CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2002; **14**:1289–1301 (DOI: 10.1002/cpe.688)

Community software development with the Astrophysics Simulation Collaboratory

Gregor von Laszewski[1,*,†], Michael Russell[2], Ian Foster[1,2], John Shalf[3,4], Gabrielle Allen[5], Greg Daues[4], Jason Novotny[3] and Edward Seidel[4,5]

[1]*Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, U.S.A.*
[2]*University of Chicago, Chicago, IL, U.S.A.*
[3]*Lawrence Berkeley National Laboratory, Berkeley, CA, U.S.A.*
[4]*National Center for Supercomputing Applications, Champaign, IL, U.S.A.*
[5]*Max-Planck-Institut für Gravitationsphysik, Albert-Einstein-Institut, Golm, Germany*
[6]*Washington University, St Louis, MI, U.S.A.*

SUMMARY

We describe a Grid-based collaboratory that supports the collaborative development and use of advanced simulation codes. Our implementation of this collaboratory uses a mix of Web technologies (for thin-client access) and Grid services (for secure remote access to, and management of, distributed resources). Our collaboratory enables researchers in geographically disperse locations to share and access compute, storage, and code resources, without regard to institutional boundaries. Specialized services support community code development, via specialized Grid services, such as online code repositories. We use this framework to construct the Astrophysics Simulation Collaboratory, a domain-specific collaboratory for the astrophysics simulation community. This Grid-based collaboratory enables researchers in the field of numerical relativity to study astrophysical phenomena by using the Cactus computational toolkit. Copyright © 2002 John Wiley & Sons, Ltd.

KEY WORDS: collaboratory; Grid computing; Globus; Cactus; Java CoG Kit; Astrophysics Simulation Collaboratory

*Correspondence to: Gregor von Laszewski, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, U.S.A.
†E-mail: gregor@mcs.anl.gov

Contract/grant sponsor: National Science Foundation
Contract/grant sponsor: Office of Advanced Scientific Computing Research, U.S. Department of Energy; contract/grant number: W-31-109-Eng-38

Received 29 July 2001
Revised 6 January 2002

1. INTRODUCTION

Solving large complex problems often requires coordinated effort by geographically distributed participants. The term collaboratory was created in the early 1990s to denote an infrastructure supporting such collaborative endeavors [1]. Typically, collaboratories have emphasized interpersonal communication and collaborative access to relatively simple resources. We are interested in extending the collaboratory concept to address problem-solving methods and environments that require the sharing and coordinated use of large numbers of distributed resources. In this context, we see considerable value for Grid technologies, which address authentication, resource discovery and resource access issues that arise when sharing distributed resources. Grids have emerged over the past decade as a distributed computing infrastructure to promote the sharing of hardware, information, knowledge and expertise across institutional and disciplinary boundaries [2,3]. We term this combination of collaborative and Grid technologies a Grid-based collaboratory.

Although the results of our research can be applied to a variety of domains, we focus our attention here on computational astrophysics, a discipline that has challenging requirements for large-scale collaborative development and application of simulation, data analysis tools. For example, a typical simulation experiment might involve the development of a simulation code from a dozen distinct components, developed by different groups; the acquisition of computer resources at one or more remote sites; monitoring and steering of the computation as it runs; and access to data resources at one or more output locations. Our goal is to develop an Astrophysics Simulation Collaboratory (ASC) that allows astrophysics and computational science researchers to cooperate as they develop and share software, computational and data resources.

Discipline scientists are typically not experts in distributed computing. Hence, it is necessary that our collaboratory hide, as much as possible, the complexity of using multiple remote resources. In the past decade Web technologies have been accepted by many scientific communities as a means of distributing and accessing services. Web pages present the information needed in an orderly and structured fashion; Web protocols can be used to access remote information and services. Yet at the same time, Web protocols and technologies are inadequate for the sharing and coordination issues that arise when running simulations on remote systems, sharing multiple resources, and so forth. Here, Grid technologies [3] can play an important role. Commodity Grid (CoG) kits [4] provide components for accessing Grid services with commodity technologies such as Java, JSP, CORBA and Python.

In this paper we describe both the framework, services, and components needed to construct Grid-enabled collaboratories, and the particular application of these technologies within our ASC. The ASC [5,6] fulfills the following general requirements posed by the astrophysics simulation community.

