Data Grids, Collections, and Grid Bricks

Arcot Rajasekar, Michael Wan, Reagan Moore, George Kremenek, Tom Guptil
San Diego Supercomputer Center
University of California, San Diego
(sekar, mwan, moore, kremenek, tgpt)@sdsc.edu

Abstract

Data grids federate storage resources. They provide a logical name space that can be used to register digital entities, a storage repository abstraction for manipulating data, and a high level abstraction for supporting user-selected interfaces. Data grids can be used to build persistent collections. Data can be stored across multiple types of storage systems with persistent copies kept in archives. Persistent identifiers can be kept in the logical name space. By integrating Grid Bricks (commodity based disk caches) with archival storage systems, one can assemble a data management environment that supports both interactive access (data picking) and long-term persistent storage. Examples of the creation of interactive data picking environments will be given that integrate Grid Brick technology with large-scale archives.

1. Introduction

Data grids are middleware that tie together storage systems that are distributed across administration domains linked by wide area networks. The name "middleware" is used to define software systems that manage distributed state information. In the case of data grids, the state information is created by the registration of digital entities into a logical name space to denote their membership in a collection. From this perspective, data grids are the middleware software systems that support the creation of collections that span storage systems located in multiple administration domains (Stockinger, 2001).

The term "Data Grid" traditionally represents the network of distributed storage resources, from archival systems, to caches, to databases, that are linked using a logical name space to create global, persistent identifiers and provide uniform access mechanisms (Foster, 1999). Examples of data grids can be found in the physics community (PPDG, 1999, GriPhyN, 2000, Hoschek et. al., 2001, NEES 2000), in biomedical applications (BIRN, 2001), for climate prediction (Hammond, S., 1999) and for ecological sciences (KNB, 1999). More recently several projects have promoted the establishment of data grids for other communities such as astronomy (NVO, 2001), geography, and earth science for plate tectonic systems (EarthScope, 2001), etc. Most of these data grids are under construction and represent different proto-typical systems for building distributed data management environments. Many of these use the SDSC Storage Resource Broker (SRB) for their distributed data management.

The ability to create a data collection is heavily dependent upon the set of abstractions that are used to hide infrastructure dependencies. From the viewpoint of a user of a collection, a common interface is desired for discovering, accessing, and manipulating the registered data. The common interface eliminates the need to understand the protocol required by each type of storage system. The use of a logical name space to register the digital entities makes it possible to create global persistent identifiers that are independent of the physical file names used within the storage systems. Because the logical name space can be organized as a collection hierarchy, it is possible to create discovery and access methods that are more sophisticated than the mechanisms traditionally provided by storage systems. In particular, it is possible to build a data grid that manages a collection that is housed in an archival storage system (Rajasekar, 2001, Moore, 2001).

The utility of data grids can be described in terms of their ability to manage persistent data (Moore, 2002). It is possible to create and manage geographically distributed replicas of the digital entities that are registered into the collection. The naming convention for the digital entities can be global in scale, making it possible to use data grids to collaboratively share access to data. In addition to use as data sharing environments, data grids can also be used to support interactive access to persistent collections, independently of the archival storage system capabilities.

Since data grids can be used to federate access to storage systems, it is possible to integrate archival storage systems with commodity based disk caches, or Grid Bricks. The concept of a Grid Brick was proposed by Chaitan Baru of SDSC and Bala Iyer of IBM in April 2002. They defined a Grid brick as a storage appliance with a particular software stack already installed on it. We particularize the definition to mean a commodity disk cache on which data grid technology is installed to provide a logical name space. By replicating the digital entities onto both the disk caches and archives, one can simultaneously provide both long-term persistent storage, and interactive access to large collections. We will look at the management, scaling, performance, and cost issues associated with incorporating Grid Bricks into data grids. We will illustrate use of this technology as a third type of storage environment, providing persistent access for interactive "picking" of data from a large collection.

2. Storage Environments

The management and analysis of data collections requires multiple types of access environments (Moore, 1997). We will consider three data management environments: persistent storage, data streaming systems for data intensive analysis, and picking environments for interactive access to single digital entities. Persistent storage and disaster recovery support are traditionally provided by archival storage systems. Data intensive analysis systems support the processing of entire data collections by caching the collection on a high performance disk system, and then streaming the entire collection through analysis platforms. Data picking environments support interactive access to popular collections. A picking environment is needed to support the random retrieval of individual digital entities. The picking environment does not need the high performance network access required by data streaming environments, nor does it need the persistence required by archives. We already use data grid technology to integrate access to archives and data streaming systems. We can also integrate storage systems that are devoted to support for data picking into data grids. The solution we have examined for persistent access systems is the creation of commodity-based Grid Bricks that can be used to assemble low cost multi-terabyte disk caches. The Grid Brick is an extension of the idea of CyberBricks proposed by Jim Gray (Devlin, 2002).

