ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

ANL/MCS-TM-235

Implementation of a Distributed Multiterabyte Storage System

by

Mark Hereld, Bill Nickless, and Rick Stevens

Mathematics and Computer Science Division

Technical Memorandum No. 235

October 1998

This work was supported 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.

Implementation of a Distributed Multiterabyte Storage System

by

Mark Hereld, Bill Nickless, and Rick Stevens

Abstract: The University of Chicago and Argonne National Laboratory are building a highly distributed storage system to meet the needs of a cross-disciplinary group of scientists and to investigate the issues involved in implementing such a system. The storage system is based on IBM 3590/3494 tape technology managed by ADSM software. High-speed wide-area networking is required to support the distributed user community. NFS, DFS, FTP, and other user access methods are supported. A simulation project undertaken to guide system development has provided useful insights even in the calibration and design phases.

1. Introduction

The University of Chicago/Argonne Terabyte Storage (UCATS) project has two primary goals:

- **Deploy a distributed HSM system** to support a number of diverse applications, providing a unified data repository [1] spanning resources at the University of Chicago and Argonne National Laboratory.
- **Provide a testbed for experimenting** with distributed mass storage systems and distributed filesystem software—all integrated into workstation, data collection, visualization, and supercomputing environments.

Subsidiary goals and criteria for success of the project include the following:

- **Support a heterogeneous user environment** that includes storage requirements ranging from backing up laptops through checkpoints of massively parallel processor supercomputers.
- **Provide user access transparency** such that the distributed nature of the system is not visible to the user. Ideally, access semantics and performance should be indistinguishable from or at least as good as a single-site system.
- **Make the distributed nature of the system a benefit to the user.** When possible the system should be optimized to perform better because of its distributed nature. Possible benefits include fault tolerance, convenience, and higher performance.
- **Minimize impact on the network.** Even though we have a high-performance network, we wish to design the system to minimize excess data motion through the wide-area network (WAN).

- **Maintain data reliability and integrity.** User confidence in the repository has to come first, and the distributed nature of the system should not reduce reliability or integrity.
- **Optimize performance** for both read- and write-intensive uses of the system. We expect to use a custom-written simulator, as well as empirical observation and experimentation, to determine optimal disk caching policies, buffer sizes, number and organization of filesystems and tape units.

To meet these goals, we are deploying a distributed storage system based on IBM's ADSTAR Distributed Storage Manager (ADSM) [2]. Based on our experiences with the I-WAY project [3,4,5], we decided to use ATM [6] network technology for the interconnections between components distributed across the Chicago metropolitan area. Argonne National Laboratory and the University of Chicago participate in the Metropolitan Research and Education Network (MREN) [7], which is an ATM network run at OC-3 (155 megabit/second) speeds by Ameritech. We use the Distributed File System (DFS) [8] to provide a unified file name space, security model, and local cache performance for end users.

2. UCATS Science User Applications

Datasets in many scientific disciplines are pressing into terabyte territory. Such large datasets will allow researchers to solve new kinds of problems but only if we develop the technologies necessary for efficiently managing, organizing, and examining these data troves. The diversity of databases, examination modes, and throughput needs makes the problem of optimizing a data storage and delivery system very difficult.

The UCATS project was founded on the needs of a handful of science projects whose research areas include theoretical and experimental astrophysics, climate studies, structural biology, and cosmic ray physics. We expect that the UCATS project will grow to serve an increasing number of diverse and demanding research areas.

Our collaborators' datasets have achieved behemoth proportion in different ways. Some experiments generate one huge dataset by performing continuous measurements of a system. Other endeavors serve a large community of experimenters with individually modest datasets.

2.1 CASA

Using the largest and most sensitive high-energy telescope yet built, University of Chicago physicists are studying air shower events that are produced by high-energy cosmic rays. The 500 meter by 500 meter Chicago Air Shower Array (CASA) is indirectly sensitive to primary cosmic rays with energies above 100 TeV.

The experiment has already produced nearly a terabyte of data, with 7 TB expected in the near term. These data will be used to study the spectrum of cosmic rays and the distribution of gamma rays from the plane of the galaxy.

In the past, such experiments have created a large cache of data on magnetic tapes, which are passed once through reduction filters that yield a selected subset of the data. The resulting compressed dataset is more manageable than the full raw dataset, but limits the questions that can be answered. Access to the entire nonfiltered dataset allows for studies requiring the greatest overall sensitivity to, for example, rare events at the highest energies detectable by CASA.

The data are collected as time-tagged events characteristically requiring 1.4 kB per event at a continuous rate of 20 events/sec. Over 2 GB/day and nearly 1 TB/year are produced. These will be accessed from several SGI workstations located at the University of Chicago. A single researcher, for example, might

pass through the entire dataset in a more or less sequential fashion with custombuilt software that accesses the stored data through an interface based on the ADSM API. Several researchers might be accessing the data simultaneously in this way and at different points in the dataset.

