SAND2000-1469C

RECEIVED
SEP 01 2000
OSTI

Constructing the ASCI Computational Grid

Judy I. Beiriger, Hugh P. Bivens, Steven L. Humphreys, Wilbur R. Johnson, and Ronald E. Rhea
Sandia National Laboratories[1], P. O. Box 5800, Albuquerque, NM 87185-1137
E-mail: jibeiri@sandia.gov

Abstract

The Accelerated Strategic Computing Initiative (ASCI) computational grid is being constructed to interconnect the high performance computing resources of the nuclear weapons complex. The grid will simplify access to the diverse computing, storage, network, and visualization resources, and will enable the coordinated use of shared resources regardless of location. To match existing hardware platforms, required security services, and current simulation practices, the Globus MetaComputing Toolkit[1] was selected to provide core grid services. The ASCI grid extends Globus functionality by operating as an independent grid, incorporating Kerberos-based security, interfacing to Sandia's Cplant™, and extending job monitoring services. To fully meet ASCI's needs, the architecture layers distributed work management and criteria-driven resource selection services on top of Globus. These services simplify the grid interface by allowing users to simply request "run code X anywhere". This paper describes the initial design and prototype of the ASCI grid.

1. Architecture description

The ASCI grid architecture provides the software infrastructure for an integrated information and simulation environment for the nuclear weapons complex in 2004. The Distributed Resource Management (DRM) project provides grid services to higher level applications. The initial implementation has two development thrusts. First, ASCI requires a core integrated environment for job submission to classified ASCI platforms at the three weapons laboratories: Los Alamos (LANL), Lawrence Livermore (LLNL), and Sandia (SNL). Globus provides these core services, with some extensions for specific ASCI requirements. Second, a software layer above the core services provides a set of common capabilities that support higher-level problem solving environments (PSEs). These capabilities include complex work management and resource brokering. This architecture demonstrates the following features:

- The use of Globus services for accessing resources in an independent ASCI environment
- Kerberos-based authentication, grid information authorization and accesss control
- A Globus interface to the two-tier architecture of Sandia's Computational Plant (Cplant™), a distributed computing resource built of commodity parts
- CORBA-based work management services to support the coordinated use of multiple resources and complex task sequencing
- CORBA-based criteria-driven resource brokering that extends job request/resource matching to support software resources and load-balancing
- Java-based client for generic work requests
- Web-based monitoring
- Integration of the grid infrastructure with CORBA, Java, and Globus-based PSEs and tools.

The DRM grid services address three aspects of the shift from platform-centric to network-centric computing: software resources, workflow, and co-scheduling. Software resources serve a large class of users and problems where many resources are suitable, and the primary interest is how soon the results can be obtained. Some simulation codes are run by a number of users at different sites, and must be available for many of the grid computing resources. Multiple versions of these codes are maintained to support different users and to ensure reproducibility of results. Requests to "run code X" are satisfied by software resources and brokering.

Workflow services manage complex task sequencing in the network environment, analogous to the complex scripts often submitted in a platform environment. Workflow can coordinate computational processing steps, computation and visualization, computation and data movement, or other resource usage. Subtasks are scheduled independently.

[1]Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Coscheduling of multiple resources will be needed to ensure that the critical resources in the ASCI grid achieve high utilization, to ensure that high priority jobs obtain resources as needed, and to enable new programming methodologies such as coupled computation and visualization.

2. Related work

The high performance distributed computing community has been researching computational grid concepts and advancing the technology in recent years [2]. The NASA Information Power Grid Project [3], which is advancing grid research into robust operational capabilities, is of particular relevance to the ASCI goal of establishing a grid-based production supercomputing environment. Globus [1] is a collection of infrastructure services that can be used to construct a grid, including services for resource discovery and information, monitoring, security based on the Generic Security System Application Program Interface (GSSAPI), and resource access. Legion [4] is an integrated distributed object computing system that provides security, storage, persistence, naming, and scheduling. Condor [5] finds idle cycles on networked workstations for high throughput applications.

Other researchers focus on software infrastructures that build problem solving environments and tools for parallel applications on top of grid services. Simulation Intranet [6] and WebFlow [7] are CORBA-based environments. Nimrod-G [8] uses Globus services directly. The use of Java for web-based distributed systems is being pursued extensively [9, 10].

