Hierarchical Storage Management System Evaluation

Thomas S. Woodrow[1]

Report RND-93-014 August 1993

NAS Systems Development Branch
NAS Systems Division
NASA Ames Research Center
Mail Stop N258-5
Moffett Field, CA 94035-1000

---

1. NASA Ames Research Center, Moffett Field, CA 94035-1000

Hierarchical Storage Management Software Evaluation

Thomas S. Woodrow

Abstract

The Numerical Aerodynamic Simulation (NAS) Program at NASA Ames Research Center has been developing a Hierarchical Storage Management System, NAStore, for some 6 years. This evaluation compares functionality, performance, reliability and other factors of NAStore and 3 commercial alternatives. FileServ is found to be slightly better overall than NAStore and DMF. UniTree is found to be severely lacking in comparison.

1 Introduction and Problem Definition

The Numerical Aerodynamic Simulation (NAS) Program has been involved with Mass Storage Hardware and Software since its inception in 1984. In 1985, the Mass Storage Subsystem (MSS) Project was initiated to create an Hierarchical Storage Manager (HSM) to meet the needs of the NAS Program. Since 1985, there have been several releases of MSS software running under MVS and UNIX on Amdahl hardware and currently under UNIX on Convex hardware. During this period, several commercial alternatives appeared. These alternatives have now been available in the market for some time and have been subjected to the testing rigors of the marketplace. It is a good time to evaluate these packages, compare features and performance, and make a determination whether to continue internal development of NAStore or embrace one or more of the commercial alternatives.

The following packages are compared: the Data Migration Facility from Cray Research, FileServ from E-Systems, NAStore from the NAS Program, and UniTree from Open Vision.

This paper is arranged as follows: configuration, functionality, performance, hands-on experience, detailed observations, and conclusions. Weights are assigned for each category and totaled to give a final recommendation. The evaluation weights for each category are shown below:

Category	Weighting Factor
Functionality	11
Performance	23
Reliability	23
Stability	23
Ease of Use	9
Outstanding Problems	4
Miscellaneous	7
Total	**100**

2. Test Configuration

Three systems were used during testing; 2 Cray Y-MPs and a Convex C3820. The two Crays are both running DMF in production and have slightly differing configurations and loads. The Convex is one of two recently acquired systems for the NAS Mass Storage Subsystem.

Columbia YMP 2E 1/16, HiPPI UltraNet, Model E IOS, 1 - 4 channel TCA (total of 4 -3 MB/s paths), 16 - 3480 tape drives in 3 STK 4400 Silos, located in the Ames Research Center - Central Computing Facility
Filesystem: composed of DS-42s and DD-62s connected to one controller - not striped
UNICOS 7.0.4.3
Sun 4/330 running ACSLS 4.0
DMF 2.0
This system functions as a dedicated file server.

Reynolds YMP 8/256, HiPPI UltraNet, Model D IOS, 1 - 4 channel BMC (total of 4 - 3 MB/s paths), 16 - 3480 tape drives in 2 STK 4400 Silos, located in the Ames Research Center Numerical Aerodynamic Simulation Facility
Filesystem: composed of DS-42s and DD-49s connected to one controller - not striped
UNICOS 7.0.4.1
Sun SPARCstation running ACSLS 3.0
DMF 2.0
This system functions primarily as a compute engine. File serving functionality is of secondary importance except in support of the computation capability.

Pancho Convex C3820, 1 GB RAM, HiPPI UltraNet, 2 - 2 channel TLI interfaces (total of 4 - 4.5 MB/s paths), 16 - 3480 tape drives in 2 STK 4400 Silos, located in the Ames Research Center Numerical Aerodynamic Simulation Facility
Filesystem: composed of 4 wide stripes using DKD-504 disks across 4 IDCs
ConvexOS 10.2
Sun SPARCstation running ACSLS 3.0
FileServ 2.1.5, NAStore 2.1.1, UniTree 1.7.1.14
This system will function as a dedicated file server.

3. Features

Major Software Components

<u>DMF (from Cray Research Inc.)</u>

- tpdaemon
- dmdaemon
- Media Specific Processes (MSP)
- Applications
  - dmput, dmget, dmlim, dmmode, several others

The Data Migration Facility is primarily composed of the dmdaemon, MSPs, some kernel modifications and some application programs. The dmdaemon handles all of the requests initiated by the user applications and initiates MSP actions. The system tape daemon, tpdaemon, allocates devices and controls tape mounts and dismounts. There are only two applications a user is ever likely to use: dmput and dmget. However, any UNIX file open will automatically cause a migrated file to be restored to disk. There are numerous other applications which are used primarily by operators. DMF uses CTREE routines to maintain database information. DMF is a relatively simple implementation and has a comparatively small number of functional pieces.

DMF makes use of UNICOS kernel hooks to initiate automatic file restoration on file opens. The structure of the inode has been modified to show the migration state of a file. File reads and writes will block until a file is resident on disk.

<u>FileServ (from E-Systems Inc.)</u>

- INGRES database daemons
  - (iidbms, dmfacp, dmfrcp, iigcn)
- FileServ daemons
  - (fs_cpyreq, fs_media, fs_mcontrol, fs_monitor, fs_cpyresp, fs_resource, fs_alloc_s,

fs_fcontrol, fs_rem_s, fs_admin, fs_data)

Applications

fsstore, fsretrieve, fschstate, fsmedinfo, fsmedlist, fsaddrelation, fsaddclass, others

FileServ uses the INGRES commercial database. There are many daemons and specialized processes which run to make FileServ function. There are a large number of applications available. One of the strengths of FileServ is the wealth of applications to check/modify status information and control variables. Another benefit is the ability to track the state of individual stores and retrieves while they are in progress.

FileServ makes use of the ConvexOS kernel modifications which cause a file open to automatically restore files from tape. File reads and writes will block until a file is restored to disk.

<u>NAStore (developed at NAS)</u>

voldaemon

rashd

Repository Controllers - Manual 3480, Manual 3420, ACS 3480

Applications

forcearc, frestore, arcbuild, archive, reclaim, volstat, volvary, vls, rls, vol, others.

NAStore has the following functional elements: the Volume Manager, *voldaemon*, uses repository controllers to mount and dismount tape volumes, Rapid Access Storage Hierarchy, *rashd*, uses the Volume Manager to move files in and out from various media through Repository Controllers. There are several applications, but only a few that users are likely to use: *forcearc* and *frestore* move files in and out from tape, *volstat* checks the queue of requests and *volvary* checks the state of tape devices. It is difficult, but possible, to see if your archive or restore is in progress. One of the strengths of NAStore is the relatively few pieces that must be running. NAStore uses BTREE routines to maintain file and tape information. One of the unique features of NAStore is that the tapes are standard ANSI format and are readable by ANSI tape readers.

Like DMF and FileServ, the ConvexOS kernel modifications are used to cause restoration on a file open. NAStore adds several fields to the Convex inode structure and therefore modifies several file related utilities to make use of the additional information stored in the inode (newfs, newst, ls, others).

<u>UniTree (from Open Vision)</u>

tpdaemon

UniTree daemons

(unamed, udiskd, utaped, uftpd, unfsmntd, tapesrvr, disksrvr, namesrvr, pdmsrvr, tapemovr, diskmovr, pvrsrvr)

Applications

uftp

UniTree is composed of a collection of specialized daemon processes, the Convex Tape Daemon, and a modified ftp daemon. There are a large number of running daemons. Under periods of high activity, we might expect high CPU load and lots of context switching. We were not able to check this due to several bugs encountered during testing. UniTree uses BTREE routines to maintain file and tape information. UniTree restricts all access to user data by forcing access through ftp or NFS. This is a restriction unique to UniTree.

