NASA Technical Memorandum 106900

Implementation of a Campuswide Distributed Mass Storage Service

The Dream Versus Reality

Betty Jo Armstead
Sterling Software
Cleveland, Ohio

and

Stephen Prahst
Lewis Research Center
Cleveland, Ohio

Prepared for the
14th Symposium on Mass Storage
sponsored by the Institute of Electrical and Electronics Engineers
Monterey, California, September 11–14, 1995

National Aeronautics and
Space Administration

Implementation of a Campuswide Distributed Mass Storage Service The Dream Versus Reality

Betty Jo Armstead
Sterling Software
Cleveland, Ohio

and

Stephen Prahst
National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio

SUMMARY

In 1990, a technical team at NASA Lewis Research Center, Cleveland, Ohio, began defining a Mass Storage Service to provide long-term archival storage, short-term storage for very large files, distributed Network File System access, and backup services for critical data that resides on workstations and personal computers. Because of software availability and budgets, the total service was phased in over 3 years.

During the process of building the service from the commercial technologies available, our Mass Storage Team refined the original vision and learned from the problems and mistakes that occurred. We also enhanced some technologies to better meet the needs of users and system administrators.

This report describes our team's journey from dream to reality, outlines some of the problem areas that still exist, and suggests some solutions.

INTRODUCTION

The Lewis Research Center is NASA's lead center for aeropropulsion, space power, space communications, space nuclear and electric propulsion, and microgravity science. Research in basic disciplines at Lewis is conducted in the areas of materials science and technology, structural mechanics, life prediction, internal computational fluid mechanics, heat transfer, and instruments and control. Lewis also participates in NASA's High Performance Computing and Communications Program (HPCCP), which is aimed at producing a thousandfold increase in supercomputing speed and a hundredfold improvement in available communications capability. These capabilities will let researchers solve today's "grand challenges"—the intensive, large-scale scientific and engineering problems critical to meeting national needs.

The Center has over 100 buildings with varying requirements for mass storage. These buildings include three wind tunnels, a full-size-engine test laboratory, and a number of other test facilities that support aeronautics, space power and propulsion, and space experiments.

The scientific computing environment includes a Cray YMP with a T3D parallel processor, a VAX Cluster, a 32-node IBM RS6000 Cluster, a 16-node IBM SP2, almost 1000 Unix workstations, several thousand personal computers, and about 400 Macintosh computers. Data acquisition systems for both steady state and transient data research are provided.

Since January 1989, a Lewis-developed Central File Archival and Migration Service (CFAM) (ref. 1) has been available to our Cray users. This service was extended to workstation users in June of 1991. CFAM is a client/server application between an IBM MVS server and Unix clients.

In 1990, a team was formed at NASA Lewis to address the mass storage requirements for the Center's entire computing complex. The goals outlined by this team were as follows:

(1) Provide long-term archival storage.
(2) Provide short-term storage for very large files, particularly from supercomputing and visualization.
(3) Provide distributed NFS access.
(4) Provide a repository for backing up critical data that resides on workstations and personal computers (PC).

The Lewis Mass Storage Team defined an architecture that consisted of a large central mass storage server with high-speed connections to central systems and a number of departmental storage servers. These centralized and departmental servers were to be connected by a dedicated network.

The architecture has been implemented with a central UniTree system connected to central systems via File Distributed Data Interface (FDDI) or via a high bandwidth High Performance Parallel Interface (HIPPI). High performance storage for supercomputing and visualization is being implemented with a Maximum Strategy file server. Departmental storage servers have been implemented with Unix file servers (Sun, DEC, and IBM). These distributed servers are interconnected via a dedicated FDDI ring. The servers provide both storage for users and a repository for centrally distributed software packages. In addition, Hierarchical Storage Management (HSM) software is being installed on the servers to provide the users with transparent migration and caching of files to the central server.

Because of software availability and budgets, the total service was phased in over three years. In this report, we describe, on behalf of the Mass Storage Team, the original system requirements, the resulting architecture, and the commercial technologies selected for each component of the mass storage service. We also describe the enhancements we made to those technologies and the lessons we learned during implementation. Finally, we describe the problems that still remain to be solved and our proposed solutions.

THE DREAM

In 1991, Lewis Research Center initiated a Mass Storage Project to provide a vendor-supported "open" mass storage service that would meet the Center's increasing storage requirements and replace CFAM. Because of budget constraints and availability of hardware and software, the service was to be phased in over several years. In the first phase of this project, the Mass Storage Team obtained end user requirements and defined the architecture of a Lewis Mass Storage Service.

Requirements

The Lewis Mass Storage Team assembled a detailed set of requirements for the Lewis Mass Storage Service. These requirements included large capacity, high reliability, an "open system" approach, a variety of storage types, fast performance, and a user-oriented interface.

Capacity.—We required the ability to store millions of files and Terabytes of data. Current estimates for the next 5 years are that we will be storing from 10 to 100 million files and 10 to 100 TB of data.

Reliability.—Our top requirement for a mass storage service was reliability since some research data, such as data acquired during an experiment, would not be reproducible. We knew that if researchers could not retrieve data they stored on the service, for whatever reason, they would lose trust and be apprehensive about using the service again.

A major component of reliability is data integrity: data must not be corrupted or lost. Traditional backup methods do not scale to Terabyte storage systems. We, therefore, require two copies of each file, one of which must be off-site.

