fnal.gov/)
[22] U. S. CMS Overview (http://www.uscms.org/Public/overview.html)
[23] A Toroidal LHC ApparatuS (ATLAS) (http://atlas.web.cern.ch/Atlas/index.html)
[24] U. S. ATLAS (http://www.usatlas.bnl.gov/)
[25] Brookhaven National Laboratory (BNL) (http://www.bnl.gov/world/)
[26] Sloan Digital Sky Survey (SDSS) (http://www.sdss.org/)
[27] SkyServer (http://cas.sdss.org/dr5/en/)
[28] SDSS Databases (http://cas.sdss.org/dr5/en/sdss/data/data.asp#databases)
[29] SDSS Data Release 5 (http://cas.sdss.org/dr5/en/sdss/release/)
[30] DataTAG (http://datatag.web.cern.ch/datatag/)
[31] TeV in layman's terms (http://public.web.cern.ch/Public/Content/Chapters/AboutCERN/CERNFuture/WhatLHC/WhatLHC-en.html)
[32] EU-DataGrid Project (http://web.datagrid.cnr.it/servlet/page?_pageid=1407&_dad=portal30&_schema=PORTAL30&_mode=3)
[33] Enabling Grids for E-sciencE (EGEE) (http://www.eu-egee.org/)
[34] DataGrid Project Description (http://web.datagrid.cnr.it/servlet/page?_pageid=873,879&_dad=portal30&_schema=PORTAL30&_mode=3)
[35] OGSA-DAI (http://www.ogsadai.org.uk/index.php)
[36] LEAD (http://www.lead.ou.edu/)
[37] caGrid (http://cabig.nci.nih.gov/)
[38] AstroGrid (http://www.astrogrid.org/)
[39] BRIDGES (http://www.brc.dcs.gla.ac.uk/projects/bridges/)
[40] eDiaMoND (http://www.ediamond.ox.ac.uk/)
[41] GeneGrid (http://www.qub.ac.uk/escience/projects/genegrid)
[42] more OGSA-DAI grid projects (http://www.ogsadai.org.uk/about/projects.php)

[43] Computational Chemistry Grid (https://www.gridchem.org)
[44] cancer Biomedical Informatics Grid (https://cabig.nci.nih.gov)
[45] caBIG (http:cabig.cancer.gov)
[46] SETI@home (http://setiathome.berkeley.edu/)
[47] BOINC (http://boinc.berkeley.edu/)
[48] World Community Grid (http://www.worldcommunitygrid.org/)
[49] Condor (http://www.cs.wisc.edu/condor/)
[50] United Devices (http://www.ud.com/)
[51] Grid-MP ™ (http://www.ud.com/products/gridmp.php)
[52] Enlightened Computing (http://enlightenedcomputing.org)
[53] Optiputer (http://www.optiputer.net)
[54] CANARIE*4 (http://www.canarie.ca/advnet)
[55] CANARIE*4 customer-empowered networks (http://www.canarie.ca/advnet/cen.html)
[56] Foster, Ian, "The Grid: Computing without Bounds", Scientific American, April 2003.
[57] Teragrid (http://www.teragrid.org)