3. Conceptual model

The ASCI grid architecture model divides needed software services into several layers and partitions as depicted in Figure 1. The model provides a target of the envisioned 2004 system to support an evolutionary development strategy. It may be abstracted as a three-tier model with domain-specific problem solving environments on top, grid infrastructure services in the middle, and resource interfaces on the bottom. Security and allocation policies must be considered at all layers of the model.

In the top tier, problem solving environments will present users with rich sets of tools and services pertinent to the problem domain. Users can focus on the scientific task, such as a product design or a parameter study, without necessarily dealing with system details such as data locality, resource availability, or platform-specific commands. Developers of PSEs, applications, and tools access the grid services layer for the management and allocation of high performance computing (HPC) resources.

In the middle tier, grid services provide the software infrastructure for accessing geographically distributed resources. These services include discovery, scheduling, reservation, allocation, monitoring, and control of collections of resources. Resource brokering services match resources to user requests based on implicit and explicit constraints. Users must receive consistent, fair, and responsive access to resources regardless of location, while resources must achieve high utilization.

In the bottom tier, an interface layer connects individual resources to the grid. The resources comprising the ASCI grid include heterogeneous computing, communication, storage, visualization, data, and software resources. This layer of the architecture isolates resource-specific interfaces and commands, providing higher level applications with consistent access mechanisms.

Security services and protection mechanisms are needed at all layers of the model. Allocation policies are traditionally implemented for individual resources. To enable coordinated use of resource sets, policy information and services must be available at higher layers.

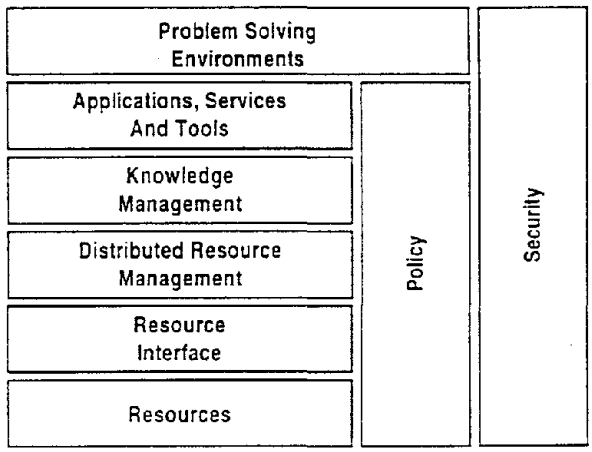

Figure 1. DRM layered architecture.

4. The ASCI grid

A prototype ASCI grid has been implemented in a limited testbed to demonstrate proof-of-concept. It is currently being extended and enhanced as the development environment for operational capabilities. The production grid, illustrated in Figure 2, will connect the laboratories and plants of the DOE weapons complex.

Initial integration prototypes have interfaced grid services to three existing problem-solving environments shown in Figure 3. The Simulation Intranet (SI) is a product design environment that provides web access to simulation, analysis, and visualization tools and manages a product-specific data repository. It is implemented in

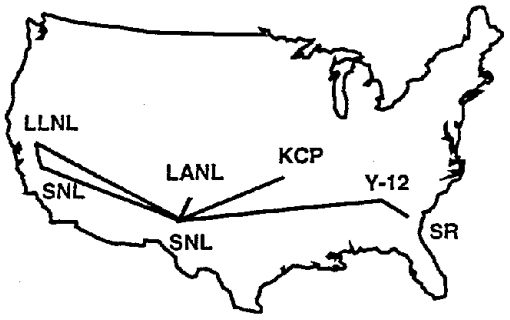

Figure 3. Planned ASCI grid.

Java and CORBA. Focusing on interactive use, the SI environment uses the DRM "run code X anywhere" capability to launch jobs on the most lightly loaded suitable resource. The Integrated Design for Exploration and Analysis (IDEA) environment provides web access to optimization tools for simulation codes. Java-based, IDEA was integrated with the grid monitoring services. Nimrod-G is a parameter study tool that provides its own brokering mechanism and interfaces directly with Globus. It was straightforward to integrate Nimrod-G into the ASCI grid; the same can be expected for other Globus-based tools.

