The Astrophysics Simulation Collaboratory: A Science Portal Enabling Community Software Development

Michael Russell†
russell@cs.uchicago.edu

Gabrielle Allen*
allen@aei-potsdam.mpg.de

Greg Daues§
daues@ncsa.uiuc.edu

Ian Foster†¶
foster@mcs.anl.gov

Edward Seidel*§
eseidel@aei-potsdam.mpg.de

Jason Novotny‡
jdnovotny@lbl.gov

John Shalf‡§
jshalf@lbl.gov

Gregor von Laszewski¶
gregor@mcs.anl.gov

Abstract

Grid Portals, based on standard web technologies, are emerging as important and useful user interfaces to Computational and Data Grids. Grid portals enable Virtual Organizations, comprised of distributed researchers to collaborate and access resources more efficiently and seamlessly. The Astrophysics Simulation Collaboratory (ASC) Grid Portal provides a framework to enable researchers in the field of numerical relativity to study astrophysical phenomenon by making use of the Cactus computational toolkit. We examine user requirements and describe the design and implementation of the ASC Grid Portal.

1 Introduction

The Astrophysics Simulation Collaboratory (ASC) project seeks to provide a web-based problem solving framework for the astrophysics community to harness computational Grids. Computational Grids are emerging as a distributed computing infrastructure to promote the sharing of information, knowledge and expertise across institutional and disciplinary boundaries as well as to promote the collective use of large-scale computational resources towards the study of complex scientific problems. The ASC is motivated by a distributed community of astrophysics and computational science researchers working to link together software, computational and data resources to promote collaboration and problem solving in the field of numerical relativity. This community makes up a Virtual Organization [4] (VO), or more specifically what we define as the ASC Virtual Organization.

The Cactus computational toolkit provides a widely used software package for solving partial differential equations in 3D including Einsteins general relativistic wave equations. The Cactus framework provides support for studying a wide range of complex astrophysical phenomenon including the merger of neutron stars and collisions of black holes(Figure 1). The modular design of the Cactus framework permits a diverse group of researchers to develop add-on software modules, termed "thorns", to integrate additional physics or numerical solvers into the Cactus framework to solve specialized problems in general relativistic hydrodynamics, radiation transport, magnetohydrodynamics and nuclear astrophysics.

The goal of the Astrophysics Simulation Collaboratory is to enhance the problem solving capabilities of the Cactus computational toolkit and collaboration among members of the astrophysics community by providing an interface to easily access computational Grids shared by the ASC Virtual Organization. The proposed solution involves the development of an application specific Grid portal, a Web-based application server that provides access to Grid services and resources such as high-peformance computers, data storage archives, information servers and code repositories. Specifically, The Astrophysics Simulation Collaboratory Portal is driven by the following requirements:

[1]Max-Planck-Institut für Gravitationsphysik, Albert-Einstein-Institut, Golm (AEI)
[2]University of Chicago, Chicago, IL, (UChicago)
[3]Lawrence Berkeley National Laboratory, Berkeley, CA, (LBNL)
[4]National Center for Supercomputing Applications, Champaign, IL, (NCSA)
[5]Argonne National Laboratory, Chicago, IL, (ANL)
[6]Washington University, St Louis, MI, (WUSTL)

- *Low-cost access to collaboratory resources from portable, low-footprint ("thin-client") interfaces.* Members of the ASC community would like a more accessible and unified environment for utilizing the Cactus Computational Toolkit [23] to construct astrophysics simulations to be run on Computational Grids. Extending upon the additional 3-tier Web-based to deliver application services was deemed a suitable approach for building such an environment, given the pervasive nature of Web technology today. In the case of the ASC Portal, the goal is to deliver *Web services* that utilize *Grid technologies* to link users with remote resources.

- *Ability to cooperate effectively on the creation and execution of complex simulation codes.* The Grid enabled application server addresses this requirement, which is critical to our ability to meet ASC goals, by developing new services for online access to and management of *code repositories* and by providing access to Grid *resource management services* including scheduling systems on high-performance computers in order to execute simulations developed from the Cactus computational toolkit.