## Grid Case Studies

[1] I. Foster, C. Kesselman and S. Tuecke, “The Anatomy of the Grid: Enabling Scalable Virtual Organizations,” International Journal of Supercomputer Applications, 15(3), 2001.
[2] I. Foster and C. Kesselman, “Globus: A Metacomputing Infrastructure Toolkit,” International Journal of Supercomputer Applications, 11(2):115-128, 1997.
[3] J. Novotny, S. Tuecke and V. Welch, “An Online Credential Repository for the Grid: MyProxy,” Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), August 2001.
[4] Open Grid Computing Environment. (http://www.collab-ogce.org/nmi/index.jsp)
[5] W. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal and S. Tuecke, “Data Management and Transfer in High Performance Computational Grid Environments,” Parallel Computing, 28 (5), pp. 749-771, May 2002.
[6] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith and S. Tuecke, “A Resource Management Architecture for Metacomputing Systems,” Workshop on Job Scheduling Strategies for Parallel Processing, pg. 62-82, 1998.
[7] I. Foster, C. Kesselman, G. Tsudik and S. Tuecke, “A Security Architecture for Computational Grids,” Fifth ACM Conference on Computer and Communications Security, pp. 83-92, 1998.
[8] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, S. Tuecke. “A Resource Management Architecture for Metacomputing Systems.” Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, pg. 62-82, 1998.
[9] R.A. Luettich, J. J. Westerink, and N. W. Scheffner, ADCIRC: An advanced three-dimensional circulation model for shelves, coasts and estuaries; Report 1: theory and methodology of ADCIRC- 2DDI and ADCIRC-3DL, Technical Report DRP-92-6, Coastal Engineering Research Center, U.S. Army Engineer Waterways Experiment Station, Vicksburg, MS, 1992.
[10] Unidata Local Data Manager, 2006. (http://www.unidata.ucar.edu/software/ldm/)
[11] P. Bogden, G. Allen, G. Stone, J. Bintz, H. Graber, S. Graves, R. Luettich, D. Reed, P. Sheng, H. Wang,W. Zhao, The Southeastern University Research Association Coastal Ocean Observing and Prediction Program: Integrating Marine Science and Information Technology," Proceedings of the OCEANS 2005 MTS/IEEE Conference. Sept 18-23, 2005.
[12] D. Huang, G. Allen, C. Dekate, H. Kaiser, Z. Lei and J. MacLaren "getdata: A Grid Enabled Data Client for Coastal Modeling," HPC2006.
[13] P. Bogden, "The SURA Coastal Ocean Observing and Prediction Program (SCOOP) Service-Oriented Architecture," Proceedings of MTS/IEEE 06 Conference in Boston, Session 3.4 on Ocean Observing Systems, September 18-21, 2006.
[14] J. Bintz et al. "SCOOP: Enabling a Network of Ocean Observations for Mitigating Coastal Hazards," Proceedings of the Coastal Society 20th International Conference, 2006.

[15] SCOOP Website, 2006. (http://scoop.sura.org/)
[15a] SCOOP Partners (http://scoop.sura.org/partners.html)
[16] North Carolina Forecasting System. (http://www.renci.org/projects/indexdr.php)
[17] S. Graves, K. Keiser, H. Conver, M. Smith. "Enabling Coastal Research and Management with Advanced Information Technology," 17th Federation Assembly Virtual Poster Session, July 2006.
[18] G. von Laszewski, I. Foster, J. Gawor, and P. Lane, "A Java Commodity Grid Kit," Concurrency and Computation: Practice and Experience, vol. 13, no. 8-9, pp. 643-662, 2001. (http:/www.cogkit.org/)
[19] K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, "Grid Information Services for Distributed Resource Sharing." Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.
[20] R. Wolski, N. Spring, C. Peterson, "Implementing a Performance Forecasting System for Metacomputing: The Network Weather Service," in Proceedings of SC97, November, 1997.
[21] OSG Council (http://www.opensciencegrid.org/About/Who_is_the_Open_Science_Grid%3F/OSG_Council_Members)
[22] OSG Virtual Organizations (http://www.opensciencegrid.org/About/OSG_Organization/Virtual_Organizations)
[23] OSG Technical Activity Groups (http://www.opensciencegrid.org/About/OSG_Organization/Technical_Activities)
[24] MonALISA Graph of OSG Activity (http://monalisa.grid.iu.edu:8080/show?page=index.html)
[25] US CMS Institutions and Members (http://uscms.fnal.gov/uscms/organization/uscms_institutes_t_members.html)
[26] U.S. CMS website (http://www.uscms.org/Public/overview.html)
[27] USCMS Software and Computing (http://www.uscms.org/SoftwareComputing/index.html)
[28] CERN Archtectural Blueprint RTAG (http://lcgapp.cern.ch/project/blueprint/BlueprintReport-final.doc)
[29] Feature: Meeting the Data Transfer Challenge, ISGTW, Jan 17, 2007 (http://www.isgtw.org/?pid=1000226)
[30] 2007 Open Science Grid Consortium Meeting, UCSD, San Diego, CA, March 5-8, 2007, Frank Wurthwein, OSG Application Coordinator, OSG Extension Lead, Experimental Elementary Particle Physics, UCSD
[31] US CMS Organization, Institution, and Member Contacts (http://www.uscms.org/Public/contact.html)
[32] SDSS Institutions (http://www.sdss.org/members/index.html)
[33] SDSS Advisory Council (http://www.sdss.org/directorate/adco.html)
[34] SDSS Website (http://www.sdss.org/)
[35] SDSS — About US (http://www.sdss.org/background/)
[36] SDSS — Contact US (http://www.sdss.org/contacts.html)
[37] How ATLAS Collaborates (http://atlasexperiment.org/hac.html)
[38] Simulating Supersymmetry with ATLAS (http://tinyurl.com/2q79p9)
[39] ATLAS Experiment Home Page (http://atlasexperiment.org/)
[40] Proth (http://primes.utm.edu/programs/gallot/)
[41] Partial Differential Equation (http://www.math.ttu.edu/~smanserv/)
[42] Title: Multivariate Minimization Using Grid Computing by K. Kulish, J. Perez, P. Smith. (http://www.cs.vu.nl/ggf/apps-rg/meetings/ggf8/kulish.pdf)
[43] PhD Thesis by Dr. Eric Albers (http://www.iemss.org/iemss2002/proceedings/pdf/volume%20uno/298_albers.pdf)
[44] SRB (Storage Resource Broker) data grid (http://www.sdsc.edu/srb/index.php/Main_Page)
[45] 3-D Studio Max graphics rendering grid (http://www.arch.ttu.edu/resources/FAQ/3D/net_render_max_animation.asp)
[46] BLAST (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/information3.html)
[47] Query tutorial (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/query_tutorial.html)
[48] BLAST tutorial (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/tut1.html)
[49] BLAST Guide (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/guide.html)
[50] PSI-BLASTtutorial (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/psi1.html)
[51] More Information on BLAST (http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/auxiliary.html)