A reliable mass storage service enables researchers to retrieve their data when they need it. Therefore, we required that system down time be minimized. Although some down time is typically needed for correcting faults and doing normal maintenance, we wanted to restrict down time to one scheduled 2-hr maintenance window per week.

Commercial Software and Support.—One of our key requirements was the use of "commercial" software and support. This requirement was driven by the limited size of our development/support staff. We reasoned that more resources would be expended on product development and support if a product had a market that extended beyond our site.

Open System.—An "open" system was also a requirement. For a mass storage service, the most important open feature is a standard format for data on the media and meta-data. With data storage quickly growing into the 10- to 100-TB region, we saw that it was becoming impossible to copy all the data from one format to the next. Instead we wanted a service that would copy data only when forced to do so by media obsolescence.

An "open" mass storage service also would allow any major system component to be replaced by a component from another vendor. This would not only allow competition but also protect us if a component vendor were to go out of business. Ideally, we wanted the ability to replace any major component of the service with a different vendor's product without copying all the data, retraining all the users, and redeveloping the administrative environment.

Most of the scientific-based mass storage systems we examined provided a standard user interface (ftp, NFS, rcp, etc.) but did not have standard data formats or administrative interfaces.

Storage Types.—We required the ability to provide a variety of storage options, including disk, tape, robotic support, and human vault support. We also required the ability to insert new storage devices as they become available.

Performance.—Our initial performance requirement was the ability to sustain five 1-MB/sec transfers for a total aggregate of 5 MB/sec. The requirement for a single transfer from supercomputers was 10 MB/sec. We expect performance requirements to increase and, therefore, require upward scalability of performance.

End User.—We required the ability to access data from the desktop environment via a file-system interface. Network File System (NFS) was the preferred protocol.

We also required standard user interfaces available on the clients: ftp, rcp and NFS. In addition we required an application program interface (API) to the storage system for custom application calls to storage.

Architecture

In designing the Mass Storage Service architecture, the key challenge was balancing the high capacity requirement, the higher performance requirement for supercomputing, and the NFS requirement for desktop systems.

Through our prior experience with hundreds of distributed clients accessing a single central NFS server, we knew that a central server would not scale to meet our needs. Based on our requirements and what we saw in the industry, we developed a distributed architecture as depicted in figure 1. The major components included a high capacity central component, a high performance component to meet the supercomputing requirements, and a distributed component to meet the distributed NFS load.

Central Component.—The central component would provide capacity in the multi-terabyte range and performance that provides an aggregate transfer rate of 5 MB/sec. The main use

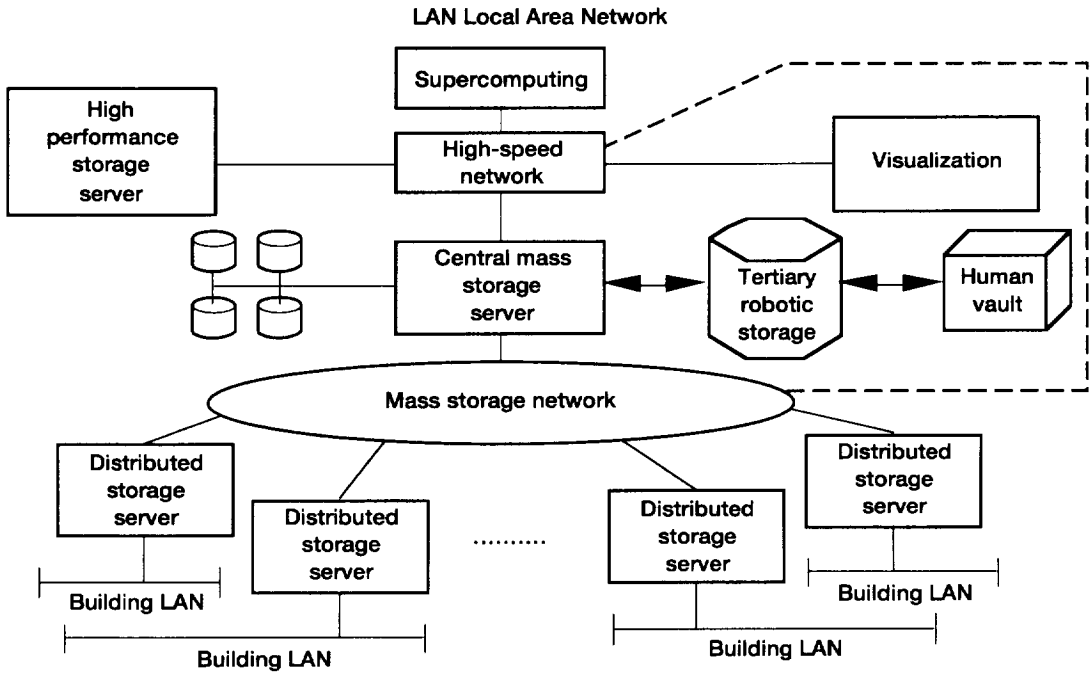

Figure 1.—Lewis mass storage.

of the central component would be for storing long-term archival data, such as acquired experimental data, backups, and migrated files from the distributed servers. The central component also would provide space for very large files that would not fit anywhere else, such as intermediate results from supercomputers. Over time, we envision the central component becoming a "back-end" service which would primarily store files that have not been recently referenced. Active files would reside on high performance and distributed components discussed in the next two sections.