- *Flexible, secure, dynamic sharing of resources across institutional boundaries.* The ASC Grid Portal must provide services that can take advantage of deployed Grid infrastructure including Grid Information Services (GIS) and the Grid Security Infrastructure (GSI) in order to obtain information and securely access distributed resources across multiple Grids. The ASC Portal builds on existing *Grid technologies* including those provided by the Globus Toolkit [2] and Myproxy online credential repository.[5]

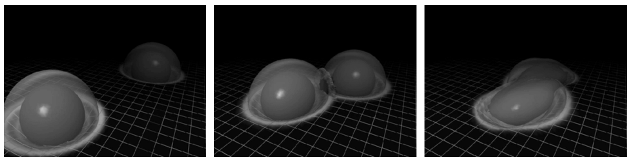

Figure 1. The ASC project investigates astrophysical phenomena involving strong dynamic gravitational fields such as this grazing collision of two neutron stars. The surfaces represent the density of the stellar matter while the warping of the gridded surface represents the gravitational flux on the plane of the collision.

2 ASC Portal Overview

We are developing a Web Portal-based application environment that addresses many of the goals described above by enabling users to work with the Cactus astrophysics development framework in a location independent manner. Here we implement an N-tier architecture that links *Internet browser clients* to a *Web application server* running our own organizational Web site. This application server provides our users with centralized access to services running computers located all over the world and connected together via Grid infrastructure.

Such an architecture is the basis for a class of application services called Web Portals. Web Portals are common to support communities of users with similar interests, such as the astrophysics community, by providing mechanisms for information sharing and access to distributed resources in a secure and well controlled manner. Many Web Portals allow users to customize the behavior of the application services so that they get the same view of their services regardless of their location.

In order to overcome the practical obstacles of accessing and sharing resources across institutional boundaries we use Grid technologies. By using Grid technologies, in our case Globus [2] middleware, we are removed from the difficulties introduced by the tremendous variety of resource types, mechanisms, and access policies we would otherwise have to handle ourselves. We are therefore better able to focus our efforts on delivering a useful application environment. The ASC Portal is an instance of an emerging class of Web Portals, or Grid Portals, that utilize Grid technologies. These Portals extend upon the Web Portal concept, as a collection Web services for sharing access to information and online communication, to include Web-based access to high-performance computing and data-storage services. In serving the astrophysics community, the ASC Portal is also an example of what is sometimes called a Science Portal, with the NCSA Chemical Engineering Workbench [17, 18], SDSC HotPage [15, 16], and the Alliance Virtual Machine Room [19] being other examples.

The result is a system that allows researchers participating in an ASC "virtual organization" (VO) [4] to construct codes from components located at multiple sites, compile on any available compile server, run on any available computer, and share results with colleagues in a controlled fashion. In constructing the ASC Portal, we have incorporated ideas and lessons learned from the portal community to produce an environment that is optimized for our own problem domain. In the process we have innovated in a number of areas that we believe will have broad applicability.

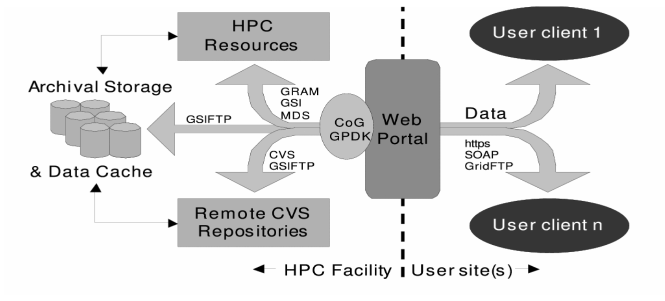

Figure 2. Web portal provides a central point of access to widely distributed Grid services.

3 ASC Portal Architecture

The ASC Portal is comprised of several software components that work together to provide a Grid enabled application server. In addition, the actual functionality provided by the ASCPortal is realized using the Cactus computational toolkit. As implemented, many of the services provided by the ASC Portal are generic enough to be useful for access to an entire different simulation code beyond Cactus. We provide an overview of the Cactus computational toolkit and an overview of other technologies required to build a Grid Portal as well as some useful supplementary technologies useful for Grid computing.