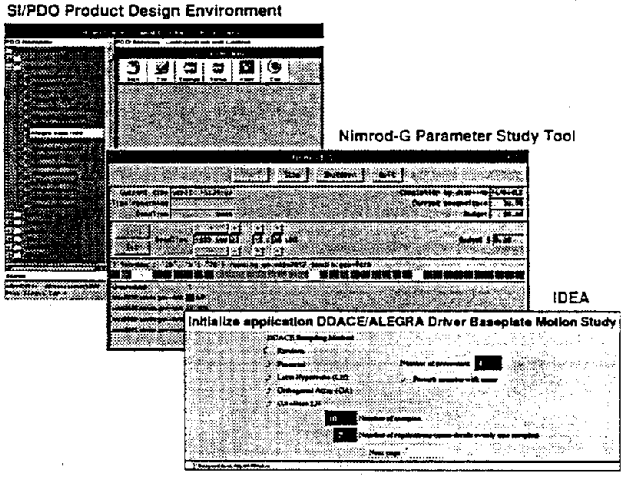

Figure 2. PSEs using prototype grid services.

5. Services for network-centric problem solving

DRM implements a Common Object Request Broker Architecture (CORBA) service layer on top of Globus. CORBA presents a standard way to componentize legacy or enterprise software in a distributed environment. This feature allows distributed deployment of work management and resource brokering capabilities in the grid. In addition, DRM implemented a Desktop Submission Tool in Java for users without specialized PSEs. The DRM component interactions are shown in Figure 4.

5.1. Work manager

The Work Manager hides the underlying complexity of the grid from scientific users and PSE developers by providing common services for resource usage. The Work Manager is also an autonomous agent that negotiates with the other DRM components to coordinate resource use and executes complex tasks in the ASCI grid on the client's behalf. The Work Manager accepts job requests, or work specification, from DRM clients. This work specification is represented as an eXtensible Markup Language (XML) document and provides users and PSEs with a grid-level "scripting" language. The work specification can consist of computations, file migrations, resource attributes and constraints, and task sequencing. A DRM client application creates a Work Manager on the server side and sends the work using the *submit* method of the Work Manager's CORBA API. The Work Manager generates a query to the Resource Broker, which returns the best match. The Work Manager uses the query results to construct a specific resource request in the Globus Resource Specification Language (RSL). Using the Globus Resource Allocation Manager (GRAM) Client API, the Work Manager submits the request to the target resource. The Work Manager receives status updates and redirected standard out and standard error streams from the GRAM and forwards these to the DRM client.

The Work Manager design supports the coordinated use of multiple resources. The initial prototype implements services for single resource requests and simple job management. It is currently being extended to support coallocation, serial, parallel, and conditional task sequencing. Lead-lag and deadline dependencies will be added as scheduling services are incorporated into the ASCI grid.

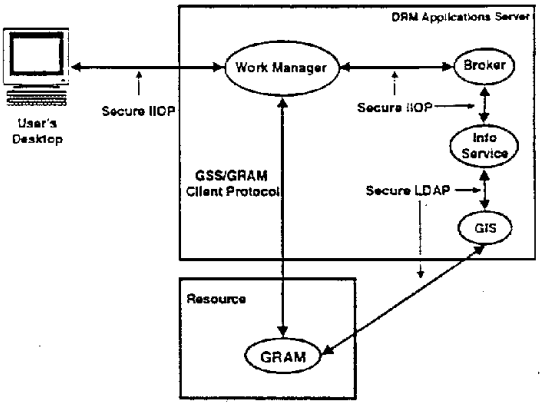

Figure 4. DRM's distributed object architecture.

5.2. Resource broker

The purpose of the Resource Broker is to select the "best" resource, allowing the user to specify only the minimum parameters of interest. The Broker locates the set of resources that the user can access and satisfy the attributes of the request. From the set of possible resources, the Broker chooses one based a small number of possible selection algorithms. With a default selection algorithm, it is possible for the user to merely specify "run code X". This contrasts with the common practice of selecting a machine or logging in directly, where the user must have prior knowledge of the resource. It also contrasts with the concept that a service is a software component installed on a resource [6], where again the user must have specific knowledge.