[52] SAS-based compute grid (http://www.sas.com/technologies/architecture/grid/index.html)
[53] "Neighbors" space simulation (http://dspace.lib.ttu.edu/bitstream/2346/1219/1/thesis.pdf)
[54] Bioinformatics Project (http://www.animalgenome.org/pigs/)
[55] "R" programming language/framework (http://www.r-project.org/)
[56] Texas Tech TechGrid (http://www.hpcc.ttu.edu/techgrid.html)
[57] Sun Microsystems (http://www.sun.com/)
[58] Streamline Computing (http://www.streamline-computing.com/)
[59] White Rose Grid Compute Node (http://www.wrgrid.org.uk/ComputeNodes.html)
[60] White Rose Grid Activities (http://www.wrgrid.org.uk/Activities.html)
[61] White Rose Grid Contact Details (http://www.wrgrid.org.uk/Contactus.html)
[62] M.L. Green and R. Miller, Grid computing in Buffalo, New York, Annals of the European Academy of Sciences, 2003, pp. 191-218.
[63] M.L. Green and R. Miller, Molecular structure determination on a computational & data grid, Parallel Computing Journal 30 (2004), pp. 1001-1017.
[64] M.L. Green and R. Miller, Evolutionary molecular structure determination using grid-enabled data mining, Parallel Computing Journal 30 (2004), pp. 1057-1071.
[65] M.L. Green and R. Miller, A client-server prototype for grid-enabling application template design, Parallel Processing Letters, Vol. 14, No. 2 (2004), pp. 241-253.
[66] C.L. Ruby, M.L. Green, and R. Miller, The Operations Dashboard: A Collaborative Environment for Monitoring Virtual Organization-Specific Compute Element Operational Status, Parallel Processing Letters, Vol. 16, No. 4 (2006), pp. 485-500.
[67] C.L. Ruby and R. Miller, Effectively Managing Data on a Grid, Handbook of Parallel Computing: Models, Algorithms, and Applications, S. Rajasekaran and J. Reif, eds., CRC Press, 2007, in press.
[68] What is Condor? (http://www.cs.wisc.edu/condor/description.html)