We expected that the internal architecture of this component would be a computing system with a variety of large storage devices, such as disk, robotic tape, human vault, and optical disk. Hierarchical Storage Management (HSM) software would manage the storage resources transparently for users.

High Performance Component.—The high performance component would meet the demands of supercomputing and visualization. Internally, this component is viewed as either a third party transfer mechanism to a high-speed disk array or a special purpose file server. We expected this component to be capable of transferring data in the range of 10 to 50 MB/sec.

Distributed Component.—The distributed component would consist of multiple servers that would distribute the load and provide networked storage close to distributed computing systems, such as desktop Unix workstations. The distributed component also would provide NFS access to storage, which most Unix workstation users require.

We defined the internal architecture of this component as a set of NFS file servers, each with enough disk storage to store all active files for its particular work area. Based upon our network architecture, we determined that an NFS file server in each major building was appropriate. Our goal in sizing each server was to provide a 30-day window of storage. In other words, all files read or written within the last 30 days would exist locally on the distributed server. Hierarchical Storage Management software would manage the storage on the servers and move unreferenced files to the central server, transparent to users.

Server Network Connection.—In looking at the expected data transfer requirements between the components, we saw that dedicated networking would be required. To provide this, a dedicated FDDI connection between all the distributed servers and the central server was envisioned. The purpose of this dedicated connection was to provide rapid access to data files that had been moved from the distributed servers to the central server. This connection would also provide a good path for backing up the distributed servers to the central server.

THE REALITY

Based on our requirements and our vision of the ideal mass storage architecture, we built the Lewis Mass Storage Service by selecting technologies available in the marketplace and enhancing them to meet our users' needs.

Central Component - Central Mass Storage Service (CMASS)

We selected UniTree (ref. 2) as the central server for two primary reasons. First, though UniTree did not completely meet the "open system" requirement, it appeared to be more open than other mass storage systems available when we made the decision. Secondly, it was available from several vendors.

We tested two prototypes including an IBM RS6000 970 and in the summer of 1992 started backing up the distributed servers to UniTree on the IBM RS6000. After analyzing the results from a governmental "Sources Sought" on mass storage, we chose Convex as the UniTree vendor for the final system. In early 1993, we acquired a Convex 3220 with a 100-GB disk cache and a Storage Technology (STK) Silo, along with UniTree Version 1.6. In late summer a Metrum RSS-48 cartridge robot for storing very large files was added. This system was installed and made available to users in August 1993.

UniTree provides the required backup of client data by providing the ability to specify two copies of each archived file. In addition, the repacking function provides a method of migrating from one media type to another. This function was used when we migrated from STK 3480 tapes to STK 3490 compressed tapes. This migration allowed us to increase the near-line storage capacity by a factor of at least four.

UniTree as delivered was reliable, but it lacked many administrative functions. In addition, it generated a plethora of logs, and most of the entries were not time stamped. UniTree also lacked a tape volume manager for shelf volumes. In addition to writing a number of scripts for backing up the UniTree data bases and analyzing the logs, the Lewis team took on seven major projects to enhance UniTree and the central service. These are described in detail in the following sections.

Workstation Backup.—One reason for providing backup to UniTree was the need to provide backup to a different physical location. The need for an automated backup function for workstations to UniTree became apparent when several users overwhelmed the system by issuing a recursive remote copy (rcp-r) to back up their workstations. Also, we determined that some users were not backing up their workstations at all.

We provided a script that supported workstation backups for Sun, SGI, HP, Dec Ultrix and RS6000 workstations. This script uses a simple configuration file that makes it easy for users to backup specified directories (including a full backup at the start). The script uses a custom Lewis version of rcp that allows piping. The script uses UniTree wisely, for example, storing the backup as one file and immediately purging backup files from the disk cache.

Common Message Logger.—We started a joint project with DISCOS (a previous UniTree developer) to provide UniTree with a Common Message Logger (CML). We designed the CML to remove the text of a message from the source and to log all messages to a common tokenized (no message text) log. With CML, every message contains a time stamp, a severity level, identity of the module issuing the message, and variable data. The actual text of the messages is contained in a file distributed with UniTree and is used by a program that interprets the tokenized log. CML is now a part of UniTree 1.8 as distributed by UniTree Inc.

Enhanced Accounting.—To provide more detailed accounting, we wrote a set of perl scripts that provides accounting data for each user. The scripts generate a database from the tape header database, the tape map and the password file, which can then be queried to generate reports. The report generator is very flexible. It provides reports by family, copy number, or location and can be sorted by user id, number of files or total amount of data for a user. A summary report shows the total number of users, files, and Gigabytes for each family, copy number and location. Data locations are comprised of near-line storage, vaulted, and off-site vaulted storage. The summary reports also document the users by both number of files and total amount of data.

Volume Manager/Off-Site Backup.—The lack of a volume manager (there is one in UniTree 1.8) made it impossible to remove tapes from the STK Silo and keep track of them. We wrote a set of perl scripts to enter and eject tapes from the STK Silo and record them in a "shelf" database. We also modified the UniTree tape map to reflect these operations. We intend to convert this "shelf" database to a general purpose Volume Manager when one is available that works with UniTree. The volume manager, along with the UniTree file copy feature, allowed us to implement an off-site backup of UniTree files.