3.1 Key technologies

Cactus

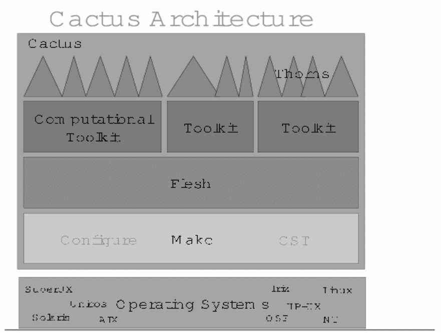

Figure 3. Diagram of the Cactus Architecture.

Cactus is a modular simulation code framework developed by the Numerical Relativity Group at the Max Planck Institute in Potsdam Germany [23, 24]. Cactus consists of the 'flesh', which coordinates the activities of modules, referred to as 'thorns' that "do the actual work"(Figure 3). The thorns interoperate via standard interfaces and communicate directly with the flesh (not with one another). This makes the development of thorns independent of one another, which is essential for coordinating large geographically dispersed research groups, and allows each group to concentrate on its area of expertise. The separation of *physics thorns* (e.g., black hole or hydro evolutions) and *CS thorns* (e.g., I/O, parallelism, AMR and performance monitoring) permit astrophysics with different levels of CS expertise to participate in the co-development.

Cactus offers several key advantages that make it ideal for Grid-wide use. Its modular design and sophisticated compilation tools support dynamic assembly of applications on a wide variety of platforms. In addition, Cactus is distributed with tools that support access to multiple code repositories for obtaining Cactus modules. Finally, Cactus applications may be constructed with a module that supports remote visualization and steering of running simulations through an HTTP interface, thus enabling seemless integration with our own client interfaces.

Java CoG and Grid services

The Java Commodity Grid Toolkit [6] provides a library for accessing Globus Grid services including the following:

- **Grid Security Infrastructure (GSI):** [**3**] Provides a secure infrastructure supporting mutual authentication and delegation using the IAIK libraries
- **Grid Information Services (GIS):** Provides access to information services using the Java Naming and Directory Interface (JNDI)
- **Globus Resource and Management (GRAM):** [**8**] Provides access to job submission to Globus gatekeepers running on Grid resources including batch scheduling systems on high performance computers
- **Grid FTP:** [**7**] Provides secure access to Grid enabled FTP servers for file transfer
- **Myproxy:** [**5**] Provides access to the Myproxy online credential repository for retrieval of certificates.

GPDK

The Grid Portal Development Toolkit (GPDK) [9] supports rapid development of Grid portals such as ours by providing a set of packages and tools for constructing Java Servlets / Java Server Pages (JSP) based application servers. It provides a set of middleware classes and Beans (Java components) for effectively managing the use of Java CoG libraries to gain access to remote services in a multi-user, multi-threaded application. In addition, it provides a generic infrastructure for building a Grid Portal with JSP and Java Beans. GPDK includes a robust project building scheme, based upon ANT [12], that offers tools for configuring Java Sevlets to run with Tomcat [11], a popular open-source Java Servlets implementation.

RDBMS

We use a relational database management system (RDBMS) to store persistent information about users and the activities they perform on the ASC Portal. In our case we use a popular open-source database server called MySQL. MySQL has gained wide-acceptance in the Web community because it is free for non-commercial use, well documented, very reliable, and provides very high-performance. However, in order to support interoperability with other relational database management systems we use the Java Database Connectivity (JDBC) package, which provides a generic interface to Standard Query Language (SQL) compliant database management systems.

MyProxy

MyProxy is a secure online certificate repository for allowing users to store short-lived certificates and then to retrieve those certificates at a later time from, say, an application server such as we employ in the ASC Portal, to enable that server to authenticate to remote services on behalf of users.

3.2 N-tier architecture

N-tier applications (Figure 4) divide functionality into 3 or more separate layers. Such applications are typically made up of *multiple clients*, an *application server* or a collection of application services made accessible on one host, services that act as *resource brokers* and the *resources* they mediate. This layering helps to maximize the sharing of resources among multiple clients because while the application server is designed to efficiently manage requests for resources among multiple clients according to the *business logic* of the application, resource brokers may be designed to optimize the use of resources according to the types of resources they manage.