The basic concept behind CyberBricks is that they can be aggregated as needed to provide the desired amount of disk space. For Grid Bricks to be feasible as an extensible storage medium, the administrative support needed to the manage user access must also be automated. Since each CyberBrick is a stand-alone system, containing not only the disk but also the associated controlling CPU and network access, data management systems are needed to provide a uniform name space across the CyberBricks and manage the distribution of files across the multiple systems. Grid Bricks integrate data grid management services with CyberBrick commodity hardware (Rajasekar, 2002). Data grids provide the ability to replicate data stored on the grid bricks into archives to provide long-term storage, and the ability to automate movement of data onto high-performance SANs for data intensive analyses.

A typical CyberBrick costs on the order of $3,500 to $4,500 per terabyte of disk cache, and contains the following components based on calendar year 2002 technology. As higher capacity disk drives become available, the cost will decrease dramatically.

- 1.7 Ghz CPU
- 1 GB of memory
- 3ware Raid controller
- 1-TB of disk (eight 160-GB IDE disk drives)
- Gigabit Ethernet network connection

CyberBricks typically run the Linux operating system, which requires user and file system administration, and operating system maintenance. The utility of CyberBricks for the provision of an interactive picking environment is strongly dependent upon the ability to minimize the administrative management overhead. Fortunately data grids make it possible to centralize user and file administration, and eliminate the need to separately administer each CyberBrick.

An environment that integrates archival storage, data picking environments, and high performance data analysis can be constructed by using data grids as the controlling data management mechanism.

3. Data Grids

The ability of data grids to simplify the administration of collections is strongly tied to the capabilities provided by logical name spaces, storage repository abstractions, and common data access mechanisms. A standard use of the logical name space is to support global persistent identifiers as a way to name the digital entities that are registered into the collection. The logical name space can also be used to register and manage the users that are allowed to access data within the distributed environment. The fundamental concept behind this approach to user administration is the use of collection owned data. To minimize the effort required to administer multiple CyberBricks, all data stored on the bricks are owned by a collection, and stored under a single collection Linux-ID. For this approach to work, users authenticate themselves to the collection, and the collection in turn authenticates its access to each grid brick. The logical name space is used to register user names, user passwords and authentication methods, and access control lists for each digital entity. This simplifies user administration of Grid Bricks to the installation of a single persistent collection Linux-ID on each Grid Brick. The data grid manages all interactions with users, eliminating the need for user administration on each Grid Brick.

The data grid storage repository abstraction is used to simplify administration of the file systems on the Grid Bricks. The organization of the files into directory/subdirectory structures, or equivalently collection/subcollection hierarchies, is managed within the logical name space. The Grid Brick disk space is maintained as a single large file system, without the need to differentiate into multiple directory structures for different types of user access requirements. All user access control is managed through the access control lists maintained in the logical name space. The only remaining administrative support required on the Grid Bricks is the installation of updates to the Linux operating system for security patches, and replacement of disks when they fail.

The persistence of data within Grid Bricks is managed by replication across multiple systems. The presence of a replica is registered into the logical name space, along with the IP address of the system on which the replica resides, and the access protocol that is required to manipulate the replica. The storage repository abstraction maps from the standard set of operations that are used to manipulate data, to the Linux I/O operations supported on the Grid Brick, as well as to the protocol used to interact with an archival storage system or a high-performance SAN. From the user perspective, the same API can be used with equivalent operations performed no matter where the data actually resides.