## Current Technology for Grids

[1] Globus Toolkit (http://www.globus.org)
[2] JSR168 specification (http://jcp.org/aboutJava/communityprocess/final/jsr168/index.html)
[3] JSR 168: Portlet Specification v1.0, Major Portal Components (http://tinyurl.com/324qrg)
[4] International Grid Trust Federation (http://www.igtf.org)
[5] HEBCA (http://www.educause.edu/HigherEducationBridgeCertificationAuthority/623)
[6] FEBCA (http://www.cio.gov/fbca/)
[7] USHER (http://www.usherca.org/)
[8] Overview of Globus security components (http://www.globus.org/grid_software/security/)
[9] Web Services Authentication and Authorization (http://www.globus.org/grid_software/security/ws-aa.php)
[10] GT 4.0: Security: Pre-Web Services Authentication and Authorization (http://www.globus.org/toolkit/docs/4.0/security/prewsaa)
[11] SimpleCA (http://www.vpnc.org/SimpleCA)
[12] NCSA MyProxy Credential Management Service (http://grid.ncsa.uiuc.edu/myproxy/)
[13] GT 4.0: Credential Management: MyProxy (http://www.globus.org/toolkit/docs/4.0/security/myproxy/)
[14] Grid Account Management Architecture (GAMA) (http://grid-devel.sdsc.edu/gridsphere/gridsphere?cid=gama)
[15] Load Sharing Facility (http://www.platform.com/Products/Platform)
[16] Load Leveler (http://www-306.ibm.com/software/tivoli/products/scheduler-loadleveler)
[17] SUN Grid Engine (http://www.sun.com/software/gridware)
[18] Altair Engineering, Inc. (http://www.altair.com/software/pbspro.htm)
[19] Condor Project (http://www.cs.wisc.edu/condor)
[20] NSF Middleware Initiative Grids Center software distribution (http://www.grids-center.org/)
[21] Monitoring and Discovery System (http://www.globus.org/toolkit/mds)
[22] Ganglia (http://ganglia.sourceforge.net/)
[23] Nagios (http://www.nagios.org)

[24] Inca (http://inca.sdsc.edu)
[25] MonALISA (http://monalisa.cacr.caltech.edu/monalisa.htm)
[27] Globus Incubator (http://dev.globus.org/wiki/Incubator/Incubator_Management)
[28] Condor-G (http://www.cs.wisc.edu/condor/condorg/)
[29] HPC Synergy (http://www.ud.com/products/hpcsynergy.phpa)
[30] Cluster Resources (http://www.clusterresources.com/pages/products/moab-grid-suite.php%20)
[31] gLite (http://glite.web.cern.ch/glite/wms/)
[32] MARS (http://www-personal.engin.umich.edu/%7Eabose/website/marshome.htm)
[33] Gratia twiki page (https://twiki.grid.iu.edu/twiki/bin/view/Accounting/WebHome)
[34] SweGrid Accounting System (http://www.sgas.se/)
[35] Parallel Virtual File System (http://www.pvfs.org/index.html)
[36] Gfarm (http://datafarm.apgrid.org)
[37] Cluster File System, Inc (http://www.clusterfs.com)
[38] Condor Directed Acyclic Graph Manager (http://www.cs.wisc.edu/condor/dagman/)
[40] Globus Community Scheduler Framework (http://www.globus.org/grid_software/computation/csf.php)
[41] Pegasus (http://pegasus.isi.edu)
[42] VOMS: Virtual Organization Membership Service (http://www.globus.org/grid_software/security/voms.php)
[43] Avaki EII (http://www.sybase.com:80/products/allproductsa-z/avakieii)
[44] Globus Data Management: Key Concepts (http://www.globus.org/toolkit/docs/4.0/data/key/)
[45] Globus PURSE: Portal-based User Registration Service (http://www.globus.org/grid_software/security/purse.php)

### Programming Concepts & Challenges

[1] Globus home page (http://www.globus.org/)
[2] Condor GT 4.0 Pre WS GRAM (http://www.globus.org/toolkit/docs/4.0/execution/prewsgram/)
[3] Unicore home page (http://www.unicore.org/)
[4] SOAP (Simple Object Access Protocol) (http://www.w3.org/TR/soap/)
[5] SDL (Specification and Description Language) (http://www.sdl-forum.org/)
[6] GT 4.0 GridFTP (http://www.globus.org/toolkit/docs/4.0/data/gridftp/)
[7] GRAM (GT 4.0 Pre WS GRAM) (http://www.globus.org/toolkit/docs/4.0/execution/prewsgram/)
[8] W3 (World Wide Web Consortium) (http://www.w3.org/)
[9] WSDL (Web Services Description Language) (http://www.w3.org/TR/wsdl)
[10] WSRF (Web Services Resource Framework) download (http://www.oasis-open.org/committees/download.php/16654/wsrf-cs-01.zip)
[11] SOAP (Simple Object Access Protocol) (http://www.w3.org/TR/soap/)
[12] Condor home page (http://www.cs.wisc.edu/condor/)
[13] Unicode home page (http://www.unicore.org/)
[14] abstraction layer (Wikipedia definition) (http://en.wikipedia.org/wiki/Abstraction_layer)
[15] GAT (Grid Application Toolkit and Testbed) (http://www.gridlab.org/wp-1)
[16] SAGA (Simple API for Grid Apps) (https://forge.gridforum.org/projects/saga-rg/)
[17] OGF (Open Grid Forum) (http://www.ogf.org)
[19] Gridlab (http://www.gridlab.org)
[20] SAGA implementation home page (http://fortytwo.cct.lsu.edu:8000/SAGA)
[21] SAGA C++ reference implementation (http://www.cct.lsu.edu/projects/Grid+Application+Toolkit)
[22] Monitoring and Discovery System (http://www.globus.org/toolkit/mds/)
[23] Grid Laboratory Uniform Environment (http://forge.gridforum.org/sf/projects/glue-wg)
[24] DataTAG (http://datatag.web.cern.ch/datatag/")
[25] GridForgea (http://forge.gridforum.org/sf/sfmain/do/home)
[26] A Globus Primer (http://www.globus.org/toolkit/docs/4.0/key/GT4_Primer_0.6.pdf)
[27] MDS web pages (http://www.globus.org/mds)
[28] Globus Monitoring and Discover (2005 Globus World)