UniTree formats, labels and controls its own file system on the Convex. This means that UniTree file systems are quite different than other file systems on the Convex. Rather than take the approach of using the kernel, UniTree runs entirely in user space with a number of daemons responsible for all event handling and file movement.

One of the problems with this approach is that access is limited to an application which links the UniTree library. UniTree provides NFS and FTP which have been modified to use the daemons for file opens, reads and writes.

An obvious strength to this approach is that it is very easy to port UniTree to new platforms. There are versions of UniTree available on systems ranging from Convex and Amdahl to IBM RS6000, SGI and Sun. There is also support for a wide range of output devices as well: IBM 3480 cartridges, Metrum VHS, 9-track round tape, others.

One artifact of a large number of UniTree ports is that each may be implemented from a different base version of the software. Most were ported and are supported by separate vendors. Many of the points brought up during testing are likely to be indicative of the large number of UniTree installations even though the implementation hardware may be different. Convex is the largest installed UniTree implementation with over 20 of 54 installations according to Max Morton of Open Vision.

Storage Model

Each of the packages manages files in collections or groups. These collections govern how files are segregated on tapes (i.e. files within the collection can be mingled on tape).

Package	**Grouping Mechanism**
DMF	filesystem
FileServ	class
NAStore	user
UniTree	family

DMF

All files within a filesystem are managed together. This results in files from several users on a single tape.

When a file is disk resident, it appears as any other UNIX file in an *ls -la* listing:

-rw-r--r-- 1 woodrow npo 309848 Jul 20 23:27 file

When it is only available on tape, this is indicated by an "m" appearing in the first column of an *ls -la* listing:

mrw-r--r-- 1 woodrow npo 309848 Jul 20 23:27 file

FileServ

There are system defined *classes* which are mapped to directory trees. All files in a class are managed together. Similar to DMF, this means that user files within a class can be intermingled on a single tape.

When a file is disk resident, it appears as any other UNIX file in an *ls -la* listing:

-rw-r--r--	1	woodrow	npo	309848	Jul	20	23:27	file

When it is only available on tape, this is indicated by an "a" appearing in the first column of an *ls -la* listing:

arw-r--r--	1	woodrow	npo	309848	Jul	20	23:27	file

The *fsfileinfo* command will also list the state of the file, the number of tape copies and whether the file is resident on DISK, TAPE or DISK and TAPE.

NAStore

All files for a specific user are managed together. NAStore uses a hot tape model for tape writes. This means a "hot" primary and backup tape, will be written and filled before a new tape is used. There are special cases where multiple hot tapes can occur; for example when a file will not fit at the end of the current hot tape. There may be space available at the end of the tape, just not enough for the current file. In this case, a new hot tape is selected, resulting in multiple hot tapes. Files which will span tapes (files larger than 200 MB) are exempted from the hot tape mechanism and will start a new tape immediately.

NAStore, unlike DMF, FileServ and UniTree does not know how much space is left on tape. This means that NAStore may try to put a file on tape before finding out if it will fit. The other packages require the system operator to configure the tape size and then use this to determine whether a file should fit before trying.

When a file is disk resident, it can appear in one of the following two ways in an *ls -la* listing: 1) as any other UNIX file, or 2) with an "a" in the first column.

-rw-r--r--	1	woodrow	npo	309848	Jul	20	23:27	file
arw-r--r--	1	woodrow	npo	309848	Jul	20	23:27	file

When it is only available on tape, this is indicated by an "A" appearing in the first column of an *ls -la* listing:

Arw-r--r--	1	woodrow	npo	309848	Jul	20	23:27	file

Since the archive state information is available from a file listing, it is easy to track the file archival state without learning any new commands.

UniTree

Files are grouped in system defined families and used similar to UNIX groups (i.e. users enter a *setfam* directive within FTP similar to *setgrp* in UNIX - all subsequent stored files are associated with the family). All files in a family are managed together and user data can be intermingled on tape. UniTree uses a hot tape model for archival. Unlike NAStore, however, UniTree proceeds linearly through tapes regardless of file size. This means that a spanning file (a file greater than 200 MB) can start in the middle of a tape. This is a decision unique to UniTree.

NFS does not display the archival state of a file. Within FTP, a *dir* or *ls* command will display whether a file is in disk cache (DK) or archived (AR). There is no mechanism to see if a file is on both disk and tape.

Some Useful Functionality

Here is a short list of some basic functions which were used or would have been used during the evaluation. Most of these are provided by several of the packages, some are unique to one package or another. Some of these are not available and are included as suggested capability.

Function	Description	D	F	N	U
List user tapes	List all tapes (primary and backup) in use by a user, class, family, filesystem.	X	X		X
List files on a tape	List all files on a specific tape.	X	X		
File Status	List the archival state of a file	X	X	X	X
	List the tapes which hold a file (primary and backup)	X	X		
List Tape Blocks Used	This is a modification of the UNIX *du* utility to list blocks used for migrated files.				
Drive Status	List the system defined drives and their current state.	X	X	X	X
Tape Status	List the % full or bytes on tape	X	X		X
	List the files known to reside on the tape.	X	X	X	
Archive/Restore Status	List the percent complete for all archive or restore operations				
	Given a user/family/class/filesystem id, identify all archives or restores active and their completion state				
Media Lists	Provide information on tape media in use (virgin/free, labeled, allocated, bad, etc.)	X	X	X	X
	Provide the option for a long or tabular presentation of this data		X		
Media Labeling in Parallel	A Parallel labeling utility.	X	X		
Media Recycle/Reclaim in Parallel	A Parallel Recycle/Reclaim utility - i.e. the operator should not have to manually parallelize this function.	X	X		
Media compaction	The ability to compact out the old dead files from a tape and hence realize some savings in the tape inventory.	X	X		X
Database Lists	List all versions of a file stored in system	X			
Misc. Configuration Stuff	Specify tape quota		X		
	Specify user files to keep on disk	X	X	X	

There is a difference between file migration, the act of copying a file to tape, and file truncation, the act of removing the file from the disk after migration. UniTree treats this as a single logical operation, i.e. its *forcemig -all* command. DMF, FileServ and NAStore all have separate utilities which perform migration and truncation. As long as migration is performed as files are written to disk, truncation can be performed as necessary to keep the disk usage within a controlled range. For DMF, FileServ and NAStore file truncation is performed very quickly and is not measured. Each HSM has a different utility to build the list of candidate files for truncation. These provide a sorted list of files eligible for truncation from disk. The list is then used to maintain a file system percent utilization.

Tape compaction is provided by all of the packages except NAStore currently. This is an important capability and will be required for production use.

FileServ and DMF are very complete in functionality. NAStore needs tape compaction. UniTree is hit for the lack of analysis tools and capability.

Scores
Functionality [11]: DMF: 11 FileServ: 11 NAStore: 10 UniTree: 8

4. Performance

NAS sizes the Mass Storage System to hold the latest 30 days of data on disk. This is based on the assumption that the most current data has the highest usage and should have the fastest access. File access from disk or the highest level of the storage hierarchy is therefore one of the most important performance elements of any HSM. Of course, some files will be accessed which are not resident on disk. This makes the tape system read performance of secondary importance. The ability of the system to process multiple requests simultaneously is important. For this reason, we measure both individual user and system aggregate performance for tape operations.