Additional strengths of the N-tier architecture lie in the use of the application server to coordinate user activity. Application servers are "always-up" services that may be used to provide everything from fault-tolerance, say by properly handling requests to remote services that fail to complete, to building online chat services for allowing users to collaborate through the application server. Application servers may also be used to monitor user activity, as in sending email reports when to users when tasks complete or producing regular reports of user activity for determining what improvements may be necessary to better meet the needs of the community.

In the context of Web applications, client applications are developed in *thin-client* technologies such as DHTML or Java applets, which may be downloaded and run from within Internet browsers such as Netscape or Internet Explorer. There are many benefits to this approach. In developing client-side applications with with say DHTML, users can access application services from hosts installed with Netscape (4.0 and above) or Internet Explorer (4.0 and above). This allows developers to more easily introduce new application services as required, or even reimplement existing application services to utilize the latest Internet technologies, without the need to download or reconfigure client software on those client hosts.

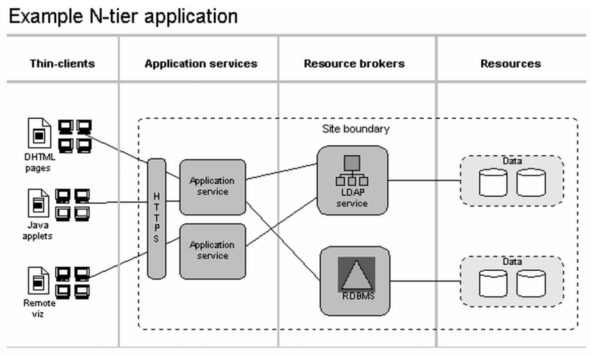

Figure 4. *N-tier applications divide functionality into layers in order to maximize the sharing of resources and services among clients.*

4 ASC Portal Implementation

In this section we discuss how we organize the key technologies describe earlier into an N-tier Web application architecture for providing application services that may be accessed from Internet browser clients (*ie. Internet Explorer 4.x+ or Netscape 4.x+*). This implementation enables users to gain access to collaboratory resources through portable thin-client interfaces. With Java CoG we can make it easier for members of the astrophysics to determine what resources they have access to and in addition to providing online tools for accessing those resources, we can offer up-to-date status information for how best to utilize them. GPDK provides the infrastructure we need to coordinate the use of Web technologies, Java CoG, and a persistent storage mechanism for effectivley delivering application services. This overall solution allows us to address the needs most central to the ASC community we wish to build and support, which are to provide an infrastructure to support community-wide development of astrophysics simulations with Cactus, distribute Cactus software and other useful applications onto remote hosts, and most important to enable users to control the use of those applications, all from within one unified environment.

Application server

The cornerstone of the ASC Portal is the application server. Our application server is implemented with Java Servlets / Java Server Pages (JSP) based techonologies. Java Servlets and JSP provide a powerful alternative to the more traditional CGI-based approach for delivering Web application services. Java Servlets hide the complexity of HTTP communication, providing a well-defined interface for manipulating HTTP sessions, while JSP provides a powerful yet easy to use mechanism for building dynamic Web-pages. These strengths, combined with the many advantages that the Java language environment offers, was a determining factor in our decision to use Tomcat, a popular open-source implementation of the Java Servlets 2.2 specification, to develop the ASC Portal application server. Tomcat acts as a container process, running in its own Java Virtual Machine (JVM), for running Java Servlets and serving documents written in JSP.

All communication with our application server however is mediated by a front-end Secure HTTP server (HTTPS). HTTPS secures HTTP communication with the Secure-Sockets Layer (SSL) protocol. Since users may send sensitive information to the ASC Portal application server, as in pass-phrases for retrieving certificates from our organizational MyProxy server, security is the most important issue we wish to address. To this end we use Stronghold, a commercial based distribution of Apache, the leading open-source Web server, that among other things makes it easier to administer a secure Web-server. Apache and Tomcat can be configured to communicate over the APJV12/13 protocol. This allows us to use Stronghold to encrypts communication between clients and the ASC Portal Web site, forwarding HTTP requests and responses between Internet browser clients and the Tomcat.

