Experiences from Simulating the Global Carbon Cycle in a Grid Computing Environment

Jason Cope*, Craig Hartsough†, Sean McCreary*, Peter Thornton†, Henry M. Tufo*†, Nathan Wilhelmi† and Matthew Woitaszek*

* University of Colorado, Boulder
† National Center for Atmospheric Research
jason.cope@colorado.edu

Abstract. We discuss our software development experiences with Grid-BGC, a grid-enabled terrestrial carbon cycle modeling environment. Grid-BGC leverages grid computing technologies to create a secure, reliable and easy to use distributed computational environment for climate modeling. The goal is to develop a system which insulates the scientists from tedious configuration details thereby increasing scientific productivity. This project is part of a collaborative effort between the University of Colorado and the National Center for Atmospheric Research to create a general grid-enabled computational framework for climate modeling. Over the course of this project we gained valuable experience deploying grid technology and learned how to create a production quality grid system. We provide an overview of our current system, describe our most salient experiences, and present a proposed production architecture for Grid-BGC.

Introduction

Grid-BGC is a grid-enabled global carbon cycle modeling system and computational framework. Using grid computing technologies, such as the Globus Toolkit [5], researchers and software engineers at the National Center for Atmospheric Research (NCAR) and the University of Colorado at Boulder (CU) implemented a prototype of a grid-enabled system which models the global carbon cycle using computational and storage resources distributed between both institutions. The ultimate goal of this system is to give scientists access to the climate models and data necessary to model the carbon cycle and minimize the complexity of running the models in a distributed computational environment.

In creating this system we gained many experiences on how to develop a stable and usable grid computing system. While NCAR actively participates in archiving climate related data on the grid (e.g., the Earth System Grid) the Grid-BGC project is the first grid-enabled modeling environment developed and deployed by NCAR and provided as a grid service to the climate modeling community. Throughout this project, we gained valuable experience constructing a distributed climate modeling environment using grid technologies. These experiences ranged from mechanical practices, such as using the Globus toolkit, to engineering practices, such as defining a sound security model.

This paper discusses the relevant experiences and lessons learned during the development of the system prototype. We provide a detailed description of the prototype along with a description of how our experiences and lessons learned influenced the final production system's design. We

start with an overview or related work, then provide a brief introduction to carbon cycle modeling, followed by an overview of the prototype architecture and implementation of the Grid-BGC system. The following section discusses our experiences and lessons learned during development and operation of the prototype. Finally, we present our proposed production architecture, future work, and conclusions.

Related Work

As the grid computing paradigm matures, more organizations are utilizing the grid computing software infrastructure to help support their scientific and computational needs. In [2], we presented an overview of similar projects (e.g., GEMCLA [6], DIRAC [14], and NorduGrid [4]). We concluded that the Grid-BGC project differed from these projects because our system provides users with a simple interface to request execution of a limited set of climate models on their behalf and guarantees that all aspects of the requested computation are accomplished with complete transparency to the scientist. In contrast, these other solutions are intended for general workflow processing and provide a generic language and parser to define and manipulate workflows, at the cost of introducing substantial complexity to the system design. Two recent external projects similar to Grid-BGC are GridChem and CRAFT.

GridChem [10] is a collaborative project between the NCSA, OSC, and TACC to interface chemists' desktop computers with a grid-enabled environment. The researchers found that the grid computing model and software is difficult for users to adjust to and addressed this in their systems design. GridChem users interact with a Java client, locally installed on users' desktops. The client communicates with a middleware server to authorize and authenticate users as well as manage their workflow tasks. The middleware server implements a customized data management scheme and utilizes Condor [8] for task scheduling. Grid-BGC differs from GridChem in several ways. Grid-BGC utilizes a centralized web-client interface and does not require the installation of a client on the users' desktops. Grid-BGC also focuses on the development of a reliable and automated computational fabric.

Project CRAFT [3] is another collaborative project that utilizes grid computing middleware to link remote resources. The ultimate goal of CRAFT is to link data collected from remote sensing instruments into event-driven meteorological models in real-time. CRAFT utilizes grid middleware to create a virtual machine room for computing the models as separate computational centers. Like Grid-BGC, CRAFT utilizes grid computing resources to offload load the execution of meteorological models.