2.2 CARS

The Consortium for Advanced Radiation Sources (CARS) is a collection of more than 140 scientists from 40 institutions nationwide who will be using the Advanced Photon Source (APS) at Argonne to study a large and diverse set of experiments in structural biology, geophysics, chemistry, and materials science. The APS will supply this diverse scientific community with a brilliant source of X-rays for images and time-resolved studies of biological molecules, geophysically relevant minerals, and new materials.

The data from all of these areas are most typically in the form of two-dimensional images about 2 MB in size. These are grouped in batches that represent either time sequences or repeated images of the system under study. For example, the data from a single researcher's experiment might require between 2 and 100 GB and take a couple of days to collect.

In this application, the administrators of the APS experimental stations write the data to the storage system. The individual researchers will have read-only access to directory partitions in their name. With many otherwise unrelated users, security becomes an important issue. Users in this group may typically transfer their experimental data only a few times, in pieces, to their local workstations and analyze it locally. This is practical over ATM for even the largest datasets. Their data will remain in the archive for about 6 months as a primary backup source. Most of the storage required for this application will reside in the Argonne library, while a subset may reside in the University of Chicago library.

2.3 Astrophysical Magnetohydrodynamics

Astrophysicists at the University of Chicago are studying turbulence in a variety of settings. Among these are stellar and planetary interiors, explosive events such as novae and supernovae, and accretion disks. The numerical experiments require direct solution of nonlinear differential equations on the fastest computers available.

The results of these numerical experiments are stored as a time-ordered sequence of large three-dimensional data cubes containing lists of the interesting physical quantities (such as energy and momentum flux). Typical problems might generate from 10 to 20 separate runs, each representing a time sequence requiring around 50 GB.

The simulations are run at Argonne on the IBM SP parallel machines. Research into efficient parallel I/O techniques is integral to this work. Once created, the data will be available for analysis, in various visualization modes, including the CAVE at Argonne and an ImmersaDesk and SGI workstations at the University of Chicago.

3. Implementation and Research Issues

3.1 Heterogeneous User Environment Support

Because the UCATS project is both a real applications environment and a research testbed, we need to support a diversity of end-user clients. We also are providing filesystems-based access, ftp access, and direct access to the ADSM server via client API interfaces. We have developed a simplified user access library, called COIL, that enables users to access the most commonly needed features of the client API. Heterogeneity demands filesystem access to different client systems via their supported interface. The API interface access is limited to storage managed by ADSM for that type of access (i.e., a client using the API

cannot directly access data stored on the HSM, which is allocated for backing up filesystem clients). This limitation improves reliability by preventing users from corrupting data backing filesystems, but it reduces flexibility. One can imagine applications that would want to have both filesystem and direct access to data on tape. Several applications plan to use the ADSM to back database applications.

COIL is motivated by three factors revolving around distributed storage: the nonuniform nature of file availability and access speed, the desire of application developers to exert some control over filesystem behavior, and the fact that some useful storage systems are simply not available to users at the traditional UNIX system call level. Some filesystems are not integrated with the operating system. For example, the HTTP file transfer protocol is not available to the standard UNIX open, read, write, and lseek calls. Unitree does provide an NFS view into its file space, but the performance of the NFS interface is very poor compared with alternatives that Unitree provides. ADSM can provide programmatic access to a file space, but the ADSM API is not integrated into the UNIX kernel.

3.2 Access Transparency

The distributed nature of the HSM should not be visible to the user unless the user desires knowledge of data locality for security or other reasons. Ideally, the user should not be aware of the fact that the HSM system may be composed of servers that are physically distributed over a wide-area network. We have designed the system to operate within a single high-level DNS namespace. Thus, at the highest level the user is not aware of the distributed nature of the system. Access semantics via the filesystems interfaces (NFS, DFS) remain the same regardless of which HSM Front End is used. Part of the unified namespace is also a requirement for a unified authentication environment for the storage system. NSF access to the system requires that the UNIX authentication namespace UIDs be made consistent across all servers and clients, something difficult to accomplish within an environment with many administrative domains (UCATS spans at least 7 administrative domains).

DFS provides a unified, cross-domain namespace. This enables a user to access files by the same UNIX path regardless of the system being used, enabling application and user portability. Unfortunately, since it is a global namespace, it can impose certain requirements on a pre-existing UNIX administrative domain. Also, a unified namespace can obscure the underlying infrastructure of the storage system, leading to mismatches between user expectations and achieved performance.

A primary determinant of user-perceived transparency is the potential difference in performance of local vs. remote accesses to data. The UCATS system is designed to minimize potential performance differences by

- providing large HSM front end servers at each site with considerable disk caching;
- interconnecting the clients workstations, front-end, and back-end servers via uniform high-speed networking (i.e., ATM OC-3); and
- encouraging users to access the HSM system via local DFS filesystem clients with a large amount of local disk. We expect to show that this two-level file caching scheme will improve client performance considerably.