The Resource Broker first queries the grid information service (GIS) for a set of resources that satisfy the request. A CORBA interface to the LDAP-based GIS has been developed to support the Broker queries. The Broker then filters the set of resources according to the criteria in the request. Requests can specify a particular resource (called a "white pages" request), or can specify criteria that the resource attributes must satisfy (called a "yellow pages" request). The conditional operators (=, <, <=, >, >=) can be applied to values for numeric attributes, and the logical operators (&, +, !) can be applied to combine attribute criteria. Any object in the GIS – a person, an organization, software, hardware – can be brokered.

Several examples illustrate how request criteria are applied. The request "cn=blue-mountain.asci.lanl.gov-lsf" specifies a particular machine, the ASCI SGI platform at Los Alamos. The request "(&(&(osname=irix)(freenodes >=32))(!(hn=atlantis.sandia.gov)))" matches any SGI machine (including Blue Mountain) with at least 32 available nodes, but specifically rejects Atlantis. The request "sw=Alegra-2D" identifies all machines that can run that code. The Broker treats the user identity as an implicit constraint, filtering out resources where the user is not authorized.

Once a set of suitable resources is identified, the "best" one is selected. The DRM Resource Broker will provide a small number of selection algorithms that will serve most requests. An initial prototype algorithm was implemented to support interactive use by choosing the least loaded resource. This can also provide load balancing. For each suitable resource, the Broker calculates a load factor by weighting the CPU loads for the previous one, five, and fifteen-minute intervals and averaging against the CPU clock speed. The maximum load factor corresponds to the least loaded resource. The Broker returns to the requestor all of the attribute/value pairs in the GIS for this resource.

The second selection algorithm is being prototyped for a queuing environment by choosing the resource with the least average wait time for a given job type. The job type maps all possible job requests to a set of characteristic job types for which historical average wait time data will be maintained. Parameters that affect how long a job waits in a queue include the resources requested, such as number of processors and time, and the relative priority of the request with respect to competing requests. Requests can be prioritized according to the fair share concept of a service ratio (SR). The SR is implemented differently at the different sites, but in all cases is a normalized measure of how much of the resource a user has consumed relative to how much the user is entitled to. The effects of competing requests have been observed in cyclic patterns in the workload characteristics for a resource, such as the pattern of requests and behavior over a typical week of workday and night and weekend shifts.

For this selection algorithm, the job type will consider the following parameters: number of processors, time requested, SR, time of day, and day of cycle. For each parameter, the possible range of values is divided into a set of subranges, or bins. For example, time requested could be binned into 15-minute intervals, SR into tenths, and time of day into 1-hour intervals. An individual job request maps to a particular 5-tuple of parameter bins. A historical record of the average wait time for each 5-tuple will be maintained by the resource. The Broker will get the SR for the user, and then the average wait time for the job type. The average wait time will be adjusted for data movement and non-routine planned outages. The resource with the least predicted wait time will be selected.

The Resource Broker is being extended to support requests for collections of resources. An algorithm for closest match selection is also desired. When a resource request can not be satisfied, the closest match selection will attempt to identify a resource that is similar to the request.

5.3. Desktop submission tool

A desktop submission tool was developed for users without domain-specific PSEs. This is a Java-based GUI for obtaining DRM services. The desktop submission tool creates a work manager and sends it a work specification. The initial implementation allows a user to input a description for reference and to specify or select an application. Optionally, the user may input command line arguments, request input file staging via text entry or browsing, or specify resource constraints. Then the user may submit the job request, monitor job status, and display standard out and standard error streams that are sent back from the work manager. The tool is being extended to support requests for new types of resources like network bandwidth, collections of resources, and complex task sequencing. The initial prototype is shown in Figure 5 and is an example of a physics code running after submission to DRM.