Terrestrial Ecosystem Modeling

Modeling the carbon cycle is accomplished through a multistage workflow composed of two climate models; Daymet and Biome-BGC (see Figure 1). This workflow transforms observed meteorological and ecosystem data, gathered from a finite number of observation stations over the past 50 years, into a high resolution grid of data. With the gridded-weather data, the workflow simulates the carbon cycle on each point in the gridded data set and produces another grid of the simulated carbon cycle data. The generated carbon cycle data can be analyzed through text analysis or visualization tools.

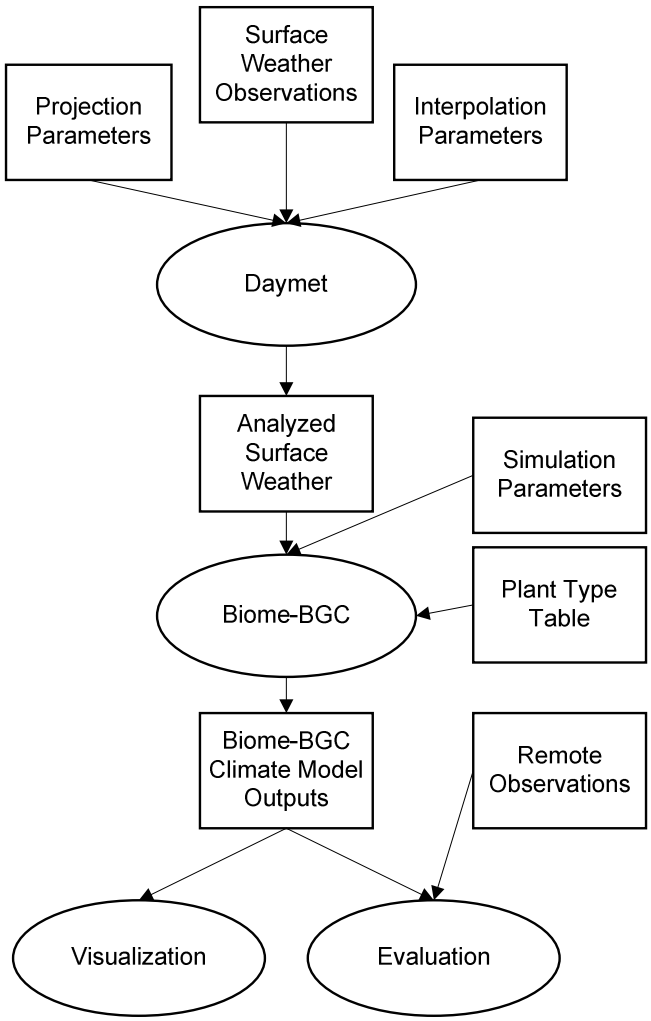

Figure 1, Grid-BGC Workflow

The first transformation performed by the workflow is accomplished through the use of the Daymet climate model [12]. Daymet interpolates historical weather data to produce a high resolution spatial grid of ground-based weather observations. These grids are subdivided into sections referred to as tiles. After the creation of the tiles, the carbon cycle for each tile is modeled using Biome-BGC [13]. This model ingests the tile of climate data along with known soil and plant data and other simulation parameters to simulate the carbon cycle for this tile over a period of time. Post processing of the data is performed to glean data relevant for a particular scientists needs from the output from Biome-BGC.

The point and tile based nature of Daymet and Biome-BGC accommodates simulations of small areas well, but quickly becomes overwhelming for scientists to manage on larger scales. These simulations are embarrassingly parallel because simulations of different tiles can be run in parallel with others. Large area simulations are performed by executing many of the point-based simulations as a collection. In order to achieve a high resolution simulation of a large area,

scientists are required to manage many simulations that compose a collection. Management of these simulations requires tedious attention to detail, including periodically monitoring running simulations, transferring data, correctly scripting configuration files for each model, and detecting failed simulations and handling the failures as appropriate. The management of these tasks is further complicated by our computational environment, as resources available to this project are located at two distinct locations: CU houses the allocated computational cluster and NCAR manages the storage systems and web portal user interface.