Client/Server Application to Add Users.—Lewis Research Center has long provided the ability for remotely joining users to the central systems. This is accomplished through a central database that currently resides on an IBM MVS system and tools provided to all Lewis administrators for managing their users' access to the central systems. We developed a client/server application to join a user to the Convex system and add the user to UniTree automatically. The administrator initiates this process by adding CMASS to the user's entry in the central database. The application also sets up the user directories and standard options files for the user. This process has relieved system personnel from a time-consuming administrative task.

Scripts to Emulate CFAM.—We developed a set of scripts to make it easier for Cray users to make the transition from Lewis' Central File Archival and Migration Service (CFAM) to CMASS. These scripts emulated the CFAM commands for storing, retrieving and deleting files, as well as listing the user's files. As part of this effort it was necessary to move 800 GB of data stored on CFAM to CMASS.

Usage Monitor.—We developed a monitor that shows CPU usage, gross network activity, and other Input/Output (I/O) on the central system. This monitor not only has helped identify bottlenecks but also has provided useful data for capacity planning on the central system.

As of February 1995, CMASS has about 2.5 TB of data and is growing at approximately 200 GB a month. Based on past history, we project the growth will not remain linear.

Distributed Component - Distributed Storage Service (DSS)

We implemented the distributed component with a set of commercial Unix file servers. As discussed earlier, we felt the only effective way to provide a large file service was to distribute the load on a set of servers. Since Lewis networks are implemented as a set of building LANS interconnected via a backbone FDDI ring, we chose to locate a server in each major building or subnet.

This arrangement provides good performance for all clients due to network locality, as well as a degree of reliability. If we had located all the servers at a central site, a network or power problem at that site would have made storage unavailable to the entire Center. This problem has never occurred with our distributed arrangement.

A total of twelve distributed servers were installed providing over 425 GB of storage. These were phased in over a period of three years with the first prototypes installed in 1991. A typical configuration consists of 20 GB of Small Computer Systems Interface (SCSI) disk, FDDI card for connection to the dedicated network, ethernet card for connection to the local segment, and 64-MB memory. Most of the servers are Suns.

We defined several administrative practices for the distributed component, including a standard directory structure and an administrative interface for the building work group. We automated the backups for the servers utilizing a set of scripts to send the backup images to the central component. The file-system interface was initially NFS with Appleshare for Macintosh and LAN manager for PC's added later.

For the Unix users, we implemented a global directory structure. A set of automount files were distributed to create a standard directory structure. The directory structure presented started with /r followed by the server name, for example /r/buckeye refers to the buckeye server. Under that, we named the file systems served from that server. We defined five standard types of file systems:

u/contains user permanent files (each user given a directory in u)
t/contains user temporary files
s/contains centrally supported software
l/contains local shared software and data
b/contains backups, mostly from PC's, Macs and the distributed servers

The u/ and b/ file systems were configured as migrated file systems using the OpenVision Enterprise Extension (HSM) software discussed next.

OpenVision Enterprise Extension and Netbackup.—OpenVision Enterprise Extension HSM is a Hierarchical Storage Management (HSM) product that increases the amount of file space available to users by migrating files from a local Unix file system to a remote Unix file system, in our case UniTree. Lewis acquired a beta version of the software in the summer of 1994, and, after some initial testing, our Mass Storage Team suggested a number of improvements. These included more separation of the migration and purge controls, an indicator in the directory listing output (ls-l) showing that files had been migrated and purged, and a stage command for users.

To reduce the time it takes to make space available when the file system fills, we suggested separating the controls for migration (copying the files to UniTree) and purge (erasing the files on the server disk cache) functions. This makes it possible to run the file system nearer its capacity. Since the migrated files have been copied to UniTree, it is not necessary to back them up. As a result, the dump files for the file system are much smaller since the backup software works in conjunction with the migration software. Netbackup backs up the i-nodes only for files that have been migrated.

The HSM software provides the following features for each HSM file system:

- The system administrator can specify the size of the smallest file that can be migrated/purged.
- Both the system administrator and the user can specify files that will not be migrated. There is a configuration-specified limit for the amount of storage a user can keep from migrating. A policy script (which can be altered by the administrator) is executed before any file is migrated. This permits sites to set their own policies for migration of files.
- Users can force migration of specified files, as well as stage in groups of files.
- Migrated files erased by end users are simply marked as obsolete and can then be deleted at an administrator's discretion. Users can request the administrator to restore an obsolete file that has not been deleted.
- Although the files on UniTree are owned by the server and have strange names, a related file also is stored that contains the information from the HSM database including the user's path name. This allows the rebuild and/or check of the HSM database against the files stored on UniTree. It also means the user must always access the file from the same server.

The user file access performance is very close to native NFS for files that have not been purged. Those that have been purged are staged in from UniTree, causing a delay that is dependent on whether the file is on the UniTree disk cache or on tape. Although a simple reference of a file will cause it to be retrieved, the stage command allows the user to bring in a group of files in parallel rather than serially.

High Performance Component.—We are currently installing a Maximum Strategy Profile XL file server to meet our high performance requirements (i.e., transfer rates of at least 10 MB/sec). This server is connected to computer systems, visualization systems, and the central mass storage component via a high performance network based on the NetStar GigaRouter (see fig. 2). We envision this server as a front end for high performance systems, similar, except for performance, to the distributed servers. We plan to work toward file migration from the file server back to UniTree.