The data grid used at SDSC is based on the Storage Resource Broker (SRB) and associated Metadata Catalog (MCAT, 2000). The system is shown in Figure 1. The SRB provides facilities for collection-building, managing, querying and accessing, and preserving data in a distributed data grid framework (SRB, 2001, Moore & Rajasekar, 2001). It provides federation of storage systems and uniform access to diverse, heterogeneous storage resources across administration domains. The Metadata Catalog holds systemic and application- or domain-dependent metadata about the resources and datasets, and methods and users that are being brokered by SRB. Together, the SRB and the MCAT provide a scalable information discovery and data access system for publishing and computing with scientific data and metadata. The SRB provides a means to organize information from multiple heterogeneous systems into logical collections for ease of use. The SRB, in conjunction with the Meta data Catalog supports location transparency by accessing data sets and resources based on their attributes rather than their names or physical locations [Baru et.al., 1998]. The SRB provides a logical representation for storage systems, data, and collections and provides several features for use in digital libraries, persistent archive systems and in collection management systems. SRB also provides capabilities to store replicas of data, for authenticating users, controlling access to documents and collections, and auditing accesses.

The SRB can also store user-defined metadata at the collection and object level and provides search capabilities based on these metadata. In a nutshell, the SRB provides the following data grid functionalities (described in some detail in (RWM, 2002)).

- Integrate data collections and associated metadata, including system-metadata, domain-specific metadata and user-defined metadata.
- Handle multiplicity of platforms, resource and data types.
- Provide seamless authorization and authentication to data and information stored in distributed sites.
- Provide transparent access to resources and attribute-based access to data and collections
- Provide a virtual organization structure for data and information based on a digital library framework.
- Handle dataset scaling in size and number
- Handle replication of data and provide replicated data management,
- Provide caching, archiving and data placement facilities,
- Handle access control and provide auditing facilities,
- Provide remote operations for data sub-setting, metadata extraction, indexing, data movement, etc.

The SRB/MCAT system provides a storage repository abstraction, a high-level abstraction to support multiple user application interfaces, and a logical name space for the registration of logical entities. The high-level abstraction for the user interface defines the set of operations that will be supported in the data grid, including collection management, metadata manipulation, data management, data movement, attribute-based data discovery, and data access. These operations are interfaced to the storage repository protocols through the storage repository abstraction. The desired operations are sent to a server that is installed on the remote storage system. The user client issues requests that are interpreted by a SRB server, sent to the appropriate remote server, processed at the remote storage system, and the results sent back to the client.