Prototype Architecture

The goal of our prototype architecture was to address the issue of global carbon cycle modeling complexity in our computational environment. The implementation of our prototype architecture is a federation of several independently functioning components (see Figure 2). These

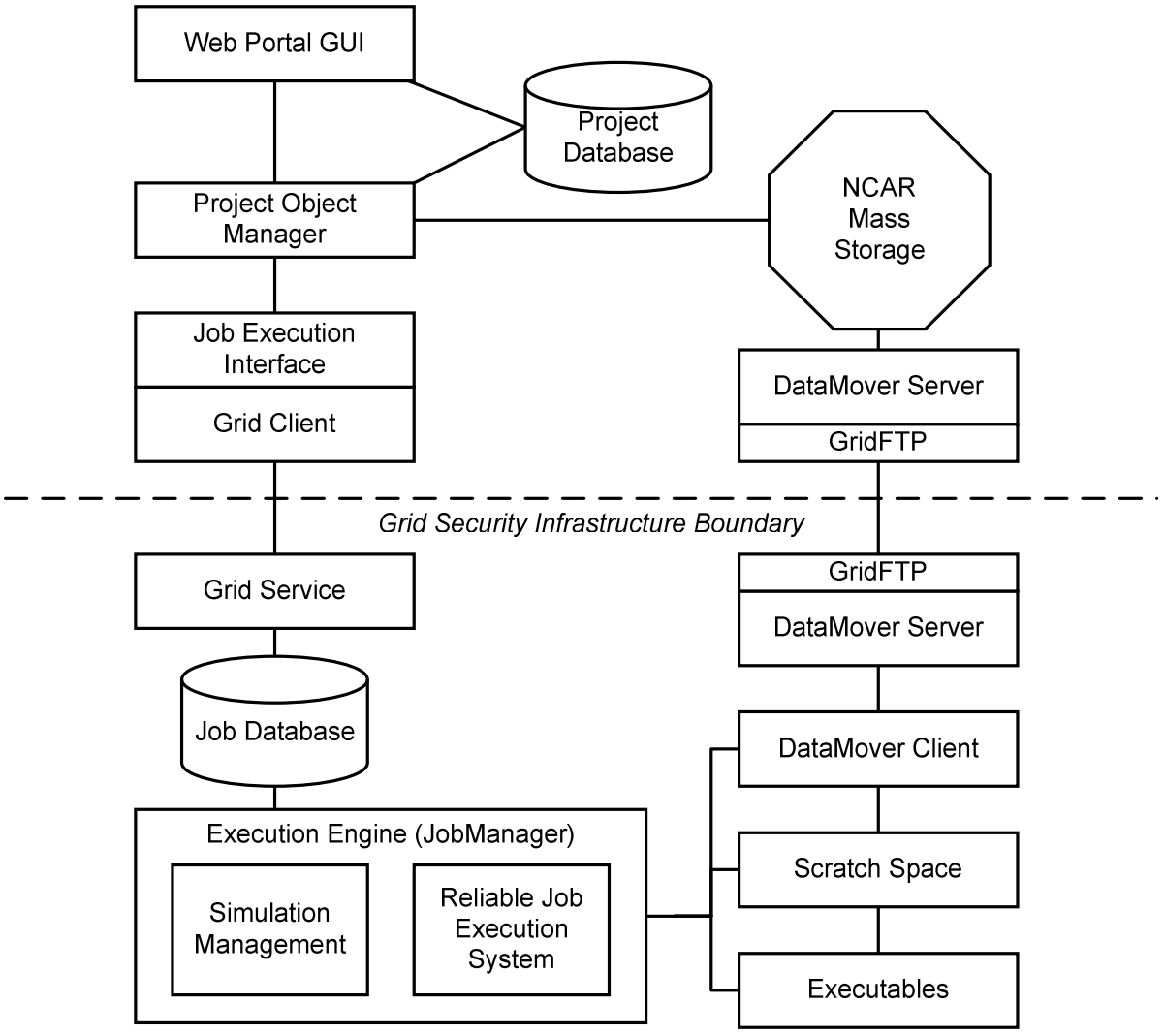

Figure 2, Grid-BGC prototype architecture

components include a web portal, a job management daemon, and a reliable data transfer utility.