The application server is also supported by MySQL, an Relational Database Management System (RDBMS), that is located on a secure host accessible by our application server. MySQL provides a reliable persistent storage service for maintaining information about users and their activities in the ASC Portal.

Finally, the Grid Portal Development Kit (GPDK) provides a useful framework for connecting the technologies described above with Java CoG for accessing Grid services.

Supporting services

Several additional services are required to support the ASC Portal. The ASC maintains an organizational MyProxy server where ASC Portal users may store short-lived certificates, on the order of one week, with an associated certificate name and password. With an appropriate certiicate name and password, a shorter-lived copy of a certificate, on the order of an hour, may then be retrieved by an user upon "logging on" to the ASC Portal. This enables the ASC Portal application server to authenticate to remote services that accept that certificate on behalf of that user without additional intervention during the lifetime of the user's session on the ASC Portal.

We maintain an organizational Grid Information Index Server (GIIS) that provides an index to the collection of resources for which we want to enable access through the ASC Portal. When adding addtional resources to the ASC Virtual Organization, we contact that site administrators responsible for those resources and request that they configure their Grid Resource Information Servers (GRIS) to register with our GIIS. When ASC Portal want to obtain more detailed information about resources on remote machines, we may then query the GRISs running on those machines directly.

In order to enable users to acquire and distribute Cactus software onto remote machines, we provide access to the Cactus CVS repository (cvs.cactuscode.org) where the Cactus development team makes the latest versions of Cactus available to the general community

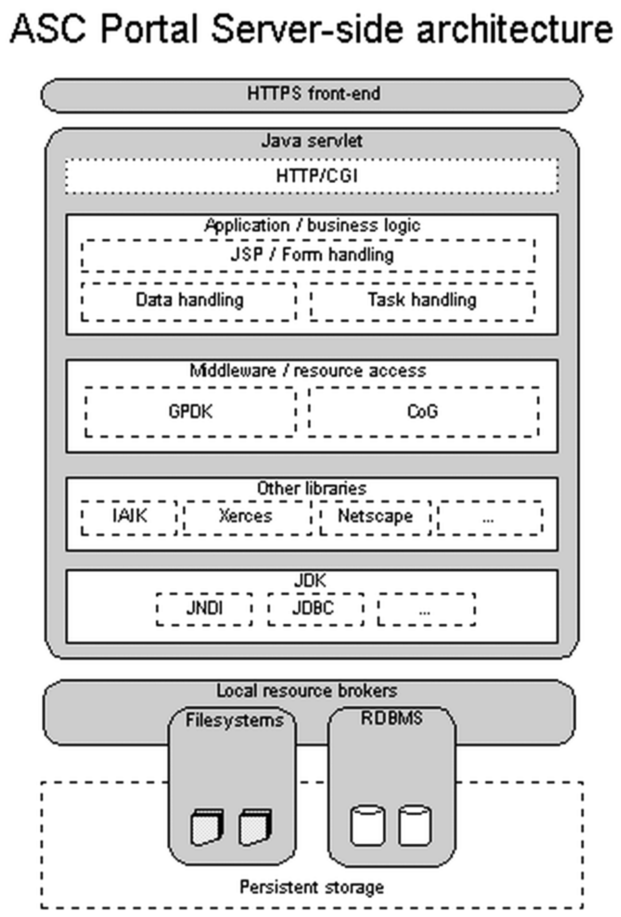

Figure 5. The ASC Portal server-side architecture and its software components.

by enabling anonymous access to that software. ASC Portal users may also gain access to restricted source code, if they are authorized to do so, through the ASC Portal.

Cactus

Cactus, as mentioned earlier, provides a modular, extensible framework for developing high-performance simulations. However, insofar as it relates to the ASC Portal, Cactus is the target platform with which ASC Portal users contruct astrophysics simulations. Thus the ASC Portal application server and its supporting services provide the necessary infrastructure for developing, distributing, and running Cactus simulations on remote resources.