- **Promote the creation of a community** for sharing and developing simulation codes and scientific results. The Grid infrastructure promotes the creation of virtual organizations (VOs) [3] within which resources are shared across institutional boundaries through Grid services.
- **Enable transparent access to remote** resources, including computers, data storage archives, information servers and shared code repositories.
- **Enhance domain-specific component and service development** supporting specific problem-solving capabilities, such as the development of simulation codes for the astrophysical community or the development of advanced Grid services reusable by the community.

- **Distribute and install programs onto remote resources** while accessing code repositories, compilation and deployment services.
- **Enable collaboration during program execution** to foster interaction during the development of parameters and the verification of the simulations.
- **Enable shared control and steering of the simulations** to support asynchronous collaborative techniques among collaboratory members.
- **Provide access to domain-specific clients** that, for example, enable access to multimedia streams and other data generated during the execution of the simulation.

2. ARCHITECTURE OVERVIEW

The ASC architecture has a modular structure and makes heavy use of Grid protocols and services to access and manage distributed resources (Figure 1). The principal components and protocols used are as follows.

- Web browsers serve as the primary end-user interface, serving dynamic HTML (D-HTML) and Java applets obtained from the ASC application server. Other end-user applications are also used: for example, for more advanced visualization functions (Figure 2).
- The ASC application server is the middle-tier entity that implements most ASC-specific logic, interacting with:
  - Grid services for purposes of resource discovery and authentication;
  - resources to initiate and manage simulation and/or access and analyze data; and
  - Web browsers and other applications for user interaction.
- An online credential repository, MyProxy [7], is used as a secure cache for user proxy credentials, hence reducing barriers to the use of PKI authentication.
- An ASC index server supports registration and subsequent discovery of resources contributed to the ASC, using MDS-2 protocols [8].
- Standard Grid protocols [3] provided by the Globus toolkit [9] are used to negotiate secure access to remote resources, including CVS code repositories.

The vast majority of ASC-specific logic is incorporated within the ASC application server. This component is responsible for handling all interactions with ASC users, for maintaining state associated with user interactions, and for coordinating the use of remote resources to meet user requests. Figure 3 shows its internal structure. Major components include the following.

- We make extensive use of commodity Web technologies such as Java Servlets and JSP.
- We use software development kits (SDKs) and APIs provided by the Java CoG Kit to execute the protocols used to access remote resources.
- We use new SDKs developed for this application (but of general utility) to address issues of code management, deployment and simulation execution on Grid resources.

The final major component of our collaboratory is the Cactus code [10], a modular simulation development framework that allows us to encourage and leverage the development of reusable components within the astrophysical simulation community.