The web portal provides a management interface to the Grid-BGC environment for both users and system administrators. The web-portal implementation requires no Grid-BGC specific software packages installed on the client machines since the web portal is maintained at a centralized location. The web portal allows users to specify simulation parameters, manage and monitor their simulations that are in progress, analyze results, and share results with other system users. A client embedded in the web portal communicates with the remote execution environment by invoking methods of a grid service residing on a computational resource. These methods include functionality to stop, start, and monitor simulations in the remote execution environment.

The job management daemon, known as the Grid-BGC JobManager, is used to monitor and coordinate the tasks executing in the computational environment. JobManager runs on the computational resources, such as a cluster. As the grid service receives communications from the web portal, the service parses and stores the requests into a persistent database. JobManager polls the database for new tasks to perform on the environment and the Grid-BGC tasks. The state of Grid-BGC simulations is periodically stored in the same database, so the grid service can query the database when task monitoring information is requested from the web portal. The decoupling of the grid service and JobManager and the use of a database to store system state allows system administrators to arbitrarily restart either component without loss of state, prior to the occurrence of a system failure.

The final component of the system is a data management utility, DataMover [11]. The utility was developed for the Earth System Grid and provides reliable file transfers, data caching, authentication, and interoperability with different storage architectures. DataMover uses GridFTP [1] as the underlying file transfer utility and implements several storage system access protocols, including a protocol to access NCAR's Mass Storage System (MSS). The Grid-BGC prototype uses this tool exclusively to transfer data between CU and NCAR. As the simulations are prepared, DataMover transfers the required input data sets to the computational facility at CU. Once the computations have completed, DataMover transfers the data back to NCAR for storage on the MSS or cached on the grid-enabled host at NCAR.

Experiences and Lessons Learned Implementing the Grid-BGC Prototype

Our prototype architecture successfully implements an end-to-end computational environment for Grid-BGC. The prototype currently manages a small portion of the workflow: the execution of the Biome-BGC model and the associated setup and finalization tasks. During the development of the prototype, we encountered several design problems that helped strengthen our future architecture of the system. Our experiences implementing the prototype include the need to implement a robust security scheme to accommodate the participating institutions security requirements, the need to create a reliable execution environment, and good practices for developing a grid-enabled computational environment.

Security Considerations

As is the case with many grid computing systems, a secure system is critical. While the grid computing middleware provides authentication and encryption mechanisms through the GSI security model, more security measures were needed. For example, NCAR not only requires that access to the MSS be authenticated, it also requires a great deal of accountability for each users

actions. A significant challenge in the prototype implementation was to accommodate the rigorous security requirements imposed by our organizations.

Grid computing system security is continually evolving and unfortunately must be reevaluated frequently. This includes addressing more advanced security needs and requirements posed by our organizations, such as one time password authentication, and the policies impacts on our system design. We have found that creating a flexible system design that can accommodate changes in security policy is essential in grid computing development. The flexibility allows security infrastructure to change with minimal impacts to other components of the system.

Reliability and Fault Tolerant Considerations

An essential quality of our computational environment is reliable execution of computational tasks. Fault tolerance in grid computing is being addressed in several areas, including workflows, data management, and task management. We found that integrating fault tolerant capabilities into our system software and grid services, in addition to the previously mentioned areas, strengthened our system design. Our grid service, daemons, and user submitted tasks store little state in volatile memory and log all critical state to a persistent database. With this design, these components can recover from faults by recovering the most recent state from the database and proceeding from this point.

Grid Computing System Development

Grid computing and the development of a grid-enabled environment was a surprisingly more difficult paradigm to become familiar with and acclimated to than we originally expected. We found that pleasant persistence in the face of frustration is an essential quality for the successful development of an environment similar to ours. The Globus toolkit provides many tools to grid developers, but usually at the cost of a complex programming model. From our experiences, a development environment capable of automating code generation for grid components substantially increases development productivity.