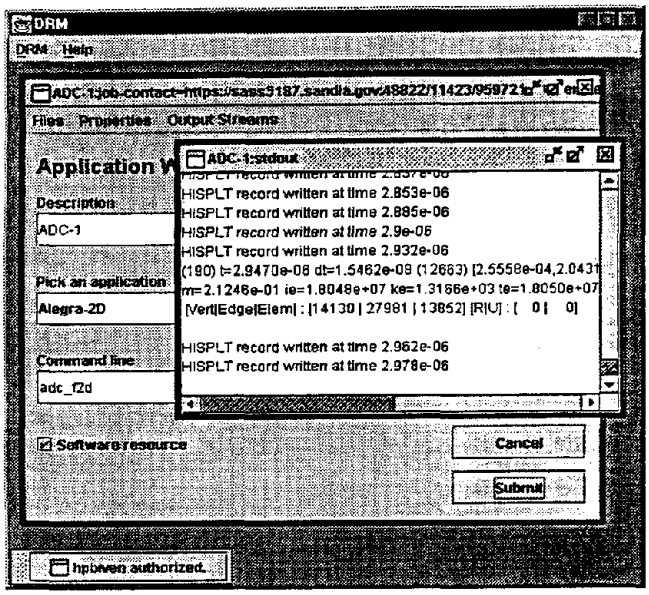

Figure 5. The desktop submission tool.

6. Extensions to Globus

The ASCI environment required extensions to Globus services. First, the Globus Resource Allocation Manager (GRAM) must be extended for computing resources with unique features. This ability was demonstrated by interfacing Globus to Cplant™. Second, monitoring extensions display node status and job attribute information. Third, ASCI requires Kerberos-based security services.

6.1. Cplant™ resource interface

One of the primary reasons that the Globus toolkit was chosen as the foundation of the DRM architecture was its ability to connect to existing resource management systems. The ASCI grid contains both resources for which GRAMs exist, and unique local managers that the Globus group has not encountered before. Fortunately the design has proven robust and accommodates these differences through a flexible data model.

The grid data model has two basic aspects that must be carefully designed and managed. First is the systems view of the grid: representations of resource attributes and state for efficient operation of grid applications. Second is the end user view of the grid: a collection of computational resources. These two notions are not always in agreement. One example is the Cplant™ architecture, which has multiple service nodes (front ends), a unique allocator called yod, internal and external network interfaces, and a customized version of Portable Batch System (PBS) as a local resource manager.

Because of the size of Cplant™, the number of users accessing it at any given time is large. Load balancing user connections today is achieved by cycling service node network names from a common name using the domain name system (DNS). This works well in a login environment since the users only have to know the common name, but quickly breaks down in a grid environment, as DNS will be unreliable due to intermediate lookups. The goal is to model the resource as a single system (the user view), while also describing the individual service nodes to Grid applications. In Globus this requires changes to the GRAM reporter and site-build daemon. Coordination of reporting features will eliminate duplicate information being written to the GIS and fail-over semantics will permit continued operation should a reporting service node die.

The yod allocator assigns nodes in an order that is efficient to the topology of its internal Myrinet interconnect. However, as a Cplant™ system grows, the compute nodes will not necessarily remain homogeneous. Disparate memory sizes, network speeds, and local disk space will determine whether a given set of nodes is desirable to the user. To request a suitable set of nodes in a heterogeneous system, changes to the node allocator, the GRAM site-build, GRAM reporter daemons, and job-manager are required.

Node allocator changes will expose the differences between compute nodes in such a way that it is possible to identify the varying attributes associated with each node and to specifically request a set of nodes. This will require that either a node identifier is bound to each compute node in the system, or a class of nodes is defined. At the time of job submittal, a set of identifiers or a node class is passed to the allocator. This approach also creates problems with the Cplant™ batch queuing system because the scheduler will have to discriminate which jobs want what nodes and when are they available.

The GRAM site-build daemon will have to be modified to report node configurations. The ability to discover node identifiers/classes and their associated attributes will allow the site-build daemon to report the current Cplant™ system configuration into the GIS, making them available to the Resource Broker. The GRAM reporter will need to report node consumption for brokering and monitoring purposes. Currently, Globus only reports the numbers of total and free nodes in such a system.

Because of changes to the data model, the job manager must know that it is running on a service-node and not on the abstract DNS definition (user view) in order to report the correct global job id back to the GRAM client. This is accomplished by ensuring that the domain name (DN) defined at the service node does not reference any of the other service nodes in the system.