Tape read performance is usually more important to the user than write performance, since most tape writes are initiated by the system and can be scheduled. Tape reads are initiated in response to an immediate need by a user. Tape read performance should therefore be as fast as is possible on the media. Tape write performance is measured for individual files and under various system loads. File migration and truncation performance from disk is measured to see the relative ability of these packages to keep up with a stream of input data.

A user workload of file reads and writes using ftp with a mix of resident and migrated files is used to make an overall relative assessment of the HSMs. This is probably the most realistic measure of user performance of all the tests since it incorporates disk, tape and network performance.

4.1 Individual file, disk read and write performance

Test Description
Files of various sizes (256K, 10M, 75M, 300M) are read from and written to disk. Sequential I/O tests are used, since file access will typically require a network transfer and will involve reading an entire file. A program called *multirate* is used to measure disk performance on the Crays and the Convex. This utility was acquired from Convex and has been used on site for some time.

There are several factors to note when looking at the disk results:

1) The DMF file systems installed on the 2 Crays deliver 8 - 10 MB/s for individual file reads and writes because of the current hardware configuration.
2) The file systems installed on the Convex can deliver 28-31 MB/s for individual file reads and writes.

Given these points, we are also interested whether the HSM delivers the native filesystem performance or degrades it.

Results

From Figure 1, NAStore and FileServ both deliver the highest performance to and from disk. Based on the measurements, DMF, FileServ and NAStore all deliver native filesystem performance. However, UniTree degrades the native file system performance by about 66%. The DMF is given 2 scores: 1) compared to the installed Convex filesystem the performance is lower - however this is not a DMF problem 2) if the filesystem were at the same or higher performance then we would expect DMF to deliver this performance - however this is an assumption. Both of these scores will be discussed in the performance summary.

Scores:
Disk Read/Write DMF: 0.31 (1.0) FileServ: 1.0 NAStore: 1.0 UniTree: 0.33

4.2 Individual file, tape read and write performance

Test Description
Files of various sizes (256K, 10M, 75M, 300M) are written to and read from tape using the HSM user commands (dmput/dmget, fsstore/fsretrieve, forcearc/frestore, put/get[2]). Elapsed wall clock time is used to measure the duration of the write/read. Measurements include the time to archive a primary and backup copy of a file. All timings take into account tape mount activity. The intent is to report rates which are as close as possible to user experienced rates.

It is important to note the following in this test:

1) the peak 3480 tape drive performance is 3.0 MB/s
2) it is interesting to note the ability of each package to attain tape drive peak
3) higher average performance is expected for larger files as mount activity is reduced as a proportion of the total elapsed time

Results
During testing, the following was observed

DMF sustained 2.0 MB/s on writes and reads
FileServ sustained 2.0 MB/s on write and 2.7 MB/s on reads
NAStore sustained 2.7 MB/s on writes and reads
UniTree sustained 1.5 MB/s on writes and reads

All HSMs exhibit an increase in performance as file size increases. Tape mount time was observed to be 20 - 50 seconds depending on silo load.

Figure 2 shows that NAStore dominates tape read and write performance, especially as file size increases. UniTree is quite a bit slower than the rest. Also, Columbia outperforms Reynolds in read tape performance with DMF. This is most likely attributable to the newer silos which likely have had engineering improvements since the installation of the silos at the NAS facility. The effect of the engineering improvements is faster tapes mounts. Unfortunately, we have incomplete numbers for Columbia due to the scarcity of dedicated Cray time.

---

2. UniTree access is through ftp. This is true even when accessing files on the local system. *Put* and *get* are the ftp commands used to move files in and out of a UniTree controlled filesystem. *Get* is also used when retrieving a file from tape.