We also found that a developer's mileage with the grid middleware will vary. The middleware provides a set of tools that help cope with most grid computing systems, but the development of additional tools to support our application's needs was necessary. Instead of leaving the model configuration for the user to define, we found that development of a tool to automate the creation of configuration files reduced the possibility of human error and interaction with the system. We also found that it was most useful to utilize as much of the established grid middleware as possible. Use of these tools will allow our environment to be re-deployed and interoperate with other environments more easily.

Shortcomings of the Prototype Architecture

From our experiences developing and deploying the prototype, we identified several weaknesses in its design. The prototype architecture deploys a monolithic grid service and daemon that jointly perform all system tasks. A better design would break apart the distinct functions provided by these two components into several modular components. This approach would separate functionality and make the components more extensible for other uses. We believe that staging temporary data to the NCAR MSS is a misuse of the storage system. Only those files that need to be archived should be transferred to the MSS. The prototype is not completely Globus compliant. It uses a non-standard data management utility, DataMover, and Grid-BGC JobManager for execution management. A better approach would utilize the standard

components and minimize the software deployment to the climate models and the system services. Porting the prototype architecture would be easier if the third-party utilities were replaced by those that are packaged with the Globus Toolkit.

Production Architecture

Our production architecture addresses the experiences and lessons learned from the implementation of the prototype (see Figure 3). The fundamental goals of our prototype still are the same for the proposed production architecture: the system should be easy for scientists to use and efficiently execute global carbon cycle models. The significant advances we propose for the production architecture is the integration of more Globus Toolkit compliant services into our system, re-structuring data management policies, breaking apart the monolithic service structure to be more service oriented and modular, and re-developing the system from the most recent release of the Globus Toolkit.