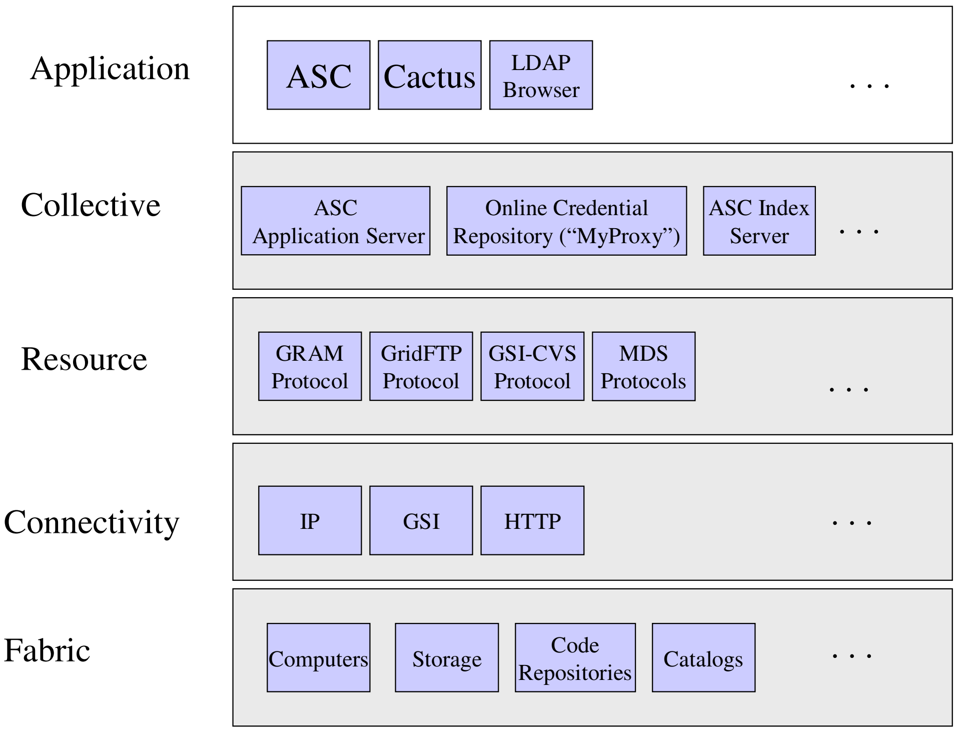


Figure 1. A protocol-oriented view of Grid architecture, showing the principal components used to build the ASC. Various components, in particular the ASC application server, are described in more detail in the text.

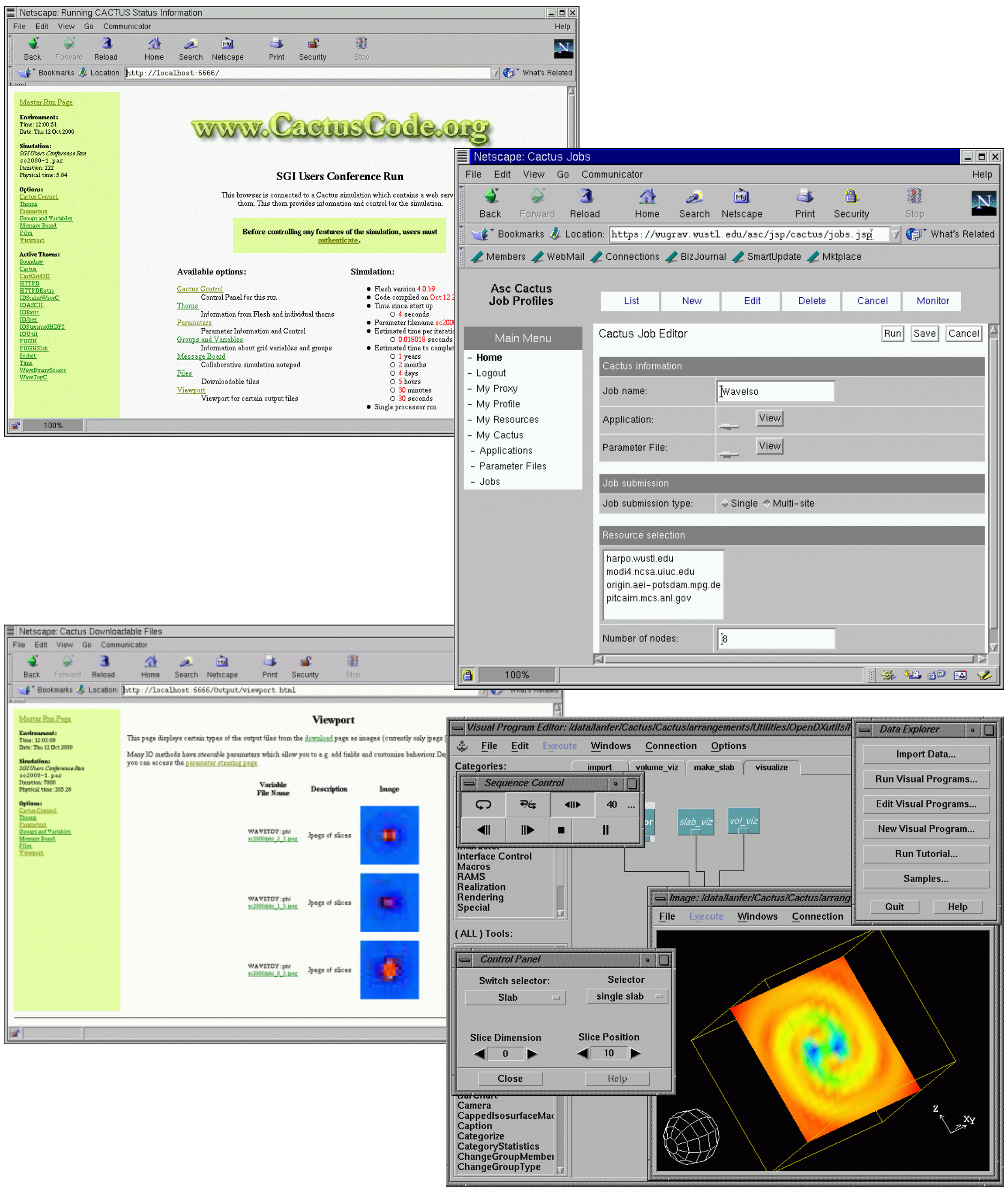
3. TECHNOLOGIES USED

We use three primary technologies to construct our collaboratory: the Cactus framework, to support development of modular simulation codes; the Globus toolkit, for Grid services; and various Java and Web technologies, to facilitate construction of the ASC application server.