We have begun a series of experiments and simulations to determine the effects of WAN bandwidth on typical applications performance. An early experiment will be to measure the access times for remote vs. local HSM front ends. Extensive cache and buffer disk in both the HSM front end systems and the client workstations support read caching and write buffering.

While transparency is generally considered a good feature of the system, occasionally a user may wish to ensure that his/her data resides at a particular site. For example, the user may wish to physically remove the media or avoid causing considerable WAN networking traffic (even if the performance would be adequate). Providing a logical to physical mapping of the filesystems and system names, as well as the

ability to force data to reside on specific logical volumes, is required. Also required is the ability to flush caches and buffers.

We may find that the user information and controls provided by standard DFS are not at a high enough abstraction level to work in the context of HSM activity. For example, DFS may not expose to the user whether a given file is available for immediate use on disk instead of requiring a tape mount. DFS may not provide the user with information about the size of the caches involved so that the user could make intelligent choices to avoid thrashing. These and other issues will be reported to the ADSM and DFS development communities.

3.3 Benefits of Distribution

The primary benefits of a distributed HSM are uniform accessibility and higher performance data access. Both of these are critical when a user is generating data at one site and analyzing the same data at another site and is freed from explicitly moving data to achieve adequate performance for both tasks. In the UCATS project many users are accessing or generating data from both their workstation or ImmersaDesk at the University of Chicago and Argonne's IBM SP or CAVE environments.

Secondary benefits of distribution include the ability to replicate data at multiple sites for improved performance or reliability and to provide redundancy of servers, allowing (in principle) one set of servers to provide recovery capabilities for their remote siblings. By administering the system as a single logical HSM, we also achieve commonality and scalability of systems support, user administration, and tool development. With appropriate interfaces and SSA-via-ATM tunneling [9,10] it may be possible, for example, to support twin-tailed disk interfaces on the ADSM servers via a WAN to provide distributed fail-over capabilities.

Future development work is aimed at providing the ability to prefetch and permanently migrate large-volumes of data from one server to another to reduce the real-time load on the WAN network and to support load balancing the robots and backend servers. We believe that significant benefits of distributed mass storage and HSMs will become apparent after the number of sites has been expanded to greater than two (e.g., to all the sites in an NSF PACI collaboration or to all the DOE laboratories).

3.4 Reliability and Integrity

ADSM is a popular commercial-quality hierarchical mass storage system. When deployed in typical configurations, it is highly reliable. The primary factor in potential reduction of reliability is the reliance on the wide-area network for critical functionality. The HSM front ends can continue to operate without constant contact with the ADSM servers, although staging and migration would wait for the contact to be restored. DFS supports redundant servers for read-only data and provides client caching for read-write data.

We are building on the reliability and integrity by choosing proven, commercial, off-the-shelf system components. The research and development activities revolve primarily around configuration and integration of those components. Our efforts will include tuning the components individually and in concert with each other. We will report our experiences to the development communities of ADSM, DFS, and other components in order that the future design of these components can be even more useful in configurations and installations such as this.

4. System Description

4.1 Hardware

The IBM 3590 (Magstar) tape drive uses ½" cartridge tapes in the form factor of the traditional IBM 3480 tape cartridges. Each 3590 tape cartridge holds 10 GB of uncompressed data. The 3590 tape drive provides transparent hardware tape compression and operates at a read/write rate of up to 14 MB/sec. Initial experimentation indicates that the maximum seek time of a 3590 drive is approximately one minute to anywhere on the tape, with the average around 30 seconds.

The IBM 3494 tape library is a robotic tape-changing system designed to work with tapes compatible with the form factor of traditional IBM 3480 cartridges. Initial experimentation indicates that the time required to select, mount, and cue a 3590 cartridge is approximately 25 seconds.

Currently we have two tape libraries, each with one 3590 tape drive and capacity for 600 3590 tapes. One tape library will be located at the University of Chicago to support the local users there; the other library will remain at Argonne National Laboratory to support the University of Chicago users of the Advanced Photon Source.

4.2 System Software

The initial requirements for the library and tape management software were availability, ease of configuration, and support for the devices. ADSM is a mature product in use in many sites across the globe, lending confidence that it is reliable and robust.

ADSM does not support parallel tape reads and writes [11,12], a feature that could be useful for some application areas. Sequential tape performance, through ADSM, is also limited because of buffer sizes and other constraints. By recognizing these bottlenecks and performing simulations of various configurations that include ADSM and front-end systems, we believe we can meet the needs of the applications while reaping the benefits of the maturity and stability of ADSM.

4.3 Logical Configuration

UCATS is configured in a client/server architecture. There are three types of servers and three types of clients, as shown in Fig. 1. Some servers act as clients; this is generally because the various types of servers provide different services, and some servers need the services of other servers.