Figure 1. Individual File Write and Read Disk Performance Rates

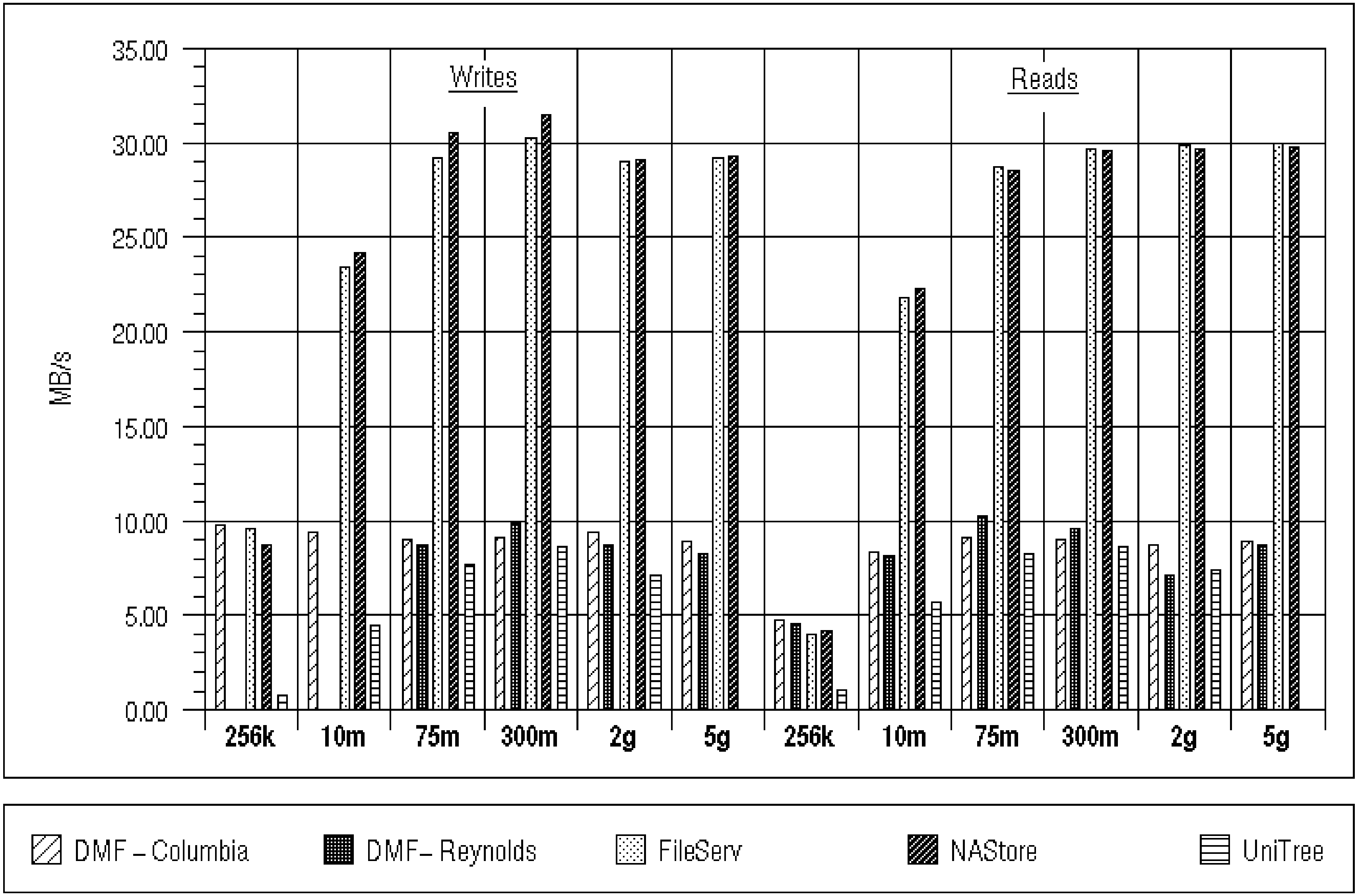
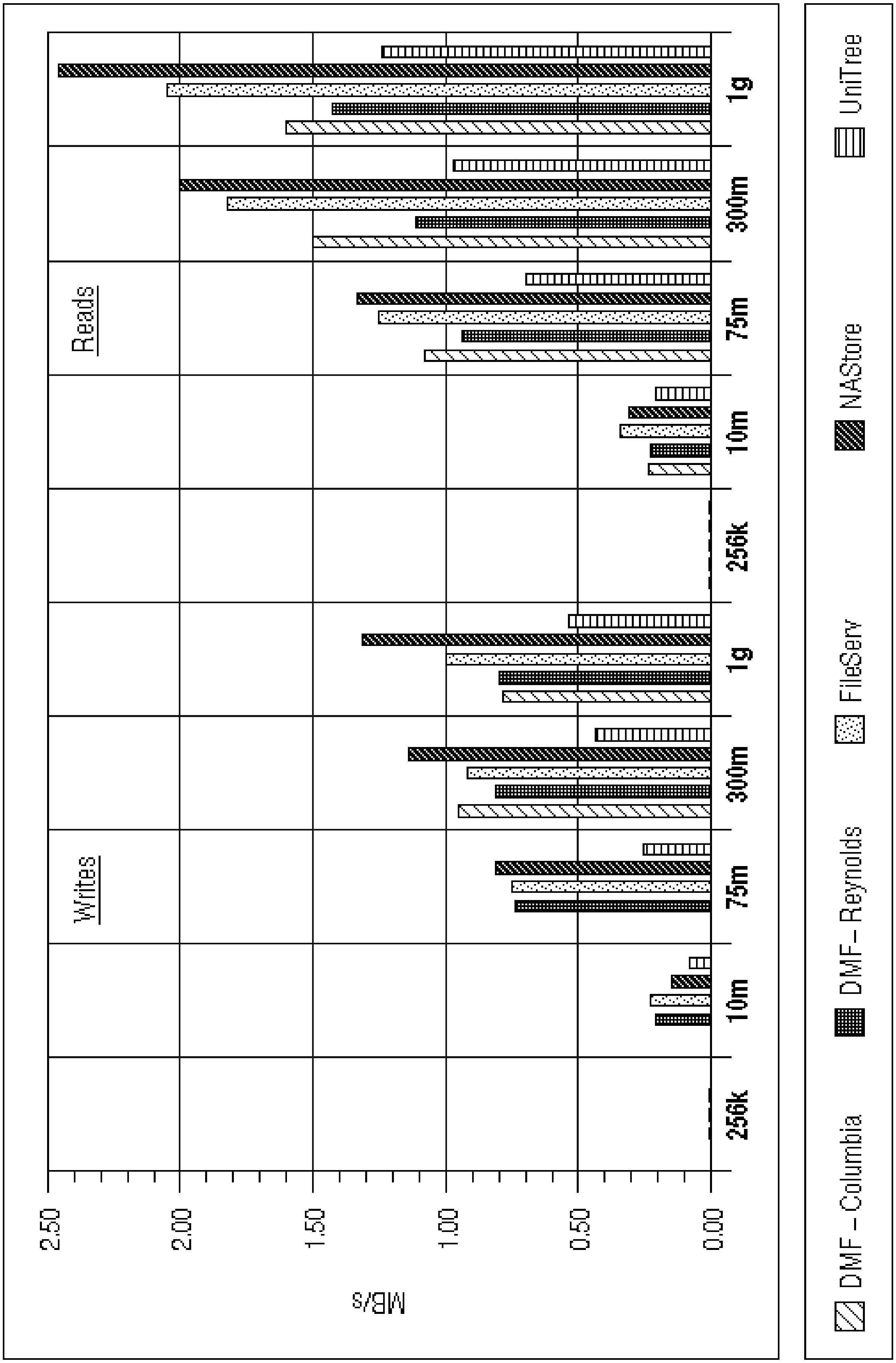


Figure 2. Individual File Write and Read Tape Performance Rates



Scores:

Tape Write	**DMF: 0.71**	**FileServ: 0.81**	**NAStore: 1.0**	**UniTree: 0.43**
Tape Read	**DMF: 0.74**	**FileServ: 0.94**	**NAStore: 1.0**	**UniTree: 0.55**

4.2.1 Tape handling optimizations

Following is a list of the kind of optimizations observed among the different systems. Given the fixed performance of the tape media, it is desirable to maximize the delivered performance from that media. Some of these optimizations are simple to implement. We hope all of the packages will continue to improve in this area.

Optimization	Exists in Package
Parallel primary and backup tape writes.	DMF, NAStore
Asynchronous primary and backup tape writes.	DMF
Pre-mount tapes for spanning file reads	NAStore
Check pending requests for a tape prior to a dismount	DMF, FileServ
	NAStore has implemented this feature but it has not yet been tested
Optimize wild card transactions (reads and writes)	DMF, FileServ, NAStore
Cache data on disk and perform only full tape writes	DMF
	UniTree - this was not working in our version
Use any available tape for a write (not just the "HOT" tape)	DMF, FileServ
Spanning files start with a new tape	DMF, FileServ, NAStore
Optimal use of tape system data paths.	NAStore

Listing an optimization does not mean that it is necessarily the best choice, it was simply observed. NAStore and UniTree both sacrifice some tape write performance in using HOT tapes. However, both may reap improved performance during restores because improved locality of user or family files should reduce tape mounts.

NAStore tries to make optimal use of the system data paths by rotating mounts through controllers. This has noticeable effect during periods of light load.

4.3 Multiple file tape read performance

<u>Test Description</u>
Multiple files are written to tape in simultaneous streams. Each stream is composed of consecutive 256K,10M, 75M, or 300M files. Streams have differing numbers of files based on file size, in an attempt to keep all streams active for the entire test. The test is run two different ways:

1) with separate store commands for each file
2) with a single wild carded store command for an entire stream.

Data in Figures 3 and 4 show the average rate across all streams against the simulated load level. Figure 3 shows the performance on the system when issuing a separate store command for each file. In this test, a higher level of tape mounts and dismounts was expected. In Figure 4, a single wild carded command performs all of the writes. In this case, the system aggregate performance is expected to rise due to reduced mounts and dismounts.

System aggregate performance over time data is also gathered for these tests, although only on the Convexes. These charts are included as an appendix. They are not used to formulate scoring, but are discussed briefly in section 6, **Miscellaneous Points and Observations**.

Results

All candidates were tested in this area, but UniTree had a problem on the Convex which caused FTP to block during file reads from tape. This made it impossible to automate testing of a multiple file restore. We were unable to work around this problem. As a result, UniTree has rather few measured results from here on. This problem does not mean that users cannot get files from UniTree file systems using FTP, but it does mean that several commands may be necessary to retrieve a single file. This was determined to be unworkable for performance testing purposes.

Figure 3 shows the average system throughput to restore files for various simulated user loads. Comparing the results from Figure 3 and Figure 2, we might expect that the system throughput to be higher. The primary reason system throughput appears lower is that we average across all file sizes. FileServ has a slight edge in performance over both DMF and NAStore in this test.