3.1. The Cactus framework: modular simulation and code creation

Cactus is a modular simulation code framework [11] that uses component technologies to facilitate the construction and management of complex codes. The Cactus project has established its own terminology for naming different components. They term the code that coordinates the activities of modules the *flesh*, and a module that performs activities and is coordinated by the flesh a *thorn*.

Thorns interoperate via standard interfaces and communicate directly with the flesh, but not with one another. Hence, the development of thorns can be performed independently of one another, which is essential for coordinating component development in large geographically dispersed research groups



Location: http://localhost:6666/

This browser is connected to a Cactus simulation which contains a web server thorn. This thorn provides information and control for the simulation.

Location: https://wugrav.wustl.edu/asc/jsp/cactus/jobs.jsp

Job name: WaveIso

Job submission type: Single

Resource selection: harpo.wustl.edu, modi4.ncsa.uiuc.edu, origin.aei-potsdam.mpg.de, pitcairn.mcs.anl.gov

Number of nodes: 8

Location: http://localhost:6666/Output/viewport.html

Switch selector: Slab

Selector: single slab

Slice Dimension: 0

Slice Position: 10

Figure 2. The look and feel of the ASC exposed through a Web browser and a customized visualization component.

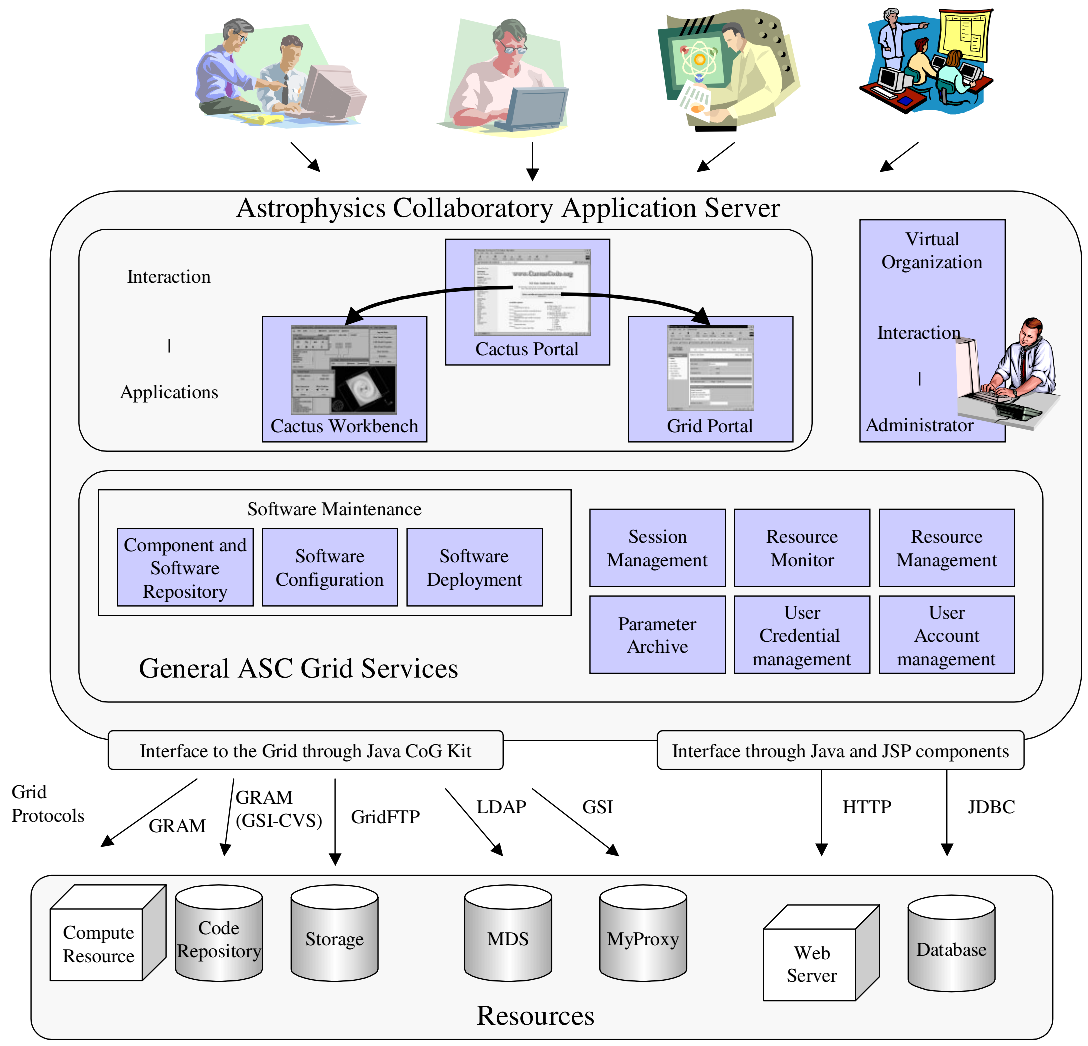


Figure 3. The structure of the ASC application server, showing the various components used to meet the requirements identified in the text.

while being able to integrate specialized thorns developed by a large variety of groups. This separation of the thorn development promotes, for example, the independent development of physics thorns to model the black hole or hydro evolutions, and information technology thorns that provide access to I/O, parallelism programming frameworks, or performance monitoring. Thus, the Cactus component model permits astrophysicists with different levels of IT expertise to participate in collaborative code development. The Cactus framework also supports sharing and cloning of thorns as part of multiple code repositories. Cactus is distributed with tools that support access to Cactus modules that are generally applicable to a variety of problem domains. An example provides a thorn that supports remote visualization and steering of running simulations through an HTTP interface to enable seamless integration with Web-based portals. Specialized components for the astrophysics community are readily available. Furthermore, the Cactus framework is used as part of the specialized service to perform astrophysics simulations. This includes distributing and running Cactus simulations on remote resources and is supported on a wide variety of platforms. We use Cactus as part of ASC to create a simulation code assembled from various thorns, contribute new thorns through a community process, and execute and monitor them on remote resources.

3.2. Globus Grid services: enabling resource sharing