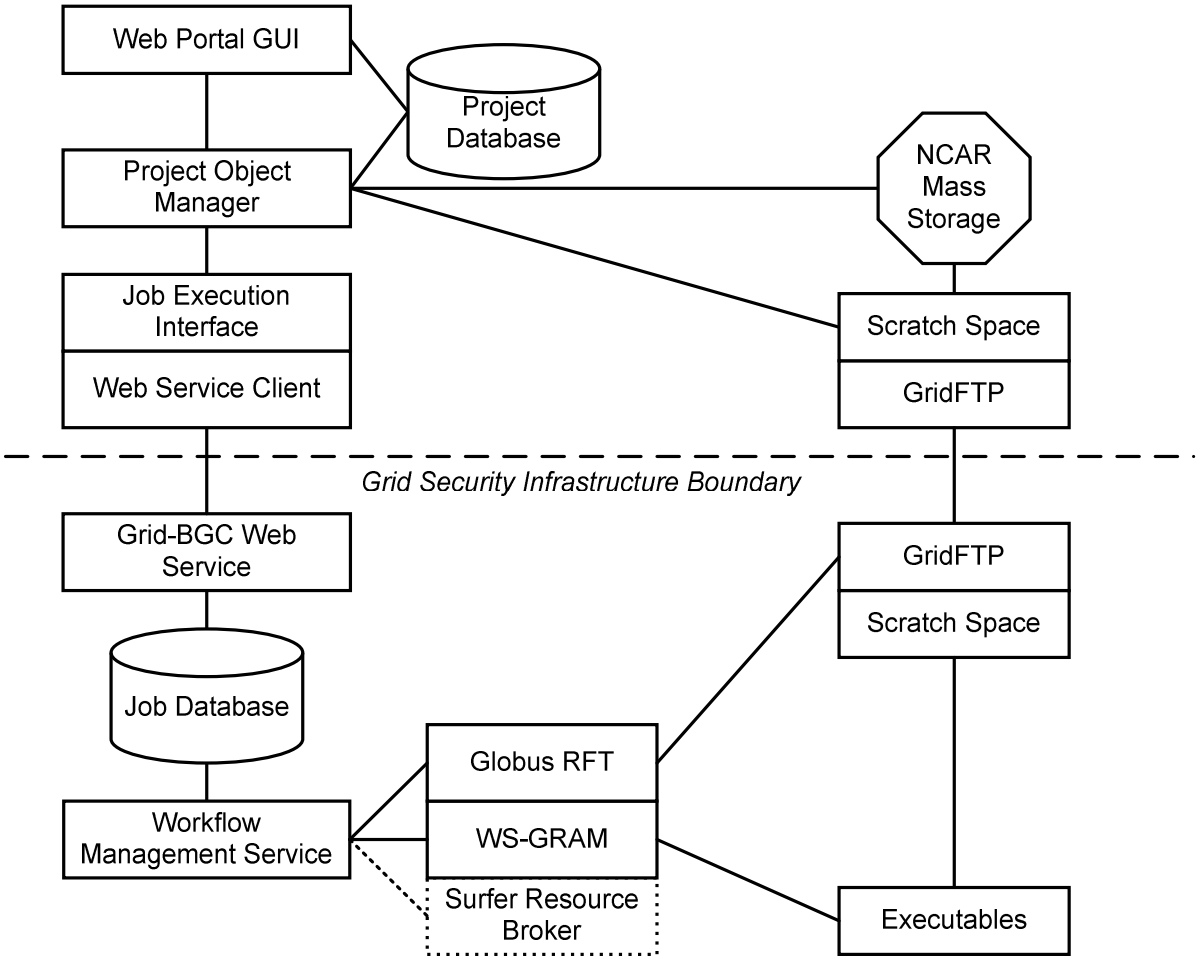

Figure 3, Proposed Grid-BGC production architecture

Our production system will take advantage of several standard services provided by the Globus Toolkit. Prior critiques of the systems design indicated that the use of WS-GRAM would help standardize the execution functionality provided by our system. Our production system will deploy a model execution web service using GRAM as the execution management engine. The service will provide the necessary mechanisms to setup and finalize the execution of the model specified in the web service configuration. WS-GRAM will be supplemented by the fault tolerant, reliable job execution, and management functionality developed during the prototype implementation. We believe that the integration of GRAM into our execution engine will make deployment of our system onto other grids much easier. Additionally, the use of the Globus data management will make our system more interoperable. We are also considering the integration of the Surfer [7] resource broker to help our user community and system allocate available resources as our environment expands from a couple of computational domains to several.

Our objective for restructuring the data management policies and functionality developed for the prototype are to create a more efficient data grid. First, we have decreased our reliance on the NCAR MSS as a staging platform. Instead of staging files through archival storage, we propose staging files using SAN-based scratch space at NCAR and manually archiving files to the MSS as needed. Additionally, we are considering removing DataMover from the system architecture. We believe we can replicate the essential qualities of it by using GridFTP, the Replica Location service, the Reliable File Transfer service, and the Data Storage Interface. Removing DataMover would eliminate the use of a third-party tool from our system design and enable us to use the more recent Globus compliant tools.

The production design also aims to make our system more modular and service oriented. Instead of a single web service to handle all tasks with the system, we propose to modularize the current monolithic grid service. We plan to break out services, such as security, file transfer, data management, job execution, and system management, from the single daemon implemented in the prototype. We believe that breaking out the services has many benefits including reducing the number of bottlenecks, making the system more manageable and scalable, and making the framework more accommodating for use by other grid-based projects in the climate modeling community.

Finally, the use of Globus Toolkit 4 (GT4) in our production prototype should help improve our overall system design for future use. GT4 is web service compliant, so re-development of our services to the standard web service interface will increase interoperability of Grid-BGC with other web service technologies. MyProxy [9] is now a standard component of GT4 and its addition will allow our security requirements to assimilate to other grid computing environments more easily. Significant improvements were made to the data management functionality of GT4, including striped server support for GridFTP, reliable file transfer through RFT, and modular support for non-GridFTP compliant interfaces. These improvements to GridFTP will enable us to better utilize our high-performance storage system through the use of striped servers and mitigate our reliance on DataMover through reliable file transfer support and the development of a modular interface to the NCAR Mass Storage System.

Future Work and Conclusions

Our future work includes developing the production system, deploying the system, and using the system to model the carbon cycle. At this time, we plan on completing the development and deployment of our system by the end of August 2005. Many of the components of the prototype

are still usable, including the web portal and many of the reliable execution components. Most of the production development will consist of porting code to GT4 and integrating new services into the system.