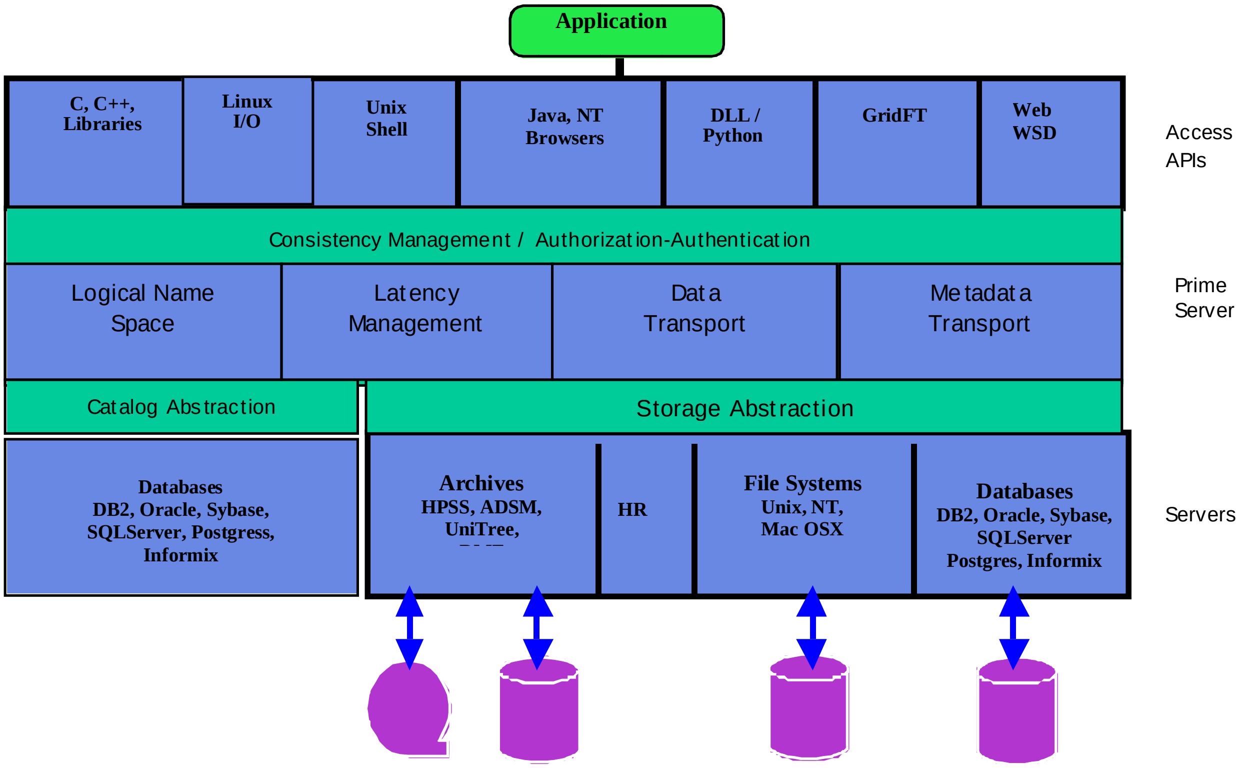

Figure 1. The SDSC Storage Resource Broker and Metadata Catalog components.

Through the data grid, the user effectively accesses data without knowing the file name (attribute-based query), without knowing the location of the data, and without knowing the access protocol required by the remote storage system.

4. Examples

The management of data for the National Virtual Observatory (NVO) builds upon all three data management environments, data archiving, data streaming, and data picking. The NVO project is funded by the National Science Foundation to provide access to images from multiple sky surveys. A typical sky survey contains 10 TBs of data, with up to 5 million images. All three modes of access are needed. For the 2-Micron All Sky Survey (2MASS), the SRB/MCAT system has been used to implement archiving, streaming, and picking environments. The collection is archived in the HPSS storage system at SDSC in 140,000 containers. To minimize the impact on the HPSS name space, the images were sorted into containers such that all images for the same area of the sky are stored in the same container. Access to the original images is managed through the SRB data handling system, which automatically retrieves the correct container from HPSS, caches the container on a disk system, and then reads the desired image from the cached container.

The data streaming environment is created by replicating the collection onto a high performance Sun SAN disk system. The data is copied from HPSS onto the SAN using parallel I/O transfer over gigabit-Ethernet networks. Once the data resides on the SAN, then analyses are made over the entire collection using a terascale IBM SP computer.

The data picking environment is created by replicating the collection onto Grid Bricks. The SRB/MCAT system can be told which replica to access. If no direction is given, the SRB preferentially accesses data that resides on a disk file system to provide the lowest possible latency. The data picking environment is being implemented in support of NVO mosaic services and image access services. These services provide web access to the entire collection, allowing users to pick the region of sky that is of interest, and dynamically retrieve selected images without incurring archival storage access latencies.

5. Grid Brick Building Experiences

Since the whole point of the Grid Brick was interactive access for data picking, a disk-based system was required. We decided at the beginning that capacity was more important than speed. We tested IBM's 7200-RPM 75-GB drives, but our initial terabyte systems were based on the Maxtor DiamondMax 81.9GB drives. When Maxtor released their 160GB drives, we began using those instead – although they were slightly more expensive per gigabyte than the smaller units, this additional expense was offset by a reduction in the number of nodes we needed to deploy for a given application. Since these drives allow us to build terabyte-scale arrays with only eight drives instead of sixteen, we can use a standard PC case and power supply instead of the more expensive server cases and multiple power supplies that we used in our original test units. Although the larger drives run at only 5400 RPM, spreading the data out across multiple spindles helps compensate for the slower speed. We have seen peak RAID-0 read performance of well over 100MB/s from eight of these drives. The Maxtor 160 GB drives have additional advantages, including peak (spin up) power requirements that are less than 1/2 that of the 81.9 GB drives and operating power requirements of less than 12 watts. With lower peak and operating power requirements, we can use a smaller power supply and we have almost no problems with heat. Indeed, eight 160 GB drives actually use slightly less power on average than a single Athlon or Pentium 4 CPU running at full speed.