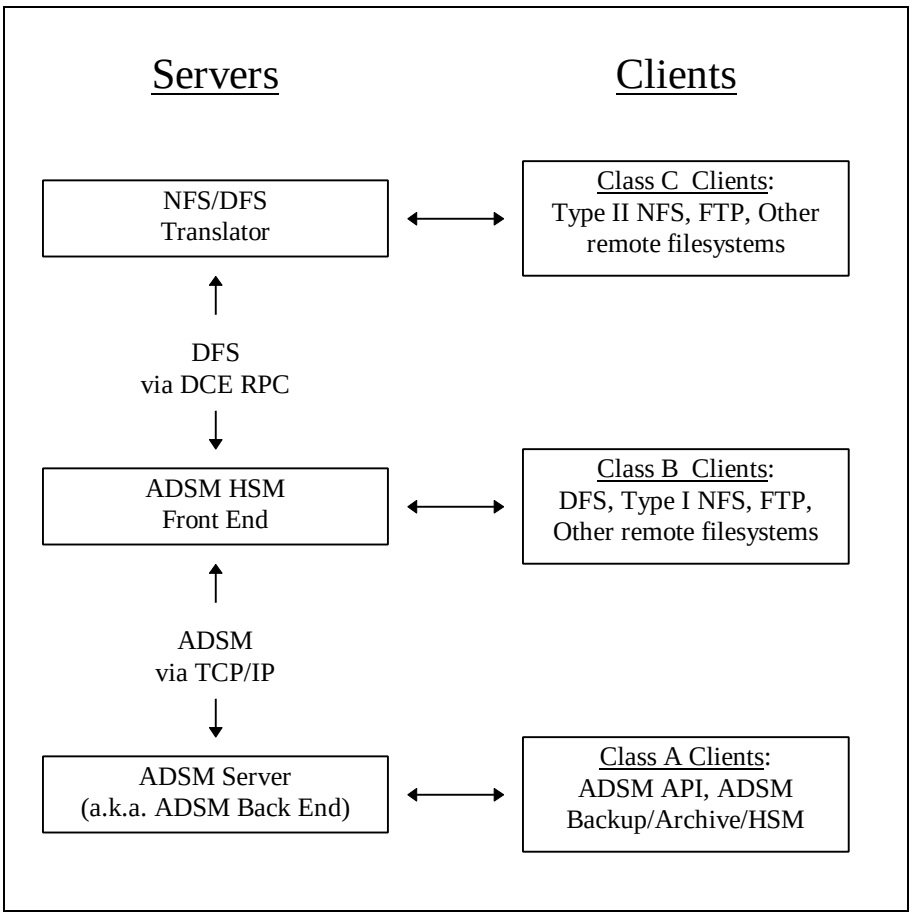

Figure 1: *Logical Client/Server Relationships*

The **ADSM server** runs the IBM ADSM software to manage the tape drives and robots.

The **HSM front end** runs the IBM ADSM Hierarchical Storage Management software against a UNIX filesystem. The HSM front end uses the services of one or more ADSM Servers. Each HSM Front End has its own UNIX filesystems, which store the data that have not been migrated to an ADSM Server. These UNIX filesystems also manage all of the filesystem metadata. Filesystem metadata are not used or managed by the ADSM server. These UNIX filesystems can be used directly by processes running on the HSM front end or can be exported to other client systems by various network protocols, including NFS, DFS, and FTP.

An **NFS/DFS translator** is used to provide NFS access to DFS filesystems. DFS is used as the wide-area filesystem. It provides a unified name space, client caching, and cryptographic security. Since not all client systems will have DFS client capability, we provide an NFS/DFS Translator to allow client systems logically nearby to benefit from the unified name space and other features of DFS.

Figure 2 shows how two clients, one at the University of Chicago and one at Argonne National Laboratory, might use UCATS through the three classes of access.