The PBS installation on Cplant™ is also customized. Therefore, it is important to incorporate native management capabilities, most notably the pingd utility. A hook into the GRAM reporter provides information on Cplant™ node usage back to the GIS. In addition, because Cplant™ PBS uses yod features, the PBS_NODES reference in submittal scripts is currently unnecessary.

These architecture features are not unique to Cplant™. They exist in other ASCI resources and any complex installation with multiple front ends and heterogeneous resources.

6.2. Monitoring services

Monitoring services are presently supported through two mechanisms; the Heart Beat Monitor (HBM) and the myriad number of grid services that report status and information to the GIS. The HBM is a collection of small-footprint Unix deamons provided as part of the Globus grid computing toolkit. Local monitors on each of the grid-enabled resources and the DRM server periodically perform the Unix *ps* command, parse the output, and determine the status of the software services that have registered as HBM clients. Each local monitor then reports to the HBM data collector (located on the DRM server) the status of its clients. The information reported by the local monitors is limited to one of four states of health. The HBM client is reported as being either active, overdue (hasn't shown up in the most recent *ps* output), down, and unknown.

In contrast to the limited amount of information provided by the HBM, the GIS provides a wealth of information about the compute resource, queue status, job performance, and network status. Most of this information is obtained by the GRAM reporter that runs on each resource for each local resource manager (fork, PBS, NQS, DPCS, LSF, etc.). Other resource specific information is reported by the site-builder, a Globus provided Unix daemon that reports relatively static resource data. End-to-end network bandwidth and latency are reported by gloperf, another piece of Globus software that provides a grid-enabled interface to netperf, an open source network performance monitor.

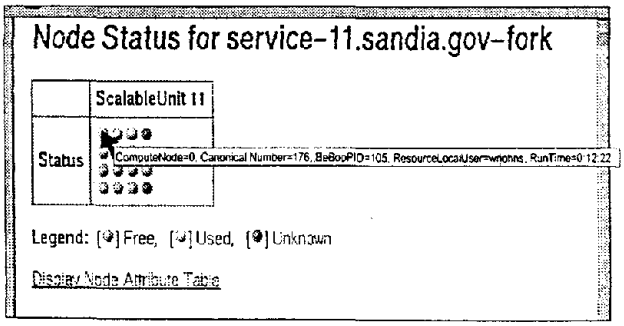

Figure 6. Web-based node status monitoring.

All of the above monitoring data is available to the user through a collection of web pages that access, parse, and present the data using CGI scripts which run on a secure web server. The use of web-based interface allows for the easy modification of the information presented. Figure 6 shows an example of CGI script customization that was used to display individual node status of ASCI's Cplant. Colored balls represent compute nodes. Yellow nodes are allocated, and display job information when placing the cursor over the node.

6.3. Security services

The ASCI grid must support classified computing in compliance with various Federal and DOE regulations. The grid services must communicate securely over multiple networks and between numerous compute resources. In the ASCI grid, Kerberos Version 5 is the primary mechanism for authentication. Kerberos and several other mechanisms are also used for protecting (encrypting) the data as it moves from grid service to grid service. The Sandia developed Generalized Security Framework (GSF) will be used to provide an abstraction layer over the Kerberos and other security libraries [11]. GSF implementations are available for both Java and C++ applications.

Users can connect via Secure Shell (ssh) to the DRM server or run grid enabled PSEs, frameworks, and applications, from their desktops. Grid-enabled software, such as the DRM Desktop Submission Tool, will establish authenticated and protected connections between user work stations and the DRM server using GSF secured Internet Inter-Orb Protocol (IIOP) connections. Users can also access a secure web server, to monitor the status of grid resources and submitted jobs.

The DRM Work Managers, Resource Broker, Information Service, and GRAM Clients will initially all be located on the same server. Connections between these services will be authenticated using Kerberos. The GIS will also be located on this server. Access to the GIS data must be authenticated and authorized. Access to the GIS LDAP server will be authenticated using a Netscape Directory Service (NDS) plug-in that provides a Simple Authentication and Security Layer (SASL) mechanism for authentication using GSSAPI over Kerberos. Access will be authorized using standard LDAP Access Control Instructions (ACIs).