In the early stages of our work, we did our RAID management in software using the Linux kernel "md" driver. This worked reasonably well for small servers, but we came to a consensus that doing this on a machine that was also likely to be a busy file server would have a significant performance cost. We note that transferring data from a Grid Brick at 120 MB/sec can use all of the CPU power. We also had trouble putting more than three ATA66 controllers in one system. This combined with the relatively low price of IDE RAID controllers and the performance problems using a master+slave configuration in a RAID led us to take another route. Our original terabyte systems used the 3ware Escalade 6800 IDE RAID controller, while our newer systems use the Escalade 7500 cards. The Linux kernel driver for this card is quite mature, and we've had very impressive performance from our system. With the new 160 GB drives now shipping, we can attach up to 1.1 TB of capacity per card for a very reasonable price. 3ware's documentation says they have tested three cards in a single machine and that there is no reason you shouldn't be able to use more, though using more than 2-3 cards would make for a very busy PCI bus. When choosing to put multiple controllers in your system, don't forget about networking – if you are using your system as a file server, every byte you read from the drives also needs to go out over a network interface. You might be able to address the bus contention problem by purchasing a system with many PCI busses if you really need a fast 4-terabyte file server in a single Grid Brick, though a better approach might be to wait for the new 320GB drives. Combining these drives with the new 12 port 3ware controllers will allow you to build extremely dense storage solutions suitable for many applications.

Early in the development stages, we decided to go with an Athlon-based system instead of an Intel-based one. This reduced our costs for motherboards and CPU, but it limited the amount of RAM we could put in the machine since at the time most Athlon boards had only 3 slots, limiting us to either 768 MB using 256 MB DIMMs or 1.5 GB using 512 MB DIMMs. Since we did this initial work, pricing and hardware availability have changed and we're now building systems based on the current generation Intel Celeron. We chose the Celeron over other options due to its low cost, low power requirements, and low heat output. Using a system board that supports the 1.7ghz Celeron also allows easy substitution of a faster Pentium 4 CPU without any other changes to the system design. This could be very useful for nodes that are doing data processing or aggregation and have additional processor requirements.

Selection of a motherboard was a somewhat more complicated process. Our original test system was built using the Athlon-based Asus A7V. When we migrated our second generation systems to the older "socket 370" Celeron, we used the Asus TUSI-M board. This board worked well with the older Escalade 6800 controllers, but it doesn't work at all with the newer 7810 or 7500 cards. After a great deal of testing and experimentation, we settled on the Gigabyte GA-8IRXP board. This board features integrated LAN, a fast bus, and support for Intel's newer Celeron CPU. However, selection of a system board and CPU may be dictated by the needs of your particular application. For help in choosing a system board suited to your application, 3ware publishes a list of boards that are certified to work with their controller cards. With some of the next generation of system boards now featuring integrated gigabit ethernet, we may choose to revisit this problem again to reduce cost and complexity.

Also note that for applications where raw performance is not a key concern, the VIA C3 processor can present an appealing alternative – with a peak power consumption of only 8.5w, this processor allows you to build extremely dense systems while greatly simplifying power and heat capacity planning.

Though our test systems were installed in standard Antec SX-1030B desktop cases, our production systems were integrated into a 4U rack mount. With the advent of 320 GB drives, it will be possible to create a Grid Brick with a capacity of 1.1 TB that can be packaged into a 1U rack mount. In standard racks, this implies the ability to manage 40 TB of data aggregated across 40 Grid Bricks in a single rack.

One of the benefits of the integrated storage approach that we used in the grid brick is that the systems require little software customization and are easy to integrate into existing management infrastructure. At SDSC, we manage hundreds of UNIX and Windows machines using management tools like GNU cfengine and Microsoft's Systems Management Server. For management purposes, a grid brick is simply a standard system that happens to have a terabyte of local disk instead of the usual 20-40 gigabytes. The 3ware controller does require a driver and software, but the driver is included with many Linux distributions and the software is simple to install. In our case, it was simply a matter of telling cfengine "these systems need the 3ware software installed". For a site doing their own system installation or managing a small number of systems manually, you simply install Linux (Redhat recognizes the 3ware controller automatically) and install the monitoring software from 3ware's web site. While our reference bricks run Linux, there is no reason that you couldn't build a similar system running Windows or one of the BSD Unix variants. The only requirements are support for the 3ware controller card and the ability to handle very large block devices.

There are only two real management caveats we've encountered. One of these is a bug: the "grub" bootloader included with current versions of Redhat Linux is not capable of booting from a block device larger than 1 terabyte. In the short term, this problem can be fixed by using the "LILO" bootloader (also included with Redhat) instead. In the long term, we hope RedHat will address this. The other problem is more complicated: we understand how to back up a few hundred machines that have a few gigabytes of data on each system. Backing up hundreds of machines with multiple terabytes of data is a much bigger problem, especially when you're trying to keep costs reasonable. For environments where it's not possible to have data redundancy at the application level, the simplest solution is to build upon the capabilities of data grids, and replicate the data onto an archive or onto a second grid brick. One can achieve high availability by using replication within the data grid, and using the ability of the data grid to fail over to a replica when the initial storage resource is unavailable. Data grids make it possible to improve reliability by using the same access mechanisms for both the original copy of the data and the “back-up” copy.