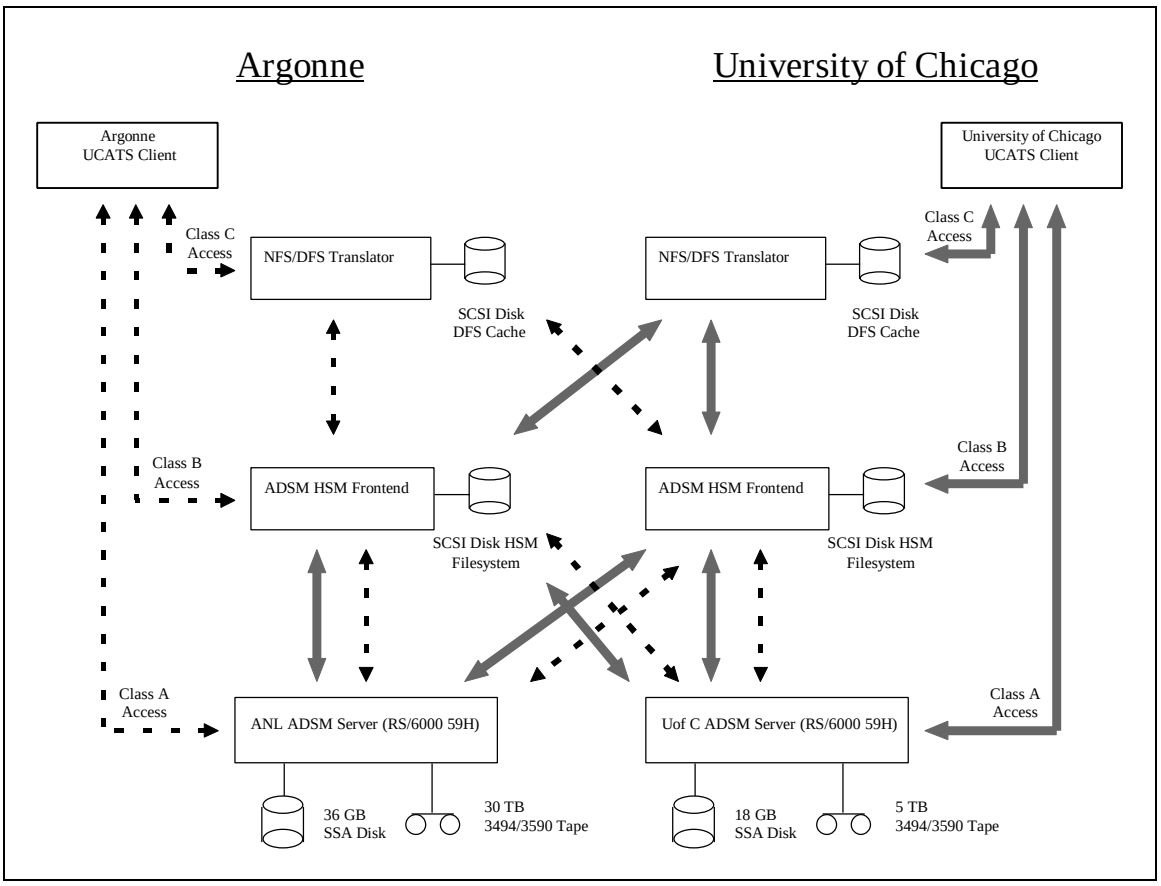

***Figure 2**: Example client interactions and data flows*

4.4 Client Access Methods

There are three classes of clients, depending on the type of server being accessed.

Class A clients run the IBM-supplied ADSM Backup/Restore software, link against the ADSM application programming interface, or run the ADSM Hierarchical Storage Management software against a local filesystem. Each Class A client has its own independent relationship with one or more ADSM Servers. Except in special cases, Class A clients do not directly exchange data with other Class A clients. An HSM Front End is a special type of Class A client.

Class B clients are serviced by HSM front ends. This service is typically in the form of some network protocol such as FTP, NFS, or DFS. To use FTP, the client user must have a user account on the HSM front end. To use NFS, the client system must be in the same administrative domain and be on a local network with the HSM Front End. Any Class B client using DFS can access any HSM front end, regardless of whether the DFS client and HSM front end are logically "close" to each other on the network.

Class C clients are serviced by NFS/DFS translators. These are systems that do not have the capability to access DFS filesystems directly, but need access to the unified name space and distributed data services that the DFS paradigm supplies. NFS/DFS translators act as DFS-protocol Class B clients.