Helper applications

Most of the application services the ASC Portal provides may be accessed with DHTML pages when users invoke our services through their Internet browsers. However, in some cases we would like to provide access to services that are ill-suited if not impossible to build client-side interfaces with standard Web-technologies. For example, Cactus simulations may be constructed with a Cactus thorn that enables datasets stored in Heirarchical Data Format (HDF5) to be streamed via ports that are configured in a parameter file provided to the simulation at startup. The ASC and others are building applications, such as Amira and LCAVision [25] for streaming and visualizing HDF5 data. By defining Multi-purpose Internet Mail Extensions (MIME) types for datasets produced by Cactus thorns, we can use those MIME types to instruct Internet browsers to launch the appropriate visualization application, if it exists on the client host, when a user attempts to access the data through the ASC Portal.

5 Application Features

Accesssing the ASC Portal

In order to gain access to services provided by the ASC Portal, users must logon to our services with a form-based logon through the ASC Portal Web site. Once logged on, users may retrieve their Grid credentials from our organizational MyProxy service at any time. However, they must do so before they may successfully complete most tasks on the remote resources we make accessible in the ASC Portal.

Personalized tools

- **User credential management:** Allows users to keep a list of user credentials they regularly use on our portal in order to more easily retrieve those credentials when they become necessary for authenticating to remote services.

- **User machine management:** Allows users to track what machines they have accounts on and where to install ASC software by default. The latter feature is utilized by our Cactus software services described below. Users may also perform various system tests on each machine with this service, for instance to test what credentials successfully authenticate to which services for each machine.

Cactus-oriented tools

- **Remote installation:** We offer tools for installing Cactus software and applications on remote sites as well as for "importing" existing installations into a user's personal environment. Users may work with multiple instances of Cactus per machine and are further enabled with access to online code repositories as well as means for replicating existing installations to additional locations.

- **Software repository access:** We provide a set of interfaces organized around the tools distributed with Cactus for obtaining Cactus modules from multiple CVS repositories. Furthermore, we allow users to store lists of Cactus thorns they regularly access. This makes it easy for users to quickly checkout those modules as needed for constructing Cactus simulations.

- **Software configuration management:** We offer tools for allowing users to maintain software configurations on a per machine basis, necessary for compiling Cactus source code into executables with the desired support libraries and compilation options. For example, such configurations may include the location of HDF libraries (Heirarchical-Data Format) on a particular machine.

- **Remote compilation tools:** A set of tools for interfacing to MAKE on remote machines for constructing Cactus simulations. These tools utilize the Thorn lists users maintain as well as the configuration information they keep per machine to quickly build Cactus source code into executables.

- **Parameter file management:** We make it easy to edit and distribute Cactus parameter files onto remote machines. These parameter files are used to activate specific thorns within a Cactus simulation as well as to initialize those thorns with parameters for directing the course of the simulation, among other things.

- **Remote simulation execution:** We greatly simplify the process of running Cactus simulations on remote resources. Our users may run simulations interactively or use our tools to construct batch scripts for submitting jobs to remote batch-queue systems.

Utilities

- **Resource inspection:** A basic tool built with GPDK to provide a structured view of the information that may be accessed through a GRIS in the latest MDS (Meta Data Service) model distributed with Globus.

- **Remote file browsing:** A basic tool built upon the Java CoG GSI-FTP client library for allowing users to browse remote file systems.

- **Custom utilities:** We make it easy for users to create and store their own tools that utilize one of the Grid services listed above for building interfaces to remote tools. Examples include using our GSI-SSH or GRAM client tools for interfacing to TAR, an archiving facility, or DF, a common Unix command for examining available diskspace on filesystems.

6 Future Directions

There are many new features we are actively working on providing to our users. This involves improvements to the user interface, advanced visualization support, data management tools, collaboration support, and development of new interaction paradigms.

One key area of development is access to more advanced and interactive client side user interfaces. Currently our primary Portal user interface is DHTML in order to ensure a universally accessible "thin client" interface that is well tested in the Web/e-commerce arena and requires virtually no investment by the user in configuring their own machine to use the client. However, DHTML is lacking in both interactivity and flexibility. We are therefore interested in supporting a wider variety of client-side interfaces including "fat client" and "slender client" interfaces.