Connections between the GRAM Clients and GRAM Gatekeepers will almost always occur across one or more network connections. Authentication will be accomplished using GSSAPI over Kerberos. Data protection will be achieved by using the Secure Sockets Layer (SSL) protocol. The remaining GRAM processes will use authenticated inter-process connections. Data transfers using the Kerberos authenticated and SSL

protected Globus Access to Secondary Storage (GASS) will also be supported.

7. Current status

An initial ASCI grid exists in a Tri-Lab testbed, and prototype grid services were demonstrated at SC99. The initial grid services include a core Globus-based environment for job submission and monitoring, and a layer of CORBA-based services, such as work management and resource brokering that support network-based problem solving methodologies. Current activities are focused on the security and robustness needed for production use. Fully integrating Kerberos into Globus is needed to obtain DOE security accreditation for use in classified networking environments. The robustness of the GIS is being enhanced through replication and referral. DRM is scheduled to be deployed on ASCI White, the new 10 Tops IBM supercomputer at LLNL, in November 2000.

8. Future work

DRM capabilities that provide sophisticated services for network-based problem solving will be added to the ASCI grid. Advance reservation, coscheduling, and workflow will allow scheduling and coordinated access to storage systems, network bandwidth, and visualization resources. A scheduler component will be added to the DRM architecture to realize grid-level scheduling. The Work Manager, Resource Broker, and GIS must also be extended to support reservations and requests for collections of resources. DRM will continue to collaborate with PSEs and participate in the Grid Forum.

9. Conclusion

DRM is developing and deploying a computational grid to support the ASCI mission. The Globus resource management infrastructure provides a highly sophisticated and extensible mechanism for adding and exploiting new and existing ASCI resources. The grid's CORBA layer on top of Globus aids scientists and problem solving environments by simplifying job management and resource discovery. All grid services communicate securely and access to the grid information service is access controlled. With these features, the ASCI grid will offer improved utilization of geographically distributed HPC resources in the Tri-Lab complex.

10. References

[1] Foster, I., and C. Kesselman, Globus: A Metacomputing Infrastructure Toolkit, *International Journal of Supercomputer Applications*, 1997

[2] *The Grid: Blueprint for a New Computing Infrastructure*, I. Foster and C. Kesselman, ed., Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1999.

[3] Johnston, W., Gannon, D., and Nitzberg, B., Grids as production Computing Environments: The Engineering Aspects of NASA's Information Power Grid, in *Proceedings of Eighth International Symposium on High Performance Distributed Computing*, August, 1999.

[4] Grimshaw, A., W. Wulf, and The Legion Team, The Legion Vision of a Worldwide Virtual Computer, *Communications of the ACM*, 1997

[5] Litzkow, M., M. Livny, and M. Mutka, Condor – A Hunter of Idle Workstations, in, *Proceedings of the 8th International Conference Of Distributed Computing Systems*, June 1988

[6] Holmes, V., J. Linebarger, D. Miller, and R. Vandewart, 1999, The Simulation Intranet Architecture, in *1999 International Conference on Web-Based Modeling & Simulation*, San Francisco, CA, January 17-20, 1999.

[7] Akarsu, E., G. C. Fox, W. Furmanski, and T. Haupt, WebFlow - High-Level Programming Environment and Visual Authoring Toolkit for High Performance Distributed Computing, *Proceedings of SC98*, 1998.

[8] Abramson D., R. Sosic, J. Giddy, and B. Hall, Nimrod: A Tool for Performing Parameterised Simulations using Distributed Workstations, *The 4th IEEE Symposium on High Performance Distributed Computing*, August 1995.

[9] Baratloo, A., M. Karaul, Z. Kedem, and P. Wyckoff, Charlotte: Metacomputing on the Web, Proceedings of the Seventh ACM SIGOPS European Workshop on System Support for Worldwide Applications, 1996.

[10] Christiansen, B., P. Cappello, M. Ionescu, M. Neary, K. Schauser, and D. Wu, Javelin: Internet-based Parallel Computing Using Java, Department of Computer Science, University of California, Santa Barbara, 1998.

[11] Detry, R., Kleban, S., Moore, P., and Berg R., The Generalized Security Framework, Presented at CSCoRE 2000, http://www.ccs.bnl.gov, Brookhaven National Laboratory, NY.