4.5 Detailed Configuration and Environment

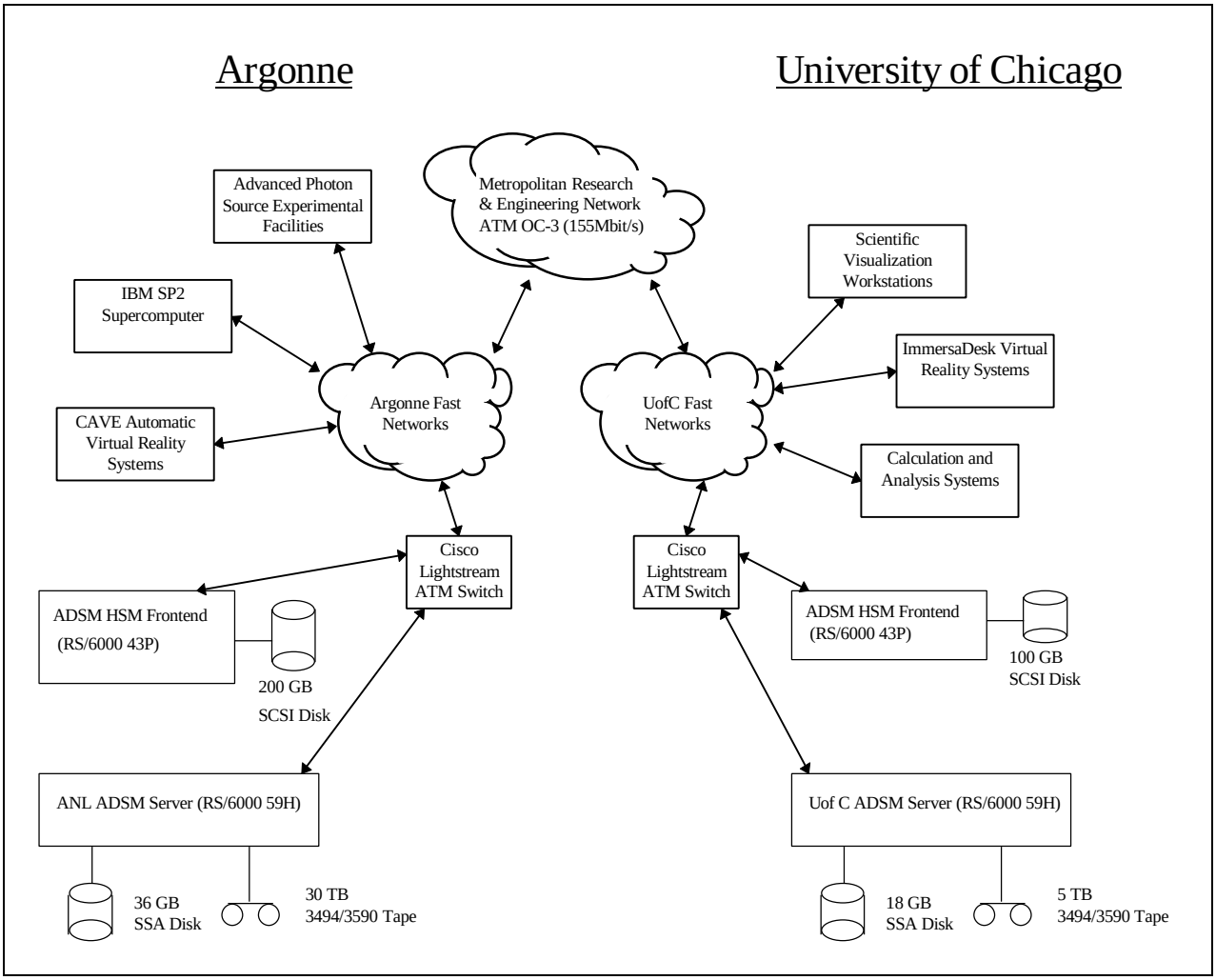

Figure 3: *UCATS Configuration Detail and Environment*

5. Simulation Project

ADSM has a variety of parameters that can be tuned to improve performance. In addition, basic throughput analysis can be used to help size the disk caches and buffers in the system. We are developing an HSM simulator [13,14,15,16] that will enable us to explore the impact of cache and buffer sizes, number of servers and placement of servers on a variety of job mixes.

Working with the simulator is helping us to understand key issues of system behavior, such as workload. Our initial simulations are small; the strategy is to gradually simulate larger, more complex representations of the UCATS system.

5.1 Design and Implementation

We have designed our simulator with special emphasis on flexibility because it is intended as a tool to help us explore implementation issues for the present storage system and for more complex distributed systems. We are writing it in an object-oriented language (C++), hoping to enjoy the benefits of ease of expansion and change and a clean encapsulation of basic functions and relationships. This approach makes for a concrete mapping of system components into simulated component objects: it is a straightforward matter to simulate different architectures.

The two main aspects of the simulation -- the workload model and the system model -- are handled separately to facilitate flexibility in design and testing. The system model includes objects for disks, tape drives, HSM front ends, ADSM server, tape robot, data caches, and data chunks. The workload is simulated by objects for clients of various types (see below). Communication between the two parts of the simulation is effected by objects representing jobs and job queues. Clients generate jobs and submit them to queues. These in turn drive the simulated system hardware.

The simulation is managed by scanning activities at fixed time intervals. The list of objects needing attention is scanned randomly at each time step. The overhead of scanning during intervals when nothing is going on is very small, and so the approach is quite efficient. On the other hand, many activities in the simulation are amenable to an event-driven approach: we compute, for example, when a given disk operation will be complete. The simulation might ultimately benefit from a hybrid approach wherein events are used to determine the size of variable time steps.

To calibrate various parameters within the components of the system model, we perform experiments with the hardware. These yield seek times, tape load time, transfer rates, and other aggregate parameters.

We have tested the system simulator at various stages of its development against a simple stochastic workload composed of Class A and Class B clients. With these experiments we have verified basic behaviors of the system: contention for disk and tape drives, data staging and migration, and cache function.