Network.—A dedicated FDDI ring connects all the distributed servers and the central mass storage server. The traffic over the ring consists of backups, restores, file migrations, and file stages. The volume of the server backups, as well as the responsiveness required for file stages, led us to this dedicated design.

ATM	Asynchronous Transfer Mode
CMASS	CMASS Central Mass Storage Service
FDDI	Fiber Distributed Data Interface
HIPPI	High Performance Parallel Interface
LAN	Local Area Network
RAID	Redundant Array of Inexpensive Disks
WAN	Wide Area Network

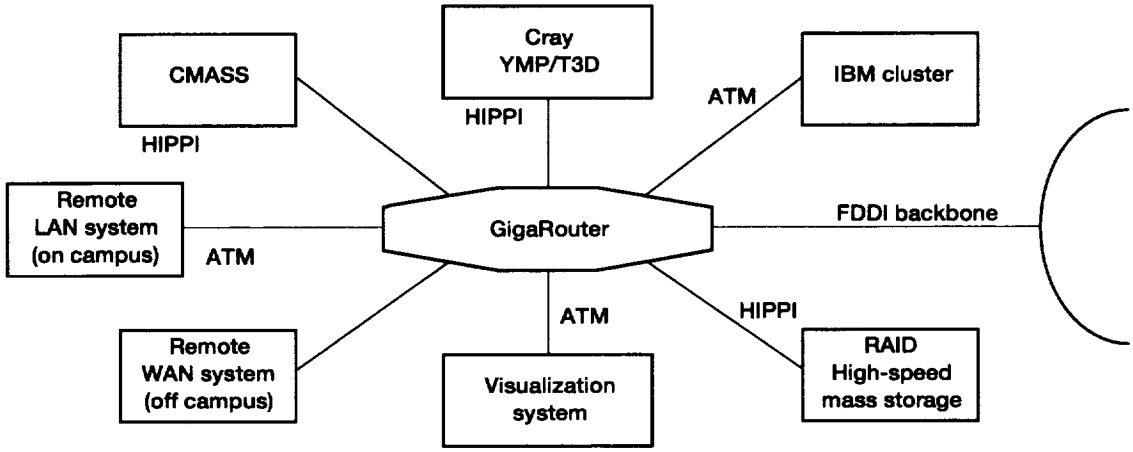

Figure 2.—Distributed high performance environment.

LESSONS LEARNED

Communicate With Users

When the Mass Storage Service was released to the users, the Mass Storage Team set up a reflector to which users could send questions and comments. Most of the traffic has been related to setting up the workstation backup, but we believe the quick responses to questions have been worth the effort. One area we would like to improve is notifying users of service outages. Currently, we use groups in Usenet and logon messages on the central systems, but we know we are not reaching all of the users.

Provide Backup Software for Workstations and Personal Computers

Initially, UniTree was made available to the users with a Graphical User Interface (GUI) ftp, regular ftp, and rcp. No workstation backup was provided. As a result, the users decided to use rcp-r for backing up their workstations with the following negative side effects.

- It created many very small files that can be scattered over several tapes.
- While it was easy to restore a single file, restoring all of the files on a workstation was almost impossible because the files were often scattered over many tapes.
- The backups were taking an unreasonably long time to create.

We quickly designed and wrote a backup script using a version of rcp that allowed piping. In the process of implementing this script, we found it was advisable to read a configuration file from the central system, which allowed changing of such items as the destination family, the maximum number of backups retained, and the final destination of backups (Metrum or STK). The backup script has become very popular and appears easy for users to install and use.

The PC and Mac users presented a slightly different problem. Since Unix style piping was not available for these environments, we implemented a file-system interface using Appleshare on the Macs and LAN Manager on the PC's. We created backup file systems on the distributed servers, which the PC's and Macs could access using any backup package. These backup file systems were set up as HSM file systems and were configured so that the backup images were quickly migrated and purged. In effect, the distributed servers were used as a spooling area for these backups while also providing the desired file system for file sharing.

Don't Assume You Know What a User Needs

We had assumed the users would like a GUI ftp interface to UniTree, so we put a lot of work into modifying MOXFTP to interface with UniTree. Now the use of MOXFTP is minimal. About ninety percent of the users prefer rcp or the backup script previously described.

Too often we discovered problems after the fact. For example, a user stored microgravity data from the Space Shuttle on UniTree

in what seemed like reasonably sized chunks. However, one of the first users to access the stored data tried to stage in over 3000 files. UniTree did not handle this request well, taking over 24 hr to retrieve the 3000 files. We talked with the user and in a more controlled fashion brought the files back in and put them (600 at a time) in tar files. This allowed the user to request 5 rather than 3000 files. In retrospect it would have been much less painful to create the tar files initially and store them as well.

In a second example, we accidently found out that another user was planning to store data about the health of the shuttle on UniTree. This case has a much happier ending because we conferred with the user before the data was stored. Here we provided an HSM file system that could hold about 20 GB on a local distributed server, and we are working with the user to guarantee that end users of the data do not make unreasonable requests but still get their data in a reasonable time.