Figure 4 shows the wild carded restore system performance. FileServ dominates at lower system loads and then NAStore outperforms all others at higher loads. Both FileServ and NAStore exhibit significantly improved system performance when file restores are wild carded. DMF (reynolds) does not display improved performance in this category which is puzzling, especially since DMF does optimize mount/dismount activity. One possibility for the performance behavior may be a rewind/seek between files restored from the same tape - this is not confirmed. The DMF columbia result at 8 simulated users suggests that columbia could outperform reynolds in this test. We were unable to complete the test matrix for columbia because of lack of additional dedicated time.

The performance difference between Figures 3 and 4 suggests that users would be well advised to aggregate file reads and writes from an HSM.

Scores:

Separate:	**DMF: .84**	**FileServ: 1**	**NAStore: .79**	**UniTree:0**
Aggregated:	**DMF: .48**	**FileServ: 1**	**NAStore: 1**	**UniTree:0**

4.4 Migrate and Truncate data from Disk

Test Description

This test measures the speed the of each package to archive and truncate files from disk. This is an important measure of how well an HSM can keep up with data as it hits the system. This test differs from the previous test, only in how it was initiated. A single, wild carded command is used to initiate the store. After the store completes, a truncate command is issued. The total elapsed time to archive and truncate the files from disk is measured.

Figure 3. Average System Throughput – Separated Reads, Various Simulated Loads

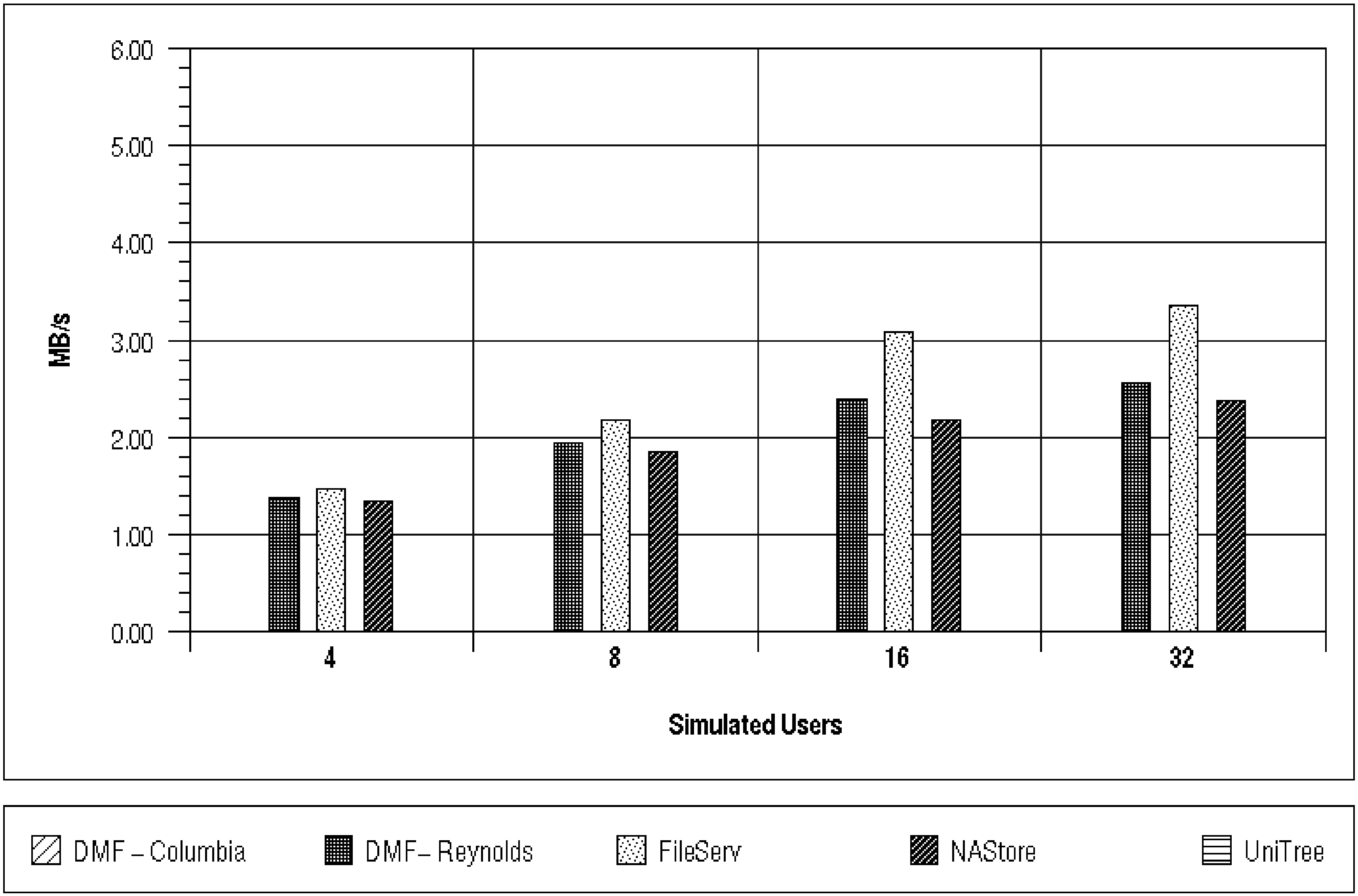
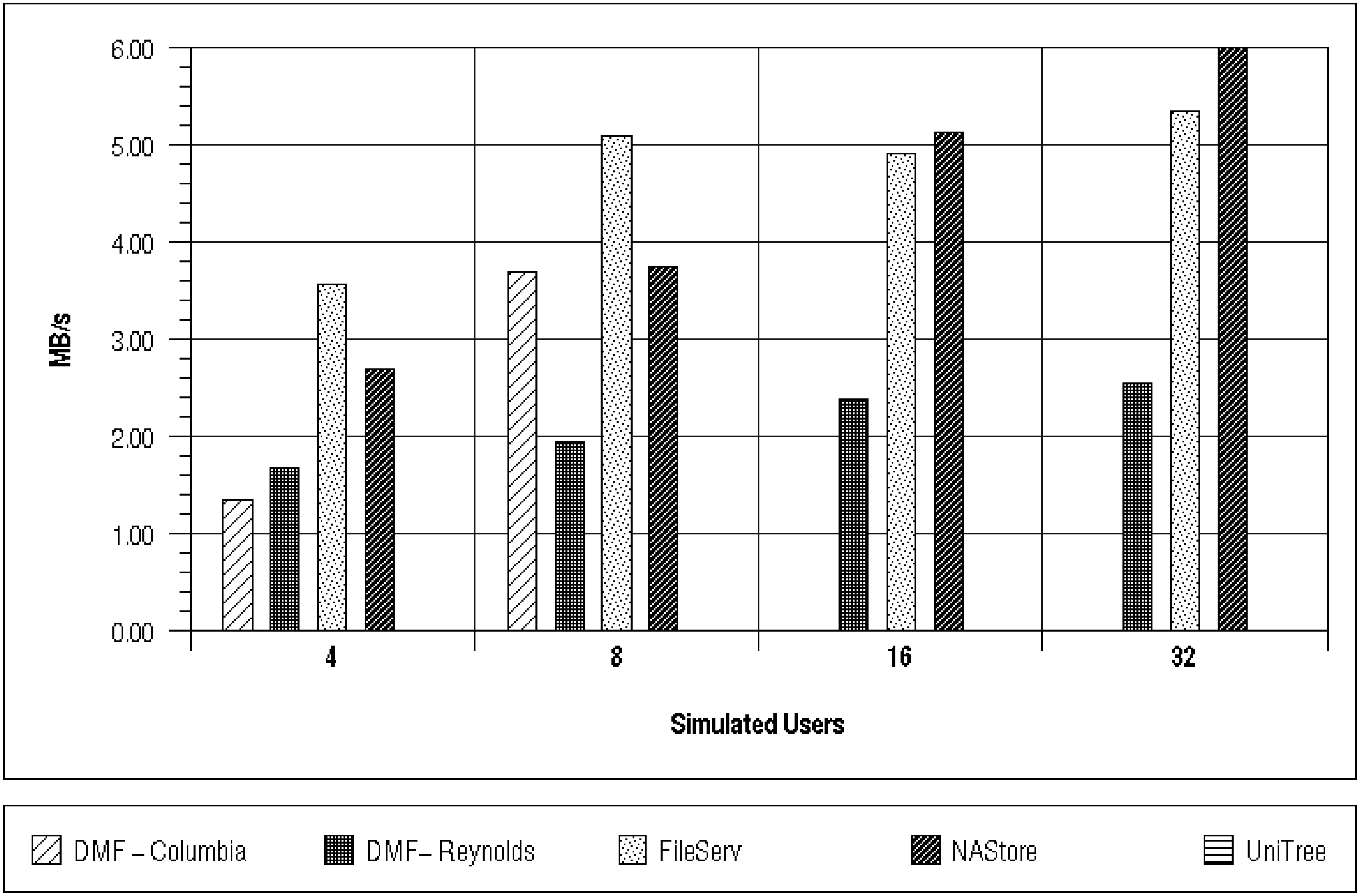