By using Grid technologies, in our case Globus [12] middleware, we minimize the difficulties introduced by the tremendous variety of resource types, mechanisms and access policies, and the complex coordination issues (e.g. relating to authentication and authorization), that we would otherwise have to handle ourselves as part of developing a collaboratory.

The Globus toolkit [3,12] is a set of services and software libraries that support Grids and Grid applications. The Toolkit includes software for security, information infrastructure, resource management, data management, communication, fault detection and portability. It is packaged as a set of components that can be used either independently or together to develop useful Grid applications and programming tools. Globus toolkit components include the Grid security infrastructure (GSI) [13], which provides a single-sign-on, run-anywhere authentication service, with support for delegation of credentials to subcomputations, local control over authorization, and mapping from global to local user identities; the Grid resource access and management (GRAM) protocol and service [14], which provides remote resource allocation and process creation, monitoring, and management services; the metacomputing directory service (MDS) [8,15], an extensible Grid information service that provides a uniform framework for discovering and accessing system configuration and status information such as compute server configuration, network status and the locations of replicated datasets; and GridFTP, a high-speed data movement protocol [16]. A variety of higher-level services are implemented in terms of these basic components.

3.3. Java framework and server pages: developing browser-accessible thin clients

3.3.1. Java server pages/servlets

To enable rapid prototyping of Web portals as part of ASC, we base much of our Web-based services on the JavaServer Pages (JSP) technology [17]. JSP uses XML-like tags and scriptlets implemented in the Java framework to encapsulate the logic that generates the content for the page. We have exposed the functionality of the application server as a set of reusable JSP components that both provide direct access to Grid services and also implement higher-level services such as monitoring of the compute resources, execution of complex jobs with file staging and cleanup abilities and maintaining session state between ASC accesses. These components can be reused in other projects.

3.3.2. JDBC

Within the ASC application server we use a relational database to maintain the state of sessions. We express all database operations in terms of Java database connectivity (JDBC) [18] calls, which provide a generic interface to standard query language (SQL)-compliant database management systems. Thus, the ASC can support all commercially viable SQL-based servers, all of which are distributed with a JDBC provider. In our current implementation, we use the popular open-source database server MySQL which is free for non-commercial use, well documented, reliable and provides high performance.