Users Need to Know About Migrated/Purged Files

An operating system such as a Unix make can reference a lot of files. If the user is not aware that the files are migrated, and perhaps even purged, Unix can take a very long time because the files are retrieved serially as they are referenced. Thus the ftp, rcp and NFS interfaces must allow the user to determine whether files are migrated, and the user must have the ability to stage files in parallel. These operations must support multiple files and wild carding.

Improve Large File System Backups

Initially, one of the uses of UniTree was to back up the distributed servers. Although this use of UniTree has been successful, the backups became very large (more than 10 GB in a single backup file) as file systems grew. The large backup files are fine for restoring the whole file system. However, they are particularly painful to use if all that is required is a single file. One of the solutions to this problem was to use HSM on the file system and use the migration function to back up user files. When done this way, the file system backup contains only the nonmigrated files and the i-nodes of the migrated files and the HSM databases. While this reduces the problem to some degree, the disk cache on the HSM file system can still be very big. We are working with Open Vision to get their backup product to break the backup files into more manageable size chunks.

Monitor CPU/Network Performance

Because the Mass Storage System (including the distributed servers) makes heavy use of the networks, network problems are often encountered before anyone is aware that anything is wrong. Therefore, it is necessary to diagnose this type of problem quickly, notify the system manager, and log the information. Although we have been successful at identifying outages and sending e-mail to the appropriate people, slow transfers, which often indicate network problems, are still not usually detected. One attempt to rectify this has been to add logging to the central system from the backup scripts. This is an exception type logging of either failures or much slower than usual transfers.

Manage Disk Cache Based on File Type and Size

Prior to installing the HSM software on the distributed servers, management of the disk cache was handled by setting user quotas. Although this method works, disk quotas for a large number of users are difficult to manage. One can take the risk of either filling the file system or always under-allocating storage. Using HSM, the problems of managing the disk cache show up in a different way. The version of HSM we are running does not support quotas for the server disk cache. As a result, one user can monopolize the whole disk cache. Currently, this is being handled manually. However, we plan to implement a gross disk quota by using the policy script to choose files to migrate.

UniTree has problems similar to HSM in managing the disk cache. For example, the backup of servers and workstations was being done at night and full backups were staggered over several weekends. These files, which in many cases were never referenced, were taking over the disk cache because the control of what files to purge was related to size and last access date. In addition, "time on the disk cache before purging" was set to 3 days, and the controls could not be set differently for different types of files. After several weekends where almost all of the nonbackup files were purged, we acquired the ability to set a "purge immediately after migration" flag. This flag is now set in all of the backup scripts that we have written, and the crunch on the disk cache has been somewhat relieved.

Because we allow users to login to the Central Mass Storage Server, we put their home directory in the UniTree file system and we were having problems with Unix "dot" environmental files being purged. We relieved this problem somewhat by setting a "do not purge" flag for an existing file. However, this flag did not remain set if the file was modified or replaced.

Reasoning that purging small files does not return much space, we finally decided that a better solution was to modify the purge selection criteria so that it would not purge files under a certain size, except in dire circumstances. (It should be noted that we can make these types of changes only because we have a source code license.)

WISH LIST FOR THE FUTURE

Our experience with the distributed servers and the CMASS server has been enlightening. Some of the problems we experienced helped us create a wish list for the future.

This wish list has come about for several different reasons. Some of the items are a result of experience with our total Mass Storage Service in its current architecture. Others have arisen

because of additional requirements from the user community. For example, one group at Lewis needs to store company confidential data. Currently, the companies involved do not trust the Mass Storage service or the networks for this type of data, hence the need for security and data encryption.

Other requirements are coming from users who need to store meta-data, which they can query for the data they need to do a given job. Finally, the people maintaining the Mass Storage Service are driving yet another set of administrative requirements.

The following sections contain a description of our dream for the future:

Open Mass Storage Systems

Mass storage systems have a particularly important need for Open Systems. As data archives grow, the difficulty in moving from one system to another becomes greater. Some vendors may be able to read other vendors' tapes, but will they all? What if your site is using a system to which few other sites subscribe? Are you going to have to do the job yourself, or will you need to spend a substantial amount of resources having it done?

Ideally, standard formats for archives will emerge so that sites will not need to rely on a single vendor, or themselves, to shepherd their archives into the future. The mass storage industry should provide this so that archives could be moved to different systems without expensive reformatting. It should be noted that the UniTree Users Group is working with UniTree Inc. and the UniTree vendors to set some guidelines for the definition of open mass storage systems. In particular, they will be documenting the media and meta-data formats for UniTree.

Standards also should provide the ability to grow a multivendor system. Interoperability standards should allow a site to integrate a new system component, such as a tape system, without requiring a major integration effort. For this reason we support the efforts of the IEEE Storage System Standards Working Group (Project 1244) (ref. 3).

Global Name Space

Early on we considered installing Andrew File System (AFS) (ref. 4) on all of the workstations and central Unix systems. At the time AFS was not available for every type of workstation and central system. Furthermore, we were not successful in selling the idea to our user community since workstation vendors were providing NFS, but not AFS. As a result, we do not have a campus-wide global name space. In fact, users storing files on the distributed servers can access the files only via the server on which they were created. The files that are migrated to UniTree are owned by a single user and the names are unique but not meaningful. We would like to find a way to convert from this type of architecture to a global name service, such as Distributed File Service (DFS) from Open Software Foundation (OSF).