Figure 4. Average System Throughput, Aggregated Reads, Various Simulated Loads



Results
DMF, FileServ and NAStore all perform in the same approximate range here, although FileServ has a slight edge. UniTree is significantly slower. One of the primary reasons for this is that UniTree writes a single stream of data (1 tape drive) and does the primary and backup copies sequentially. This seems to be a feature in all UniTree versions according to discussions with other sites.

Scores:
Migrate/Truncate: DMF: 0.81 FileServ: 1.0 NAStore: 0.74 UniTree: 0.17

4.5 Workload performance disk and tape read and write

Test Description
This test incorporates elements of network, disk and tape performance by simulating the activity of 26 simultaneous users using ftp over UltraNet. The user data population was examined on the previous storage system and users were consulted to define a workload profile for the system. The workload definition follows:

User Type	Description	File Volume	% of population
1	Workstation Backup	1 - 150K files	15
2	Miscellaneous Small	50 - 20K files	20
3	CFD steady state	5 - 10M files 5 - 75M files	45
4	CFD small unsteady	20 - 75M files	15
5	CFD large unsteady	50 - 300M files	5

Ideally we would have run increasing loads of users based on this breakdown till we saturated the system. In the future, perhaps we will do this. In the interest of timeliness we selected a representative load; 26 users.

Each storage management system is initially configured with a predetermined number of files resident on disk and migrated to tape. Simulated users issue ftp get and put commands to move files in and out of the system under test. The test is driven from a Cray C-90 and uses native UltraNet paths to ensure that neither the driving system CPU nor the network are the bottleneck. High, low and average performance are calculated and plotted by file size.

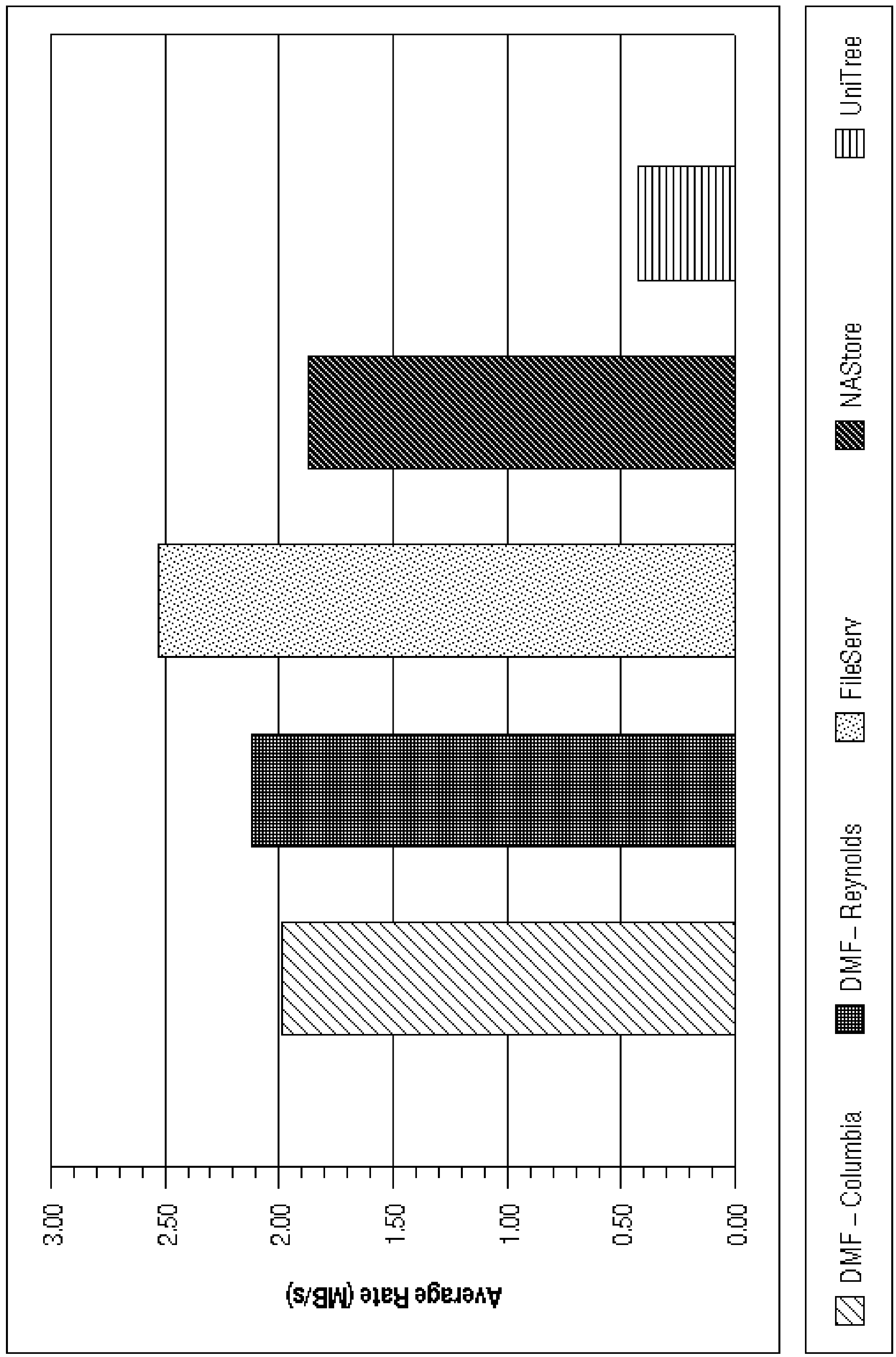
One unintended limitation imposed on this test is the C-90 filesystem. Since gets and puts utilize the C-90 file system, this creates a ceiling for individual file transfers. In the future, we will use a faster file system on the driving system.