We found that the development of the Grid-BGC prototype has been productive, providing experience with grid computing development, security, and reliability. The prototype also highlighted weaknesses in our system design and helped us address these weaknesses in our proposed production architecture. We anticipate our user community will utilize the system to perform scientific studies by the end of October 2005.

Acknowledgements

University of Colorado computer time was provided by equipment purchased under DOE SciDAC Grant #DE-FG02-04ER63870, NSF ARI Grant #CDA-9601817, NSF sponsorship of the National Center for Atmospheric Research, and a grant from the IBM Shared University Research (SUR) program. NASA has provided funding for the Grid-BGC project through the Advanced Information Systems Technology Office (NASA AIST Grant #NAG2-1646) and the Terrestrial Ecology Program.

References

1. Allcock, B., Bester J., Bresnahan, J., Chervenak, A. L., Foster, I., Kesselman, C., Meder, S., Nefedova, V., Quesnal, D., Tuecke, S. Data Management and Transfer in High Performance Computational Grid Environments. Parallel Computing Journal, Vol. 28 (5), May 2002.
2. Cope, J., Hartsough, C., Thornton, P., Tufo, H. M., Wilhelmi, N., Woitaszek, M. Grid-BGC: A Grid-Enabled Terrestrial Carbon Cycle Modeling System, Euro-Par 2005, August 2005.
3. Droegemeier, K.K., K. Kelleher, T. Crum, J.J. Levit, S.A. Del Greco, L. Miller, C. Sinclair, M. Benner, D.W. Fulker, and H. Edmon, 2002: Project CRAFT: A test bed for demonstrating the real time acquisition and archival of WSR-88D Level II data. Preprints, 18th Int. Conf. on Interactive Information Processing Systems (IIPS) for Meteorology, Oceanography, and Hydrology., 13-17 January, Amer. Meteor. Soc., Orlando, Florida, 136-139.
4. Eerola, P., Kónya, B., Smirnova, O., Ekelöf, T., Ellert, M., Hansen, J. R., Nielsen, J. L., Wäänänen, A., Konstantinov, A., Ould-Saada, F. The NorduGrid Architecture and Tools. Proceedings of Computing in High-Energy and Nuclear Physics (CHEP 03), La Jolla, California, March 2003.
5. Globus. The Globus Project, 2004, http://www.globus.org/A
6. Kacsuk, P., Goyeneche, A., Delaitre, T., Kiss, T., Farkas, Z., and Boczko, T. High-level Grid Application Environment to Use Legacy Codes as OGSA Grid Services. Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing (GRID 2004), Pittsburgh, USA, 8 November 2004.
7. Kolano, P. Z. Surfer: An Extensible Pull-Based Framework for Resource Selection and Ranking. International Symposium on Cluster Computing and Grid 2004, April 2004.
8. Litzkow, M., Livny, M., and Mutka, M. Condor - A Hunter of Idle Workstations, Proceedings of the 8th International Conference of Distributed Computing Systems, pgs 104-111, June, 1988.

9. Novotny, J., Tuecke, S., Welch, V. An Online Credential Repository for the Grid: MyProxy. Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.
10. Milfeld, K., Guiang, C., Pamidighantam, S., and Giuliani, J. Cluster Computing through an Application-oriented Computational Chemistry Grid. LCI: Linux Revolution 2005, May, 2005.
11. Sim, A. J. Gu, A. Shoshani, V. Natarajan. DataMover: Robust Terabyte-Scale Multi-File Replication over Wide-Area Networks. Proceedings of the 16th International Conference on Scientific and Statistical Database Management, 403, 21 June 2004.
12. Thornton, P.E., S.W. Running, and M.A. White. Generating surfaces of daily meteorological variables over large regions of complex terrain. Journal of Hydrology, 190: 214-251, 1997.
13. Thornton, P.E., S.W. Running. An improved algorithm for estimating incident daily solar radiation from measurements of temperature, humidity, and precipitation. Agricultural and Forest Meteorology, 93: 211-228, 1999.
14. Tsaregorodtsev, A., Garonne, V., and Stokes-Rees, I. DIRAC: A Scalable Lightweight Architecture for High Throughput Computing. Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing (GRID 2004), Pittsburgh, USA, 8 November 2004.