5.2 Workload

It is much harder to arrive at a model for handling the vagaries of the workload. This is not our problem alone: in simulations of real systems this is generally true. The chief sources of our difficulty are that (1) we have little a priori knowledge about the individual client application characteristics, and (2) the targeted diverse clientele guarantees complicated descriptions of job mix.

The initial stochastic model that we used to verify the basic operation of the system simulation proved to be too simple.

Our experience with the system and with the science clients has led to a compact and practical method for modeling the workload. We intend to model several workload archetypes or atoms that represent attributes of a dataset and use pattern.

Our working set of these atomic workloads is characterized as follows:

Large, flat, high-throughput	High-end visualization applications driven by large undifferentiated datasets: e.g., Visible Human data
Large, flat, small-transfer	Low-end random access of small chunks of data (e.g., second-grade children accessing a digital library through a browser)
Large, with good metadata	High-end visualization, or query language, characterized by high proportion of hits on a well-defined metadata (e.g., astronomy database)
Backup and archive	Throughput is important, but often can be done during non-peak time

Basing our workload model on atoms helps in guiding measurements of real workloads, iteratively adjusting the workload atomic model parameters, and systematically calibrating realistic workload mixes.

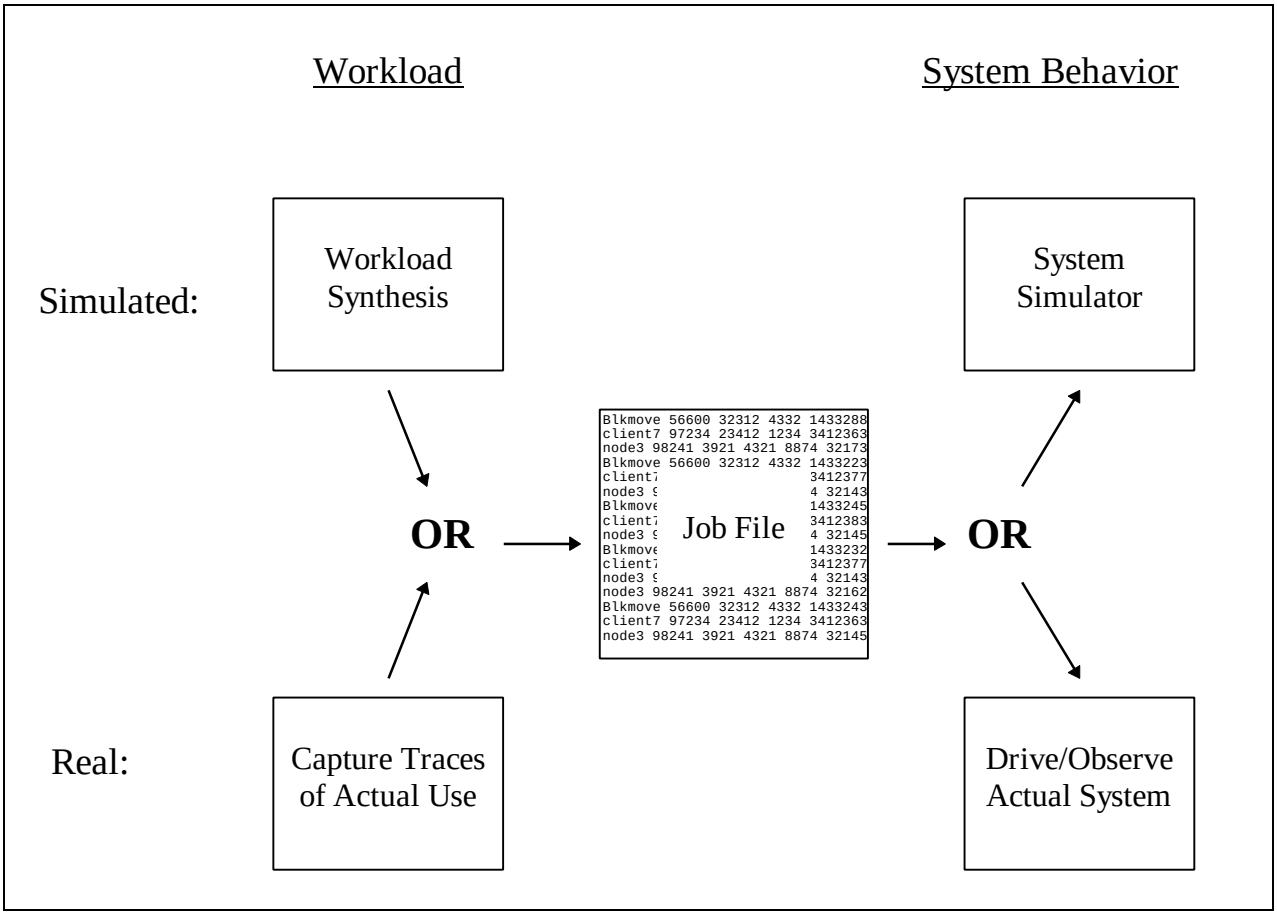

***Figure 4**: Standard Job File for Simulation and Evaluation*

Figure 4 shows the relationship between the workload synthesizer and the system simulator. These two simulator portions will communicate through an intermediate file consisting of the sequence of job requests. With this architecture, we can easily perform a number of useful experiments, that will help us to calibrate the various pieces of the simulator. By driving both the system and the system simulator with

the same job file, we can directly compare simulator with real-world performance—an important technique for validating the simulator. This division will also allow direct comparison of the workload simulator with job files generated by monitoring the real system, invaluable in calibrating our workload models.