The following are considered in ranking the packages:
1) the average performance at each file size
2) the variation in performance at each file size

Results

Figure 6 shows the high, low and average rate for reads and writes by file size. The data points on the X axis are broken out by package at each file size. DMF, FileServ and NAStore results are shown from left to right.

Figure 5. Effective System Rate to Archive/Truncate 8.0 GB of User Files



NAStore performs better than both DMF and FileServ at all file sizes. In almost all cases, NAStore has less variation between the high and low rate than the others. FileServ, although usually 2nd in average performance, has a large fluctuation in performance at each file size.

Given the C-90 file system limitation mentioned above, we still see significant performance differences between the packages.

Scores:
Workload: DMF: 0.65 FileServ: 0.82 NAStore: 1.0 UniTree: 0.0

4.6 Performance Totals

Performance tests were broken into 4 functional areas and given the following weights:

Disk Read and Write	100
Tape Read	55
Tape Write/Migrate	20
Workload	25

These weights reflect the relative importance of disk read and write performance and tape read performance. As long as the tape write performance can keep up with the daily load, it is adequate. The workload reflects all of these categories, but is perhaps a more representative measure of user performance.

Existing systems were utilized for performance testing. We are well aware that both Convex and Cray are capable of configuring systems with more or faster hardware. We made a conscious choice to measure existing systems and rank them based on the current configurations. File system performance, within reason, is largely a cost issue.

The table below shows only the totals for the performance scoring. The complete, weighted scoring table is Figure 7.

Total Points	**210**
DMF	**100.25* (169.25)**
FileServ	**192.27**
NAStore	**192.42**
UniTree	**40.47**

From a performance standpoint, FileServ and NAStore are identical. DMF is also a strong contender. The DMF performance total is 169.25 - a strong third, were we to assign full points for the disk performance test. Again, we ran against existing hardware configurations. UniTree would definitely be higher without functional problems on tape restores.

5. Hands-On Experience

This sections summarizes experiences during installation, configuration and usage of the packages.

Figure 6. Workload Hi, Low and Avg. Rates by Size/Package

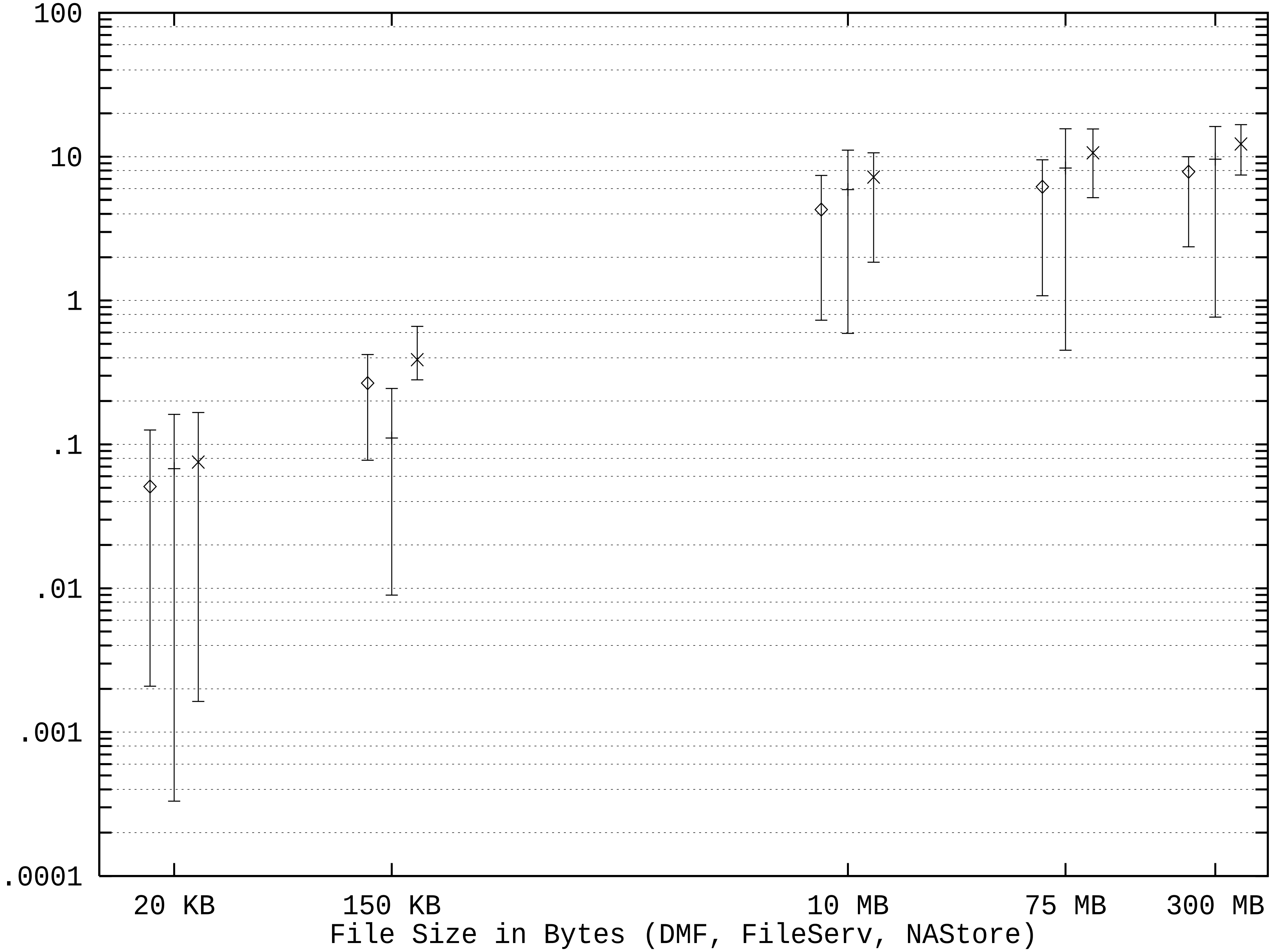


Figure 7. Performance Rank of HSMS Alternatives

	Disk Read/Write			Indiv. Tape Read			Indiv. Tape Write			Syst Tape Read (sep)		
	weight	act/max	points	weight	act/max	points	weight	act/max	points	weight	act/max	points
DMF	100	0.31 (1.0	31 (100	15	0.74	11.1	7	0.71	4.97	20	0.84	16.8
FileServ	100	0.99	99	15	0.94	14.1	7	0.81	5.67	20	1	20
NAStore	100	1	100	15	1	15	7	1	7	20	0.79	15.8
UniTree	100	0.27	27	15	0.55	8.25	7	0.43	3.01	20	0	0

	Syst Tape Reads (agg)			8.0 GB migrate			Workload		
	weight	act/max	points	weight	act/max	points	weight	act/max	points
DMF	20	0.48	9.6	13	0.81	10.53	25	0.65	16.25
FileServ	20	1	20	13	1	13	25	0.82	20.5
NAStore	20	1	20	13	0.74	9.62	25	1	25
UniTree	20	0	0	13	0.17	2.21	25	0	0

Total Points	210
DMF	100.25 (169.25)
FileServ	192.27
NAStore	192.42
UniTree	40.47

5.1 Reliability

An exhaustive reliability test including testing of failure modes was not undertaken. Therefore results in this section are limited to experience during testing. There was no data loss or corruption by any of the packages during performance or functionality testing.