Slender clients involve very simple downloadable clients such as Java Applets, Signed Java Applications or simple binary codes where the bulk of the interface remains on the server side. Slender client GUI's offer better interactivity and flexibility than DHTML, but require more time investment on the part of the user to ensure their client-side execution environment is capable of executing the code. The "fat clients" are traditional monolithic applications such as the OpenDX visualization tool. In these situations, the Portal merely acts as a broker for creating direct connections between the client and various Grid services. The Portal ends up acting as a sort of Napster for peer-to-peer interactions between clients and distributed resources.

A constant issue for the ASC user community is remote access to large scale visualization resources. One of the thorniest issues in supporting remote visualization applications is maintaining client software binaries for a large number of operating systems on workstations; each of which can come in a wide array of configurations. However, one common feature nearly all modern workstations is the presence of a Java-capable web browser. There are a also a variety of new data formats and browser plug-ins that have superseded VRML as a means to share 3D data on the internet. We would like to leverage off of the pervasiveness of these technologies to expand the accessibility to the Portal using a variety of techniques to improve resource-awareness and interactivity of the system. For example, the simplest approach would use entirely image-based mechanisms for serving visualization to the browser using image-push. Using javascript, one can manage a matrix of pre-computing images from a variety of angles. When the user grabs the object with a mouse, the javascript program can select the image of the most appropriate angle, providing to the user the illusion of interactive rotation of the object. Browser-embedded Java Applets can provide more sophisticated GUI functionality and more sophisticated latency-hiding mechanisms to further improve the sense of interactivity experienced by the user. Using Java's threading capabilities, the Applet can sustain interaction over extremely low-bandwidth connections by making background requests for view-dependent data/geometry reductions on the more powerful Visualization Server as the object is rotated by the user or by using continuous background downloads of imagery and geometry at progressively higher resolutions when the user pauses. The use of browser embedded thin Java clients have distinct advantage over traditional client deployment strategies in that the most recent version of the client is always downloaded to the user's workstation upon every use, which reduces software support problems. This will be an important consideration as we prepare for deployment of this technology to the ASC user community.

Another area of development is the creation of robust data management services, such as tools to assist users in finding appropriate storage space on remote computers, archival storage management services, meta-data browsing, and automated replication of data to high-performance disk caches to improve visualization system performance. This data management capability is critical for providing access to remote visualization services as well as sharing and understaning the large volumes of archived results produces by this community of researchers.

We are also very interested in developing tools that improve collaboration among our users using both synchronous and asynchronous methods. Asynchronous collaborative services would allow users to create and manage their own shared code repositories. An example of a synchronous services is a client application that supports multi-user shared access to running simulations and datasets in a controlled manner. This, however, is dependent somewhat on standard Grid-wide access control mechanism that would enable individuals to share access to data, software, and other resources with others across institutional boundaries.

In the long term we intend to explore alternative modes of interaction with our Grid Portal. One such scenario, we like to call Grid Pilot, involves a client application that allows a user to setup tasks offline that are scheduled to run once they reconnect to the internet and synchronize their client with the portal system. This is a familiar mode of operation for users of Palm Pilots and email clients like Eudora. This mode of operation is important for users of portable computers and users on increasingly pervasive wireless networks.

We continue to work on extensions that improve the robustness and generality of this architecture through continued collaborations with the Java CoG and GPDK teams, and other Grid Portal communities.

7 Conclusions

The ASC project serves a widely distributed community of researchers who all hold a common interest in using the Cactus computational framework to understand complicated astrophysical phenomena. We feel that our Portal implemnetation is an effective model for building any solution that entails distributing and managing data and software among remote resources utilizing standard Web and Grid technologies. These and other early Portal designs are critical steps in developing the software infrastructure necessary to make the widespread deployment of Grid Application Servers feasible. We expect Web-based Grid Application Servers will become increasingly pervasive as this software infrastructure matures and become the leading paradigm for High Performance Computing application environments. The ASC, along with other Science Portal efforts, are on the cusp of a revolution that is going to change the way we interact with HPC environments for solving the Grand Challenge problems of the future.