One feature of the HSM software that will help achieve this goal is that the relationship between the user path and name is preserved in the HSM File Database. In addition, each file on UniTree has a companion file containing the same information contained in the HSM File Database. From this information, we should be able to convert to a Global Name Space for files. Our vision is that DFS will become a part of every Unix system and that we will successfully transform our Mass Storage Service to exist in that world.

Archive Management

As the amount of data archived increases, the management of that data becomes important. The issue of archive management has been discussed to a great degree. We also have this requirement. The current method of organizing and locating data is outdated. It is based on Unix since all of our storage systems are based on Unix. Users organize their files via a hierarchical directory structure and file sharing is through Unix permissions, which are inadequate. Some areas of special concern are described in the sections which follow.

Corporate Data Storage.—Research data is often stored in individual user accounts. When individuals leave the organization, there should be a procedure to determine what to do about their data. Whether the data is assigned to someone else depends on how conscientious the research group is. We need a corporate-level data storage environment where important information is stored independent of individuals. The challenge is in developing a method of doing this that is painless to our researchers.

Storage of Meta-Data.—We require the storage of additional meta-data (ref. 5) with archived files. This meta-data might be part of the archived file structure or might be stored separately in a related file or database.

Application Program Interface.—We require an interface for information access. The query method may be the correct approach, but we are not sure. Human factors should play a key part in this goal. How do people think when they are looking for information? What organization and tools will help them wade through Terabytes of data? We believe that some amount of research will go on in the data access area for some time. To facilitate this, mass storage systems need improved interfaces over ftp, NFS, and rcp. We would like to see an Application Program Interface (API) to the storage system so that operations such as staging, retrieving, moving, and so forth, can be implemented reliably under program control. Although ftp is adequate for accessing stored files, there may still be a need for the API. Examples are rearranging data into files to more nearly match the users reference pattern, extracting meta-data from files as they are stored and grouping of like files from a user-initiated program.

Grouping of "Like Files."—Another practical aspect in this area is that of grouping "like files" together on the same media. If information is related, it is best not to spread it out over

thousands of tapes. As mentioned previously, with UniTree we solved this problem by collecting related files into one file using tar. However, this is a crude solution to the problem.

Maintenance and Capacity Planning

The system engineers who maintain the Mass Storage Service have identified several ways to reduce the time it takes to manage the Mass Storage Service and to present the end users with a transparent, secure, and reliable way to store data. Currently, system people spend too much time analyzing logs, managing tapes, spotting network problems and moving files around. In addition, the storage service must provide data for capacity planning.

Performance Monitoring and Logging.—A system manager should be able to tell when the mass storage service needs attention by checking status displays that continually show the state of the service. These displays should not only show the current state of the system, but also allow the manager to look at past history. The system manager must be able to identify problems (current and past), bottlenecks, and resource status (where resources include CPU, memory, disk cache, the network, and near-line storage media). Historical data must be recorded to provide for capacity planning. In order to answer end user questions, a manager must also be able to track files in the system from the time of creation, through migration, and including deletion.

Transaction Logging.—Most of the logs in mass storage systems seem to be more related to development of systems and are mostly used to diagnose problems. The transactions being discussed here are storing and retrieving of files on mass storage. Transaction information such as the start and end time, user id, client node name, and file size for each transfer should be logged. Summary information should include the number of files stored, the number of files retrieved, and the total amount of data stored and retrieved for each user. This data is needed for both reporting system usage information and as an aid to system administrators.

Currently the only transactions recorded in UniTree are those that use ftp. However, a large part of the data is stored and accessed through NFS on the servers and rcp directly to UniTree. The actual log entries should contain the date and time, the file name, the user id, client node name, the number of bytes transferred and the transfer rate. This kind of log would aid in spotting network and storage media problems.

Dynamic Hierarchies.—We define a hierarchy as the path a file takes through the Mass Storage Service. For example, a user may store a file on one of the servers, which then moves it to the disk cache of the central server, then to a near-line robot, and finally to a cartridge sitting in a vault. Currently, the time a file stays on each media depends on time last referenced and its size. This hierarchy is usually defined by the media types and cost. The media types may include multiple performance disks, optical disks, and multiple kinds of tape. The site manager should have the ability to define multiple migration/caching hierarchies (ref. 6) and determine which files or families of files belong to the various hierarchies. For example, the manager may decide that files smaller than a specified threshold number of bytes should remain on the disk cache, and that all files classed as backup should be sent directly to tape. The end user should also have the ability to choose the hierarchy for a file, since the hierarchy chosen will determine the cost and performance for storing the file.

Media Management.—To handle media with unpredictable lifetimes effectively, a utility is needed that can detect when media is nearing the end of its useful life. The system should allow this procedure to take place automatically when a cartridge is loaded for use. If a tape is found to be near failure, all the data should be moved to a new cartridge and a message sent to the system logs/monitors that the old cartridge can be removed from the system. For those tapes that are infrequently referenced, a sniffer that identifies deteriorating media should be provided. The site manager should also be able to set in motion a slow migration from an old media type to a newer one. This is to prevent the Mass Storage Service from ending up with media it can no longer read or a crash conversion of data to a new media. Although UniTree provides the repacking function, the control is by family rather than media type.

Security and Encryption

As presently constituted, the Mass Storage Service at Lewis cannot provide storage for proprietary data. This situation must change since we have a requirement to store company confidential data. We, therefore, need at least one server and the central server to support both Kerberos and encryption. Eventually, we would expect this requirement to be satisfied by DFS and standard encryption methods.