6. Administration and Usage of Bricks

The use of the SRB as the data management system over the Grid Bricks makes the system administration of the bricks very simple. In fact, when the brick was installed, apart from the root user ID, only one non-root user ID was created, called bricksrb. The whole 1 TB user file area is managed under the control of the user ID bricksrb. The SRB is installed as an RPM and any patches for the software are implemented in pre-specified Preventative Maintenance schedules. Hence, with no users to manage and the file resource under the control of SRB, system management becomes very simple and many distributed Grid Bricks can be managed by a single system administrator. The SRB monitoring system is used to check the usage of the file resource in the Grid Brick and if it reaches a high water mark, either the resource can be write-locked (disallowing any new files being created on it) or files can be automatically moved off to near-line storage (tape systems [Wan et.al. 2003]) using the caching and container facility of SRB.

The Grid Bricks went through rigorous testing before deployment. First, the disks in the brick are tested for more than two days with a set of rigorous writing/reading programs. This is to weed out any disks that may have manufacturing defects or early mortality. Once these tests are done, we can be somewhat safely assured of a long MTBF. After this testing, the SRB server is started on the Grid Brick and another round of rigorous testing, lasting a week, is performed on the Bricks. In this test, parallel streams of files are read and written from the disks at various degrees of parallelism. Also, as part of the tests, varying file sizes are also used from a few KiloByte files to file sizes of over 50 GigaBytes. This again tests the scalability of the SRB. The Grid Brick performance should be compared to some baseline characteristics (based on the experience of the first brick) before they are ready for actual deployment. During the regular use of the Grid Bricks, test routines are also regularly run to check the performance of the Grid Bricks. These tests are performed to see if any new bottlenecks have arisen. These tests are made to find slowdown in performance due to disk failures, SRB optimizations, network loading and overall system health

Currently, the Grid Bricks are being used as part of several on-going projects at SDSC. One of the projects is to deploy the collection of visualization artifacts (images, movies, 3d simulations, etc) that have been created by the Visualization Group at SDSC. The SRB will be used to manage this collection, along with its rich support for metadata. Metadata describing the physical characteristics of the artifact as well as the themes of the artifact, along with annotations and commentary will be used to provide a very enjoyable browsing and searching experience for the user.

In another project, a Grid Brick is being used to store and serve digital education material for the National Science Digital Library (NSDL) project. In this project, web sources are crawled, based on their contents useful for education curriculum modules. Copies (snapshots) of these web pages and supporting web pages are stored in the SRB. Currently, this project has generated nearly 100 GB of storage and is expected to grow to nearly one TeraByte in the near future. An interesting aspect of the project is its use of XML-based metadata. The metadata is gathered by another collaborator on the project and is converted into XML files. The SRB is used to interpret these XML files, extract the metadata and associate them with the web pages stored in the SRB. The aim of the project is to archive snapshots of web pages that form part of a science curriculum. By archiving the snapshots and making them available, a teacher can retain access to material that in effect remains unchanged for the duration of the course and beyond.

A third project, getting underway, is with the National Archives and Records Agency (NARA). In this project, a Grid Brick system deployed at SDSC will be integrated with storage resources at NARA and the University of Maryland. The ability to integrate Grid Bricks into a data grid will be used to provide a distributed persistent archive for the NARA agency.

7. Summary

Data sharing environments can be built that are based on logical name spaces to provide global names, grid bricks for data picking environments, SANs for data streaming environments, and persistent archives for long term storage. Data grid technology provides the essential infrastructure for federating the systems. Grid Bricks provide an attractive cost effective storage environment for persistent access to large collections to support the picking of individual digital entities.

8. Acknowledgements

This research has been sponsored by the Data Intensive Computing thrust area of the National Science Foundation project ASC 96-19020 “National Partnership for Advanced Computational Infrastructure,” the NSF National Science, Technology, Engineering, and Mathematics Education Digital Library, the NARA supplement to the NSF NPACI program, the NSF National Virtual Observatory, and the DOE ASCI Data Visualization Corridor. We also acknowledge Peter Ashford who helped during the design and building of the first TeraByte brick.