References

[1] National Research Council *National Collaboratories: Applying Information Technology for Scientific Research*, (National Academy Press, 1993)

[2] I. Foster, C. Kesselman, *Globus: A Metacomputing Infrastructure Toolkit*, Intl J. Supercomputer Applications, 11(2):115-128, 1997.

[3] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke, *A Security Architecture for Computational Grids*,

Proc. 5th ACM Conference on Computer and Communications Security Conference, pg. 83-92, 1998.

[4] I. Foster, C. Kesselman, S. Tuecke, *The Anatomy of the Grid: Enabling Scalable Virtual Organizations*, (*to be published in Intl. J. Supercomputer Applications, 2001*).

[5] MyProxy web site: *http://dast.nlanr.net/Features/MyProxy/*

[6] Gregor von Laszewski, Ian Foster, Jarek Gawor, Warren Smith, and Steve Tuecke. *CoG Kits: A Bridge between Commodity Distributed Computing and High-Performance Grids*, In ACM 2000 Java Grande Conference, San Francisco, California, June 3-4 2000.

[7] Grid Forum GridFTP Specification DRAFT: *http://www.sdsc.edu/GridForum/ RemoteData/Papers/gridftp spec gf5.pdf*

[8] K. Czajkowski, I. Foster, C. Kesselman, S. Martin, W. Smith, and S. Tuecke, *A Resource Management Architecture for Metasystems*, Lecture Notes on Computer Science, 1998.

[9] Grid Portal Development Kit: *http://dast.nlanr.net/Features/GridPortal*

[10] Documentation of Java Server Pages (JSP): *http://www.sun.com/products/jsp*

[11] Documentation of TomCat JSP Server: *http://jakarta.apache.org*

[12] Documentation of ANT build tool: *http://jakarta.apache.org*

[13] Documentation on Servlet API: *http://www.sun.com/products/servlets*

[14] ASCportal Homepage: *http://www.ascportal.org*

[15] M. Thomas, S. Mock, J. Boisseau, *NPACI HotPage: A Framework for Scientific Computing Portals*, Presented at the 3rd International Computing Portals Workshop, San Francisco, December 7, 1999.

[16] HotPage Home: *http://hotpage.npaci.edu*

[17] K. Bishop, B. Driggers, and J. Alameda, *Distributed Collaboration for Engineering and Scientific Applications Implemented in Habanero, a Java-Based Environment. Concurrency: Practice and Experience*, Vol 9, No 11, pp 1269-77, November 1997.

[18] K. Bishop, J. Alameda, *A Prototypical Example of the Chemical Engineer's Workbench*, AAAS/SURA/NCSA Meeting on Networking Resources for Collaborative Research in the Southeast,June 1998. http://www.aaas.org/spp/dspp/rcp/gamtg/agenda.htm.

[19] Alliance Virtual Machine Room Home: *http://www.ncsa.uiuc.edu/SCD/Alliance/VMR/*

[20] W. Benger, H.C. Hege, A. Merzky, T. Radke, E. Seidel, *Efficient Distributed File I/O for Visualization in Grid Environments*, ZIB Technical Report SC-99-43, January 2000

[21] G. Allen, W. Benger, T. Dramlitsch, T. Goodale, H.C. Hege, G. Lanfermann, A. Merzky, T. Radke, E. Seidel, J. Shalf, *Cactus Tools for Grid Applications, accpted for publication in Intl. Journal of Cluster Computing (2001).*

[22] Homepage for TIKSL project: *http://www.zib.de/Visual/projects/TIKSL/*

[23] G. Allen, W. Benger, T. Goodale, H.C. Hege, G. Lanfermann, J. Masso, A. Merzky, T. Radke, E. Seidel, J. Shalf, *Solving Einstein's Equations on Supercomputers*, IEEE Computer, Dec. 1999, pp. 52-59.

[24] Cactus Homepage: *http://www.cactuscode.org*

[25] LCAVision Homepage: *http://zeus.ncsa.uiuc.edu/ miksa/LCAVision/*