6. Future Directions and Conclusions

We are investigating three areas that should improve performance and/or minimize WAN traffic. First, we are developing simple mechanisms to support third-party transfers when the client and HSM front end servers are on opposite sides of a WAN connection. Second, we are developing software to track user access patterns in the data to enable data migration decisions. For example if a user repeatedly is accessing data stored on an remote ADSM server, resulting in repeated transfers due to local cache capacity limits, we mark these data as a candidate for automigration to the "local" ADSM backend so that future requests will not result in WAN traffic. Third, we are experimenting with migrating the logical HSM front end from one site to another. This strategy can reduce networking traffic when a large data set read from the ADSM backend is being cached to the HSM front end but when the user access pattern is sparse; thus, by moving the assigned HSM front end from local to remote, we reduce the networking traffic to just the data being requested by the user, rather than the full dataset. Currently ADSM does not support many of these capabilities. We are investigating extensions to ASDM servers to support this advanced functionality.

With this system we have an opportunity to provide exceptional service to our clients and to investigate the behavior and performance of a distributed storage system built from off-the-shelf technology. By providing service to real users, we will validate (or be forced to re-evaluate!) our expectations for user access patterns. We will be able to evaluate the current distributed storage technology in a real-world situation and have actual experiences to share with the various development communities, along with other groups building distributed storage systems.

References

[1] Grossman, R. L., et al., " A Prototype of a Digital Library for Numerical, Statistical, and Scientific Data and Application to the Federal Budget," Proc. Digital Libraries 1994: Conf. Theory and Practice of Digital Libraries, ACM, 1994.

[2] IBM Corporation. ADSM – Product Overview.
http://www.storage.ibm.com/stssorage/software/adsm/adprov.htm

[3] Foster, I., Geisler, J., Nickless, W., Smith, W., and Tuecke, S., "Software Infrastructure for the I-WAY High-Performance Distributed Computing Experiment," in Proc. 5th IEEE Symp. On High Performance Distributed Computing., IEEE Comptuer Society Press, 1996, pp. 562-571.

[4] DeFanti, T., Brown, M., and Stevens, R., " Virtual Reality over High-Speed Networks," in IEEE Computer Graphics and Applications, Volume 16, Number 4, 1996, pp. 42-43.

[5] DeFanti, T., Papka, M., and Stevens, R., eds., "I-WAY: Wide Area Supercomputing Applications,'' special issue of The International Journal of Supercomputer Applications and High Performance Computing, Volume 10, Number 2/3, Summer/Fall 1996.

[6] The ATM Forum, Educational Module: ATM Basics.
http://www.atmforum.com/atmforum/atm_basics/notes1.html

[7] Academic Computing Services, Networking Services & Information Technologies, The University of Chicago, 1996. http://www.uchicago.edu/a.resource-guide/ss.s-hpm.html

[8] Open Software Foundation, OSF DCE DFS Administration Guide and Reference, Revision 1.1. Prentice Hall PTR, Upper Saddle River, New Jersey, 1995.

[9] RFC 1701 Generic Routing Encapsulation (GRE).

[10] RFC 1702 Generic Routing Encapsulation over IPv4 Networks.

[11] Coyne, R. A., and Hulen, H., "An Introduction to the Mass Storage System Reference Model, Version 5," Proc. 12th IEEE Symp. Mass Storage, Monterey, CA, IEEE Computer Society Press, April 1993.

[12] Coyne, R. A., Hulen, H., and Watson, R. W., "The High Performance Storage System," in Proc. Supercomputing 93, Portland, OR, IEEE Computer Society Press, Nov. 1993.

[13] Conte, T. M., and Gimarc, C. E., Fast Simulation of Computer Architectures, Kluwer Academic Publishiers, 1996.

[14] Ferrari, D., Serazzi, G., and Zeigner, A., Measurement and Tuning of Computer Systems, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

[15] Krishna, C. M., ed., Performance Modeling for Computer Architects, IEEE Computer Society Press, 1996.

[16] Harrison, P. G., and Patel, N. M., Performance Modeling of Communications Networks and Computer Architectures, Addison-Wesley, New York, 1993.