3.3.3. Java CoG Kit

Within the ASC application server, we use SDKs provided by the Java Commodity Grid Toolkit [4] to access Globus toolkit services, as follows:

- **Security:** access to GSI services and to the MyProxy [7] online credential repository for retrieval of certificates.
- **Information services:** access to MDS information services through the Java naming and directory interface (JNDI) [19].
- **Job submission:** remote job submission and management through GRAM client APIs and protocols. This allows access to Grid resources including batch-scheduling systems on high-performance computers.
- **File transfer:** secure access to GSI-enabled FTP servers for file transfer [16] as well as access to GASS servers [20].

3.3.4. Other collaboratory technologies: collaboration through commodity technologies

We intend to integrate other collaborative technologies such as audio/video conferencing, chat rooms, news and e-mail groups. We have refrained from doing so in this phase of the project because the access to the Grid and the code repository was of most benefit to the user community we targeted first.

4. PORTAL-SPECIFIC SERVICES AND COMPONENTS

In this section we discuss services and components that we have developed to support the Web-based collaboratories. In all cases we rely upon GSI security wherever possible.

4.1. Secure login

In order to gain access to services provided by the ASC, users logon to the ASC portal with a secure Web form. Once logged on users may retrieve their GSI proxy certificates from the ASC MyProxy server at any time. From there, users can make use of our application services without the need to supply additional passwords until either the user logs off or their GSI proxy certificates expire.

4.2. User management

We have developed classes for maintaining information about users of the collaboratory and the tasks they perform. Users can track the remote jobs they have started for instance, or the file transfers they have performed. Users can also maintain a list of GSI certificates they use to gain access to remote resources and easily determine to which services their certificates successfully authenticate.

In addition, users can be granted or denied access to the functions of our Web-based collaboratory, in particular for enabling select users to administer the ASC application server, including adding or removing users from the collaboratory, monitoring user sessions and starting or stopping application server tasks.

4.3. Grid resource management

We attempt to model computational resources and the services they support in an intuitive way, by providing classes that abstract the details of the MDS queries required to gather information from remote resources and the services available on those resources. Additional information about each resource, such as the location of software not otherwise made available to MDS, can be maintained in the application server's primary database and published to users as needed.

4.4. Grid service management

As mentioned before, we can track what services are available on remote computational resources, where for each service we attempt to provide intuitive interfaces for invoking, testing and monitoring user requests to those services. In most cases, we extend upon the Java CoG Kit to allow user activities to be written to the ASC application server database and provide wrappers that unify the interfaces to those services.

4.5. Software management

We support commonly used tools for managing the development of software within the collaboratory. In particular, we provide interfaces for invoking CVS and MAKE on remote computers with interactive GRAM jobs, thus taking advantage of GSI security. In the case of CVS, we worked with the Globus project to develop GridCVS. GridCVS extends the CVS protocol and service to include GSI-authentication and we are developing a Java-based client that allows checkouts/checkins to be performed from/to GSI-enabled CVS repositories from third-party locations, where remote file browsing is performed with our GridFTP Java classes, extended from the Java CoG Kit.

4.6. Cactus management

Since the ASC supports application development with Cactus, we have developed tools that are useful for manipulating Cactus applications and data on remote resources from our Web-based collaboratory. Some of these tools are now described.

- **Parameter file management** tools to allow users to maintain a set of parameter files important for the execution of an application. A database is used to maintain the various parameter files and share them within the collaboratory.
- **Simulation management** tools to allow executing domain and non-domain Cactus jobs. Cactus jobs can be invoked as GRAM requests or be run from batch scripts that include pre- and post-processing instructions.

4.7. Visualization

We are developing tools that allow users to run applications for visualizing remote datasets. Currently, for those users running X-servers on their local desktop, we provide a Web interface that enables users to search for files with our DHTML-based GridFTP file browser on remote machines and subsequently launch visualization applications that exist on those machines, such as LCAVision, an application developed by the ASC for visualizing HDF5 data, with the application's display set to the user's local desktop. We have found this works well in practice. However, we are planning on developing additional tools for manipulating remote datasets so that desired portions of remote datasets can be viewed locally as well.