CONCLUDING REMARKS

The Lewis Research Center has designed and implemented a Mass Storage Service consisting of twelve NFS servers on a dedicated FDDI ring, a central server running UniTree, and a high performance file server. The distributed servers use hierarchical storage management software that moves data not recently referenced to the central server. The central server provides multi-Terabyte capacity and can be scaled to higher capacity by adding additional storage devices. A Maximum Strategy file server provides storage for clients with higher performance requirements such as supercomputing and visualization.

This configuration, along with the UniTree and Enterprise Extension software, meets the requirements for ***Capacity, Reliability*** and ***Commercial Software and Support***. The ***Open System*** requirement has been partially met by using commercial software that runs on multiple platforms. We are still

depending on the IEEE Storage Systems Standards Working Group to provide standards that would allow mixed vendor components. We are working with the UniTree User's Group (UTUG) to define standards for tape formats and the data bases that contain meta-data. At this point in time, we can substitute another vendors UniTree without converting all of the archived data.

We can add new storage devices as they are supported by UniTree vendors, meeting our ***Storage Type*** requirement. We have implemented an offsite backup of UniTree data tapes and meta-data and UniTree has proven to be quite reliable, meeting our ***Reliability*** requirement.

Distributed users access the service through a file system interface, typically NFS, on the distributed servers which satisfies the ***End User*** requirement. Files not recently accessed are migrated to the central server. High performance clients access the Maximum Strategy file server using Network File System (NFS). Direct access to the central UniTree system is also provided via rcp and ftp. These commands are supported on ethernet, Fiber Distributed Data Interface (FDDI), and High Performance Parallel Interface (HIPPI) networks.

Performance requirements have been met, except when the central server becomes too heavily loaded. Also meta-data access on the central server (commands such as ls-l) has been a source of complaints. We plan to address these problems by moving user access to the distributed servers and having the central server play the role of a "back-end" service.

After 2 years of operation, approximately 3 TB (consisting of 2 million files) of data are stored by the service. The service is reliable but requires much care and maintenance and we are constantly working to improve the service. Our emphasis is now moving toward improved data management using more meta-data and improving performance.

REFERENCES

1. Perry, R.S.: Now That the Storage Train's Here, Can We Get Everybody on Board? Proceedings of the Twelfth IEEE Symposium on Mass Storage Systems, Apr. 1993, pp. 37–44.
2. McClain, F.: DataTree and UniTree: Software for File and Storage Management. Digest of papers: Tenth IEEE Symposium on Mass Storage Systems, May 1990, pp. 126–128.
3. IEEE Reference Model for Open Storage Systems Interconnection. Mass Storage System Reference Model, Version 5, IEEE Storage Systems Standards Working Group (Project 1244), Sept. 8, 1994.
4. Goldick, J.S., et al.: An AFS-Based Supercomputing Environment, Proceedings of the Twelfth IEEE Symposium on Mass Storage Systems, Apr. 1993, pp. 127–132.
5. Bedoll, R.; and Kimball, C.: The Importance of Meta-Data in Mass-Storage Systems. Digest of papers: Tenth IEEE Symposium on Mass Storage Systems, May 1990, pp. 111–116.
6. Buck, A.L.; and Coyne, Jr., R.A.: Dynamic Hierarchies and Optimization in Distributed Storage Systems. Digest of papers: Eleventh IEEE Symposium on Mass Storage Systems, Apr. 1991, pp. 85–91.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 1996	Technical Memorandum

4. TITLE AND SUBTITLE

Implementation of a Campuswide Distributed Mass Storage Service
The Dream Versus Reality

5. FUNDING NUMBERS

WU–None

6. AUTHOR(S)

Betty Jo Armstead and Stephen Prahst

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E–9580

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM–106900

11. SUPPLEMENTARY NOTES

Prepared for the 14th Symposium on Mass Storage sponsored by the Institute of Electrical and Electronics Engineers, Monterey, California, September 11–14, 1995. Betty Jo Armstead, Sterling Software, 21000 Brookpark Road, M.S. 142–2, Cleveland, Ohio 44135 (work funded by NASA Contract NAS3–26100); Stephen Prahst, NASA Lewis Research Center. Responsible person, Stephen Prahst, organization code 1380, (216) 433–5240.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 62

This publication is available from the NASA Center for Aerospace Information, (301) 621–0390.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

In 1990, a technical team at NASA Lewis Research Center, Cleveland, Ohio, began defining a Mass Storage Service to provide long-term archival storage, short-term storage for very large files, distributed Network File System access, and backup services for critical data that resides on workstations and personal computers. Because of software availability and budgets, the total service was phased in over three years. During the process of building the service from the commercial technologies available, our Mass Storage Team refined the original vision and learned from the problems and mistakes that occurred. We also enhanced some technologies to better meet the needs of users and system administrators. This report describes our team's journey from dream to reality, outlines some of the problem areas that still exist, and suggests some solutions.

14. SUBJECT TERMS

Mass storage; Hierarchical storage management; Distributed computing; Data management

15. NUMBER OF PAGES
12

16. PRICE CODE
A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

National Aeronautics and
Space Administration

Lewis Research Center
21000 Brookpark Rd.
Cleveland, OH 44135-3191