9. References

[1] 2MASS, http://www.ipac.caltech.edu/2mass/.
[2]Baru, C., R, Moore, A. Rajasekar, M. Wan, (1998) “The SDSC Storage Resource Broker,” Proc. CASCON'98 Conference, Nov.30-Dec.3, 1998, Toronto, Canada.
[3]BIRN, “Biomedical Informatics Research Network”, (http://www.nbirn.net/).
[4]EarthScope, (2001) “EarthScope”, (http://www.earthscope.org/).
[5]Foster, Kesselman, C., “The Grid: Blueprint for a New Computing Infrastructure,” Morgan Kaufmann, San Francisco, 1999.
[6]GriPhyN, (2000) “The Grid Physics Network”, (http://www.griphyn.org/proj-desc1.0.html).
[7]Stockinger, H., O. Rana, R. Moore, A. Merzky, “Data Management for Grid Environments,” European High Performance Computing and Networks Conference, Amsterdam, Holland, June, 2001.
[8]Devlin, B., J. Gray, B. Laing, G. Spix, “Improve Your Scalability Vocabulary “, (http://www.clustercomputing.org/ARTICLES/tfcc-4-1-gray.html), 2002.
[9]Hammond, S., (1999). "Prototyping an Earth System Grid", at the *Workshop on Advanced*

Networking Infrastructure Needs in Atmospheric and Related Sciences, National Center for Atmospheric Research, Boulder CO, 03 June 1999. (http://www.scd.ucar.edu/css/esg/presentations/nlanr/index.htm).

[10]Hoschek, W., Jaen-Martinez, J., Samar, A., Stockinger, H., and Stockinger, K. (2000) "Data Management in an International Data Grid Project," *IEEE/ACM International Workshop on Grid Computing Grid'2000*, Bangalore, India 17-20 December 2000. (http://www.eu-datagrid.org/grid/papers/data_mgt_grid2000.pdf).

[11]KNB, (1999) "The Knowledge Network for Biocomplexity", (http://knb.ecoinformatics.org/).

[12]MCAT, (2000) "MCAT: Metadata Catalog", SDSC (http://www.npaci.edu/dice/srb/mcat.html).

[13]NEES, (2000) "Network for Earthquake Engineering Simulation", (http://www.eng.nsf.gov/nees/).

[14]NVO, (2001) "National Virtual Observatory", (http://www.srl.caltech.edu/nvo/).

[15]PPDG, (1999) "The Particle Physics Data Grid", (http://www.ppdg.net/, http://www.cacr.caltech.edu/ppdg/).

[16]Moore, R., "Knowledge-based Grids," Proceedings of the 18th IEEE Symposium on Mass Storage Systems and Ninth Goddard Conference on Mass Storage Systems and Technologies, San Diego, April 2001.

[17]Moore, R., "Preservation of Data, Information, and Knowledge," Proceedings of the World Library Summit, Singapore, April 2002.

[18]Moore, R., and A. Rajasekar, (2001) "Data and Metadata Collections for Scientific Applications", High Performance Computing and Networking (HPCN 2001), Amsterdam, NL, June 2001.

[19]Moore, R., C. Baru, P. Bourne, M. Ellisman, S. Karin, A. Rajasekar, S. Young, "Information Based Computing," Proceedings of the Workshop on Research Directions for the Next Generation Internet, May, 1997.

[20]Rajasekar, A., and M. Wan, (2002), "SRB & SRBRack - Components of a Virtual Data Grid Architecture**",** *Advanced Simulation Technologies Conference (ASTC02)* San Diego, April 15-17, 2002.

[21]Rajasekar, A., M. Wan, and R. Moore, (2002), "MySRB & SRB - Components of a Data Grid," *The 11th International Symposium on High Performance Distributed Computing (HPDC-11)* Edinburgh, Scotland, July 24-26, 2002.

[22]Rajasekar,A., R. Moore, "Data and Metadata Collections for Scientific Applications", High Performance Computing and Networking (HPCN 2001), Amsterdam, Holland, June 2001.

[23]SRB, (2001) "Storage Resource Broker, Version 1.1.8", SDSC (http://www.npaci.edu/dice/srb).

[24]Wan, M., A. Rajasekar, P. Andrews, R. Moore, "A Simple Mass Storage System for the SRB Data Grid", Proceedings of the 20th IEEE Symposium on Mass Storage Systems and Eleventh Goddard Conference on Mass Storage Systems and Technologies, San Diego, April 2003.