(http://www.globus.org/toolkit/presentations/GlobusWorld_2005_Session_9c.pdf)
[29] GridLab (http://www.gridlab.org/)
[30] iGrid (http://sara.unile.it/%7Ecafaro/software.html)
[31] GridShib website (http://gridshib.globus.org/about.html)
[32] Shibboleth (http://shibboleth.internet2.edu/)
[33] Internet2 (http://www.internet2.edu/)
[34] GridShib Technical Overview (http://grid.ncsa.uiuc.edu/presentations/gridshib-tech-overview-apr06.ppt)
[35] OpenPBS (http://www-unix.mcs.anl.gov/openpbs/)
[36] LSF (http://www.platform.com/Products/Platform.LSF.Family/)
[37] LoadLeveler (http://www-128.ibm.com/developerworks/eserver/library/es-loadlevel/index.html)
[38] Maui (http://www.clusterresources.com/pages/products/maui-cluster-scheduler.php)
[39] Moab (http://www.clusterresources.com/pages/products/moab-cluster-suite.php)
[40] Globus Resource Allocation Manager (http://tinyurl.com/2shtao)
[41] Torque (http://www.clusterresources.com/pages/products/torque-resource-manager.php)
[42] e-Compute (http://www.altair.com/software/ecompute.htm)
[43] Nordugrid (http://www.nordugrid.org/)
[44] Condor User Tutorial,^KUK Condor Week, ^KNeSC,^KOctober, 2004
(http://www.nesc.ac.uk/talks/438/11th/user_tutorial.ppt)
[45] Condor manual (http://www.cs.wisc.edu/condor/manual/v6.4/)
[46] AIST Grid Scheduling System
(http://www.aist.go.jp/aist_e/aist_today/2006_20/hot_line/hot_line_21.html)
[47] NAREGI GridVM (http://tinyurl.com/24nenc)
[48] Keahey's Virtual Workspace (http://workspace.globus.org/papers/)
[49] Globus Toolkit GRAM (http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=4045)
[50] PBSPro (http://www.altair.com/software/pbspro.htm)
[51] GridFTP (http://www.globus.org/toolkit/docs/4.0/data/gridftp/)
[52] Reliable File Transfer (http://www.globus.org/toolkit/docs/4.0/data/rft/)
[53] GT 4.0 RFT Command Reference
(http://www.globus.org/toolkit/docs/4.0/data/rft/RFT_Commandline_Frag.html)
[54] GT 4.0 RLS (http://www.globus.org/toolkit/docs/4.0/data/rls/)
[55] Network Storage Technology (http://www.cs.wisc.edu/condor/nest/)
[56] Flexibility, Manageability, and Performance in a Grid Storage Appliance
(http://www.cs.wisc.edu/condor/nest/papers/nest-hpdc-02.pdf)
[57] SRM/DRM (https://twiki.grid.iu.edu/twiki/bin/view/Integration/SrmDrm)
[58] srmcp (https://twiki.grid.iu.edu/twiki/bin/view/Documentation/StorageSrmcpUsing)
[59] Gratia twiki page (https://twiki.grid.iu.edu/twiki/bin/view/Accounting/WebHome)
[60] Full Project Definition
(https://twiki.grid.iu.edu/twiki/bin/viewfile/Accounting/WebHome?filename=AccountingProjectDefinition1.doc)
[61] SweGrid Accounting System (http://www.sgas.se/)
[62] SGAS Download (http://www-unix.globus.org/toolkit/docs/4.0/techpreview/sgas/)
[63] SGAS Installation and Administration Guide (http://www.sgas.se/docs/SGASInstallConfig.pdf)
[64] SGAS Administration Guide (http://www.sgas.se/docs/SGASAdmin.pdf)
[65] OSG Gratia Project (https://twiki.grid.iu.edu/twiki/bin/view/Accounting/WebHome)
[66] Directed Acyclic Graph Manager (DAGman) (http://www.cs.wisc.edu/condor/dagman/)
[67] Condor manual (http://www.cs.wisc.edu/condor/manual/v6.4/)
[68] Swift (http://www.ci.uchicago.edu/swift/index.php)
[69] GriPhyN Virtual Data System (http://www.griphyn.org/news/index.html)
[70] A Swift Tutorial (http://www.ci.uchicago.edu/swift/guides/tutorial.php)
[71] The SwiftScript User Guide
(http://www.ci.uchicago.edu/swift/guides/userguide.php#engineconfiguration)
[72] Swiftscript Language Reference Manual (http://www.ci.uchicago.edu/swift/guides/languagespec.php)
[73] Planning for Execution in Grids (http://pegasus.isi.edu/)
[74] GriPhyN Virtual Data System Quick Guide (http://pegasus.isi.edu/docs/QuickGuide.pdf)