5. IMPLEMENTATION

Our portal can be accessed from any DHTML enabled browsers. We have tested our software with such popular browsers such as Internet Explorer v4 and Netscape v4. The Cactus software runs on almost all major operating systems. A prototype for Windows NT is available. New remote resources are brought into the collaboratory by implementing the Grid protocols and services used for remote discovery, job submission and data transfer. In general, these ports are provided by the Globus toolkit. The Web server is based on a secure version of Apache with JSP extensions. Persistent portal session management is achieved via MySQL.

6. APPLICATION

ASC was originally developed in support of numerical simulations in astrophysics, and has evolved into a general-purpose code for partial differential equations in three dimensions. Perhaps the most computationally demanding application that has been attacked with Cactus is the numerical solution of Einstein's general relativistic wave equations, in the context, for example, of the study of neutron star mergers and black hole collisions (Figure 4). For this purpose, the ASC community maintains an ASC server and controls its access through login accounts on the server. Remote resources integrated

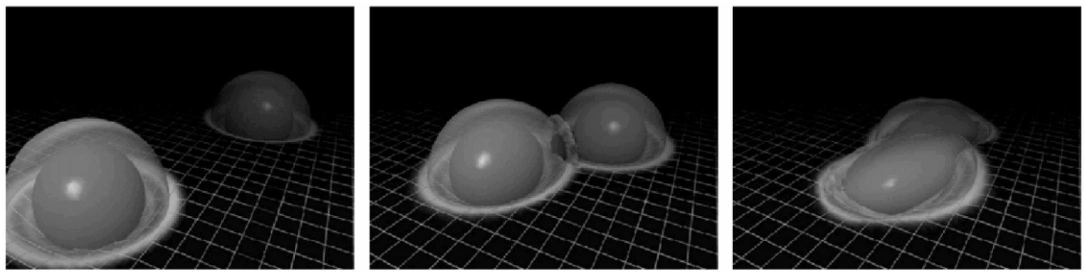


Figure 4. The ASC project investigates astrophysical phenomena involving strong dynamic gravitational fields such as this grazing collision of two neutron stars. The surfaces represent the density of the stellar matter, while the warping of the gridded surface represents the gravitational flux on the plane of the collision.

into the ASC server are controlled by the administrative policies of the cite contributing the resources. In general, this means that a user must have an account on the machine on which the service is to be performed.

The modular design of the Cactus framework and its easy exposure through a Web-based portal, as described in this paper, permits a diverse group of researchers to develop add-on software modules that integrate additional physics or numerical solvers into the Cactus framework.

7. RELATED WORK

Much research has been conducted in the area of portals. Most relevant to our efforts are the Swing based portal components distributed with the Java CoG Kit, the JSP based CoG Kit (GPDK), and the development of Jetspeed based portlets. More information about these projects can be found at www.cogkits.org and www.globus.org/cog.

8. FUTURE DIRECTIONS

We plan to improve our collaboratory in several ways, with enhancements to the user interface, advanced visualization support, data management tools, collaboration support and development of new interaction paradigms.

While testing our framework within the astrophysics users community we have observed limits of our initial thin clients based on DHTML. Therefore, we plan to investigate a wider variety of client-side interfaces including fat client and slender client interfaces. These will also include the use of Java3D as an alternative display framework. In cases where speed is essential, we use fat clients that are traditional monolithic applications, such as the OpenDX visualization tool. In these situations, the ASC application server merely acts as a broker for creating direct connections between the client and various Grid services. The application server, in this sense, would serve as a sort of Napster for peer-to-peer interactions between clients and distributed resources.

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, metadata 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 understanding the large volumes of archived results produces by this community of researchers. In the long term we intend to explore alternative modes of interaction with our Grid portal through asynchronous collaborative methodologies. We also plan to reuse the ASC framework in other application domains, for example, in high-energy physics.

9. CONCLUSION

We have described an online collaboratory framework that supports the collaborative development and use of numerical simulation codes. This framework uses a combination of familiar Web and Java technologies; Grid services for secure access to and coordinated use of distributed resources; and custom components for collaborative code development and distributed simulation. We have described an application of this framework within the astrophysics community, and suggest that the framework can also be applied within other application domains with similar requirements.

ACKNOWLEDGEMENTS

The development of an ASC joins five institutions (Washington University, Rutgers University, National Center of Supercomputing Applications, University of Chicago, University of Illinois) in an NSF funded Knowledge and Distributed Intelligence program. We thank the Numerical Relativity Group at the Max Planck Institute in Potsdam, Germany for developing the Cactus code. This work was supported in part by the National Science Foundation's Knowledge and Distributed Intelligence project 'Astrophysical Simulation Collaboratory', and by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under contract W-31-109-Eng-38.