[75] Security Assertion Markup Language (SAML) (http://xml.coverpages.org/saml.html)
[76] Open Grid Forum Security groups (http://www.ogf.org/gf/group_info/areasgroups.php?area_id=7)
[77] International Grid Trust Federation (IGTF) — Grid;s Policy Management Authority (http://www.gridpma.org/)

## Joining a Grid: Procedures & Examples

[1] SURAgrid Web site (http://www.sura.org/suragrid)
[2] SURAgrid-IBM partnership (http://www.sura.org/news/docs/IBMSURAgrid.doc)
[3] SURAgrid User Management and PKI Bridge Certification Authority (https://www.pki.virginia.edu/nmi-bridge)
[4] TIGRE (http://tigreportal.hipcat.net)
[5] Virtual Data Toolkit (http://vdt.cs.wisc.edu/)
[6] SURAgrid Server Stack website (http://omnius.hpcc.ttu.edu/SURAgrid_wiki/ServerStack)
[7] Globus Toolkit 4.0 (http://www.globus.org/toolkit/docs/4.0/)
[8] GSI OpenSSH (http://grid.ncsa.uiuc.edu/ssh/)
[9] UberFTP (http://dims.ncsa.uiuc.edu/set/uberftp/)
[10] MyProxy (http://grid.ncsa.uiuc.edu/myproxy/)
[11] Condor-G (http://www.cs.wisc.edu/condor/condorg/)
[12] VDT supported operating systems (http://vdt.cs.wisc.edu/releases/1.6.1/requirements.html)
[13] pacman (http://www.archlinux.org/pacman/)
[14] SURAgrid PKI Bridge Certification Authority and User Management System (https://www.pki.virginia.edu/sura-bridge/scl/)
[15] grid-mapfile section (https://www.pki.virginia.edu/nmi-bridge/scl/#gridmapfile)
[16] International Grid Trust Federation (http://gridpma.org)
[17] GSI version of SSH (http://grid.ncsa.uiuc.edu/ssh/)
[18] Step 6: Install the GSI-OpenSSH Server (http://grid.ncsa.uiuc.edu/ssh/install.html#install_server)
[19] SURAgrid Support e-mail list (mailto:suragrid-support@sura.org)
[20] VDT Support page (http://vdt.cs.wisc.edu/support.html)
[21] OSG website (http://www.opensciencegrid.org)
[22] Members of the OSG Consortium (http://www.opensciencegrid.org/About/Who_is_the_Open_Science_Grid%3F/Consortium_Members)
[23] OSG partners (http://www.opensciencegrid.org/About/Who_is_the_Open_Science_Grid%3F/Partners)
[24] OSG Researcher fields (http://www.opensciencegrid.org/Science_on_the_OSG/Research_Highlights)
[25] Grid3 (http://www.ivdgl.org/grid2003/)
[26] Virtual Data Toolkit (VDT) (http://vdt.cs.wisc.edu//index.html)
[27] NSF Middleware Institute (NMI) (http://www.nsf-middleware.org/default.aspx)
[28] OSG@Work (http://twiki.grid.iu.edu/twiki/bin/view)
[29] OSG Education and Training (http://twiki.grid.iu.edu/twiki/bin/view/Education/WebHome)
[30] OSG Workshops (http://twiki.grid.iu.edu/twiki/bin/view/Education/GridWorkshops)
[31] OSG Research Highlights (http://www.opensciencegrid.org/Science_on_the_OSG/Research_Highlights)
[32] SRM collaboration working group (http://sdm.lbl.gov/srm-wg)
[33https://twiki.grid.iu.edu/twiki/bin/view/Storage] Storage Group (OSG)
[34] BeStMan (http://datagrid.lbl.gov/bestman)
[35] dCache (http://www.dcache.org)

## Typical Usage Examples

[1] Alfred E Neuman (http://www.answers.com/topic/alfred-e-neuman)
[2] About SURAGrid (http://www.sura.org/programs/sura_grid.html)
[3] SCOOP institutions (http://violet.itsc.uah.edu:8080/gridsphere/gridsphere?cid=partners)
[4] Office of Naval Research/ (http://www.onr.navy.mil/)
[5] NOAA's Coastal Services Center (http://www.csc.noaa.gov/)

[6] U.S. Ocean Action Plan/ (http://ocean.ceq.gov/actionplan.pdf)
[7] Global Earth Observation System of Systems (http://www.epa.gov/geoss/)
[8] Integrated Earth Observing System (http://www.noaa.gov/lautenbacher/oceanology.htm)
[9] SURA Coastal Ocean Observing and Prediction (SCOOP) Program< (http://scoop.sura.org/)

### Related Topics

[1] OSI Model — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/OSI_model)
[2] Transmission Control Protocol — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Transmission_Control_Protocol)
[3] User Datagram Protocol — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/User_Datagram_Protocol)
[4] IPv4 — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/IPv4)
[5] The Security Flag in the IPv4 Header (http://www.ietf.org/rfc/rfc3514.txt)
[6] RFC 791 (http://tools.ietf.org/html/rfc791)
[7] Ethernet — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Ethernet)
[8] Data_link_layer — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Data_link_layer)
[9] Jumbo Frames — Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Jumbo_Frames)
[10] Gigabit Ethernet Jumbo Frames (http://sd.wareonearth.com/~phil/jumbo.html)
[11] ethtool (http://sourceforge.net/projects/gkernel)
[12] Enabling High Performance Data Transfers (http://www.psc.edu/networking/projects/tcptune/)
[13] TCP Tuning Guide (http://dsd.lbl.gov/TCP-tuning/TCP-tuning.html)
[18] Taxonomy of Network and Service Monitoring Approaches (http://www.rnoc.gatech.edu/cpr/taxonomy.html)
[22] ping, From Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Ping)
[23] traceroute, From Wikipedia, the free encyclopedia (http://en.wikipedia.org/wiki/Traceroute)
[24] Iperf — The TCP/UDP Bandwidth Measurement Tool (http://dast.nlanr.net/Projects/Iperf/)
[25] One-Way-Ping (OWAMP) (http://e2epi.internet2.edu/owamp/)
[26] Bandwidth Test Controller (BWCTL) (http://e2epi.internet2.edu/bwctl/)
[27] Network Diagnostic Tool (NDT) (http://e2epi.internet2.edu/ndt/)
[28] Tobi Oetiker's MRTG — The Multi Router Traffic Grapher (http://oss.oetiker.ch/mrtg/)
[29] HP OpenView Network Node Manager Smart Plug-in for IP Multicast (http://www.openview.hp.com/products/mcast/)
[30] Ganglia Monitoring System (http://ganglia.sourceforge.net/)
[31] Monalisa — Monitoring the Grid since 2001 (http://monalisa.cacr.caltech.edu/monalisa.htm)
[32] PERFormance Service Oriented Network monitoring ARchitecture (http://wiki.perfsonar.net/jra1-wiki/index.php/PerfSONAR_About)
[33] ESnet PerfSONAR Traceroute Visualizer (https://performance.es.net/cgi-bin/level0/perfsonar-trace.cgi)
[34] Cacti, the Complete RRDTool-based Graphing Solution (http://cacti.net/)
[40] Texas Tech TechGrid (http://www.hpcc.ttu.edu/techgrid.html)

# Use of This Material

Source generation complete.