REFERENCES

1. NRC. *National Collaboratories: Applying Information Technology for Scientific Research*. National Research Council, 1993.
2. Foster I, Kesselman C (eds.). *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 1999.
3. Foster I, Kesselman C, Tuecke S. The anatomy of the Grid: Enabling scalable virtual organizations. *International Journal of High Performance Computing Applications* 2001; **15**(3):200–222. www.globus.org/research/papers/anatomy.pdf.
4. von Laszewski G, Foster I, Gawor J, Lane P. A Java Commodity Grid Kit. *Concurrency and Computation: Practice and Experience* 2001; **13**(8/9):643–662. http://www.globus.org/cog/documentation/papers/cog-cpe-final.pdf.
5. Cartwright R, Dennis J, Hood R, Jump JR, Kennedy K. Parasol: A laboratory for parallel software technology research. *Technical Report TR86-41*.
6. Russell M, Allen G, Daues G, Foster I, Goodale T, Seidel E, Novotny J, Shalf J, Suen W-M, von Laszewski G. The Astrophysics Simulation Collaboratory: A science portal enabling community software development. *Proceedings 10th IEEE International Symposium on High Performance Distributed Computing (HPDC10)*, San Francisco, 2001. IEEE Press, 2001. http://www.cactuscode.org/Showcase/Publications.html.

7. Novotny J, Tuecke S, Welch V. An online credential repository for the Grid: MyProxy. *Proceedings 10th IEEE International Symposium on High-Performance Distributed Computing*. IEEE Press, 2001; 181–199.
8. Czajkowski K, Fitzgerald S, Foster I, Kesselman C. Grid information services for distributed resource sharing. *Proceedings 10th IEEE International Symposium on High-Performance Distributed Computing*. IEEE Press, 2001. http://www.globus.org.
9. Foster I, Kesselman C. Globus: A metacomputing infrastructure toolkit. *International Journal of Supercomputer Applications* 1997; **11**(2):115–128.
10. Allen G, Benger W, Dramlitsch T, Goodale T, Hege HC, Lanfermann G, Merzky A, Radke T, Seidel E Shalf J. Cactus tools for Grid applications. *International Journal of Cluster Computing* 2001; **4**(3):179–188. http://www.cactuscode.org/Showcase/Publications.html.
11. Allen G, Benger W, Goodale T, Hege HC, Lanfermann G, Masso J, Merzky A, Radke T, Seidel E, Shalf J. Solving Einstein's equations on supercomputers. *IEEE Computer* 1999; **32**(12):52–59. http://www.cactuscode.org.
12. Foster I, Kesselman C. The Globus Project: A status report. *Future Generation Computer Systems* 1999; **15**(5–6):607–621.
13. Butler R, Engert D, Foster I, Kesselman C, Tuecke S, Volmer J, Welch V. Design and deployment of a national-scale authentication infrastructure. *IEEE Computer* 2000; **33**(12):60–66.
14. Czajkowski K, Foster I, Karonis N, Kesselman C, Martin S, Smith W, Tuecke S. A resource management architecture for metacomputing systems. *Proceedings of IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing*, 1998; 62–82.
15. von Laszewski G, Fitzgerald S, Foster I, Kesselman C, Smith W, Tuecke S. A directory service for configuring high-performance distributed computations. *Proceedings 6th IEEE Symposium on High-Performance Distributed Computing*. IEEE Press, 1997; 365–375.
16. Allcock W, Foster I, Tueke S, Chervenak A, Kesselman C. Protocols and services for distributed data-intensive science. http://www.globus.org/research/papers/ACAT3.pdf [2001].
17. Hall M. *Core Servlets and JavaServer Pages (JSP)*. Prentice-Hall, PTR/Sun Microsystems Press, 2000.
18. Hamilton G, Cattell RGG, Fisher M. *JDBC Database Access with Java: A Tutorial and Annotated Reference*. Addison-Wesley: Reading, MA, 1997.
19. Lee SSR. *JNDI API Tutorial and Reference: Building Directory-Enabled Java Applications*. Addison-Wesley: Reading, MA, 2000.
20. Bester J, Foster I, Kesselman C, Tedesco J, Tuecke S. GASS: A data movement and access service for wide area computing systems. *Proceedings of IOPADS'99*, 1999.