During NAStore beta test, it was discovered that exec did not block correctly on files which were non-disk resident. This resulted in erratic behavior until the file was entirely disk resident. A simple kernel change was implemented and this was cured.

Scores
Reliability [23]: **DMF: 23** **FileServ: 23** **NAStore: 23** **UniTree: 23**

5.2 Stability

Each package has unique stability issues.

DMF

- We encountered difficulty when bringing down DMF with tape write processes active. This turned out to be a cockpit error, but the experience points out the complexity involved with this systems.
+ Overall DMF is rock-solid stable.

FileServ

- Out of 20 - 30 sessions, we did cycle the software several times to clear hung behavior. This was related to a bug in the first tested release. In the second release, this problem was fixed.
- There was a problem involving badly formed silo addresses, by the mount utility. This had the effect of flipping some address information and making tapes unavailable. The work around involved running an SQL script continuously in the background. This script worked fine, but added a significant load to the database. During heavy load, we could get into a confused state. This was also repaired in the second release we received to complete testing.
+ A system crash occurred during FileServ testing. We were able to restart the software after reboot without problem.
+ FileServ is rather simple to start and stop. Once started, it is very stable.

NAStore

- Corrupted individual user rash index files. Rash recreated these indexes later with no data loss. This bug bas been repaired and tested.
- Tape devices were vary'ed off during high levels of activity. This meant that we closely monitored activity during NAStore tests. This was attributed to an error prone TLI driver on the Convex. There were several instances when the voldaemon core dumped and disappeared during high incidence of TLI errors. There were some changes made in NAStore to back off on a drive when a high number of errors were seen, this did reduce the number of drives which were vary'ed off significantly and removed the voldaemon core dumps.
+ NAStore stayed up and worked well after it was initiated.

UniTree

- Encountered problems getting NFS to work at first, eventually got things working with help from Convex.
- There were numerous times when it was difficult or impossible to dismount the UniTree NFS partition during shutdown. This had an effect on the stability of the entire system, even after a reboot.
- UniTree has a tendency to halt completely when it encounters a bad or badly labeled tape. This

meant constant monitoring of the log files during testing.
+ In general UniTree could be trusted to stay up if it did not encounter and tape problems.

It is difficult to determine the stability of software over a short period of time. There is extensive local experience with DMF in production usage on several Cray systems. There is also less extensive, but still significant experience with NAStore during beta test (4 - 5 weeks) and during unit and integration testing (2 - 3 months). Our experience with UniTree and FileServ has been for a short period of time (2 - 4 weeks). This experience certainly influences our impressions on stability.

Scores
Stability[23]: DMF: 23 FileServ: 21 NAStore: 21 UniTree: 15

5.3 Configuration, Documentation and Ease of Use

DMF
+ Documentation is terse, but complete
+ Tape recycling/compaction is very simple to use
- Some operator utilities seem needlessly difficult to use (ex: dmvdbgen)
- If a file is on disk, the user may not be able to determine if it is also on tape.

FileServ
- Adding tape media to the system requires a pass through the cap door. This means that even if you already have a silo full of virgin tapes, FileServ would like you to remove them and define them by entering them through the cap. This problem has been fixed for the next version.
+ Tape labeling is done in parallel using all drives - painless and fast.
+ Tape recycling is very simple.
+ Easy to define classes, add relations, change configuration, administer package.
+ Easy to view archive activity (fschstate, fsqueue, fsmedinfo)

NAStore
- Tape labeling is manual, sequential, single stream (however, the user can start up multiple streams with separate lists) - user supplies a list to a utility
- Tape recycling is manual.
- On-line documentation is not as strong as the other packages.
+ Strong software architecture documentation.

UniTree
+ Configuration is mostly localized to just a few files
- FTP/NFS access is maddeningly restrictive.
- knowledge and viewing of UniTree log files is essential to monitor activity

Scores
Config, Doc, Ease of Use [9]: DMF: 7 FileServ: 9 NAStore: 7 UniTree: 5

5.4 Outstanding Issues

DMF
- Dmdidle is required to force data to tape, when there is not enough data waiting for archival. This is a root only command. If a user determines that he wants to force a file to tape, but the system does not have a tape full ready to write, the user will block until the system has enough data to fill a tape. It is not possible for a user to find out how much data must be sent to work around this situation.
- Limit of 8 processes per MSP. There is a primary and secondary Media Specific Process. Although there are 16 tape drives, during restores all files are required to go through the primary MSP, unless a tape error is encountered. This places a seemingly arbitrary limit on the number of simultaneous restores possible. This has apparently been fixed in a patch to be released very soon.

FileServ

- Flipped silo identifiers on each tape with the initial version of FileServ we tested. This was resolved with a bug fix.
- Tape entry through cap door in this version of the software. This has been fixed in the next version.

NAStore

- Rash indices were corrupted several times. This has been fixed and tested.
- Tape devices were vary'ed off during high load. A work around is in place.
- Voldaemon died several times - sometimes requiring a reboot to clear a hung named pipe. This has been fixed and tested.

UniTree

- Single tape failure halts migration
- "quote wait get" hangs
- Extremely difficult to measure performance
- Wrapping files > 2 GB. During an ftp put of a 5 GB file, the system was observed to wrap the file at 2 GB back to 0 and then continue. The final file size was 1 GB. There were no errors reported.

Scores
Problems[4]: DMF: 3 FileServ: 4 NAStore: 4 UniTree: 0

6. Miscellaneous Points/Observations

- It is difficult to tell the state of an individual user transaction with DMF. During "puts", files are gathered in a caching directory until a full tape is ready, then written. Although the dmput may return immediately, a user's files may not get out to tape for some time. During "gets", files may be in the process of restoration, but unless the user can make the association between his tape ids and what is currently mounted, he is unable to tell if his transaction is active or not (this is especially true on a busy system).

+ After doing a DMF dmput, a user's disk utilization is immediately reduced by the size of the file, even though the file may not have left the disk cache. This is certainly a desirable feature for the user who is running at or near his quota on a Cray.

+ DMF used a small percentage of the system CPU. During the simulated user activity to disk, system usage ranged from 1 - 5%. This percentage is based on an 8 CPU system.

- DMF uses a proprietary tape format.

- When the FileServ daemons are not present (i.e. running) directories and files under FileServ control cannot be listed or read.

- FileServ's reliance on INGRES means the FileServ administrator should be versed in SQL. INGRES is also one of the major bottlenecks in FileServ performance, since all transactions must refer to the database. DMF, NAStore and UniTree all utilize BTREE or CTREE for database functionality rather than use a commercial database.

- FileServ developed a high system load on the Convex. During the 32 Simulated User restore, system load averaged 50 - 80%. While this is not critical, it is a warning sign. NAStore was well below 50% utilization. We were not really able to drive UniTree hard enough to know how it behaved under load.

- FileServ uses a proprietary tape format.

+ NAStore is the only package which delivers bytes immediately as they hit system memory.

+/- NAStore sacrifices some archival performance to group files by user. This is based on the assumption that the individual user will benefit during restores, since his files will be closer together (fewer mounts during restores). I think the positive aspect of faster restores far outweighs the negative aspect of slower archives.

- NAStore is only in use at NAS. The only support for NAStore is provided locally. The only users who have experience with the system are at one location.

- It is difficult, but possible to track the state of an individual *forcearc* or *frestore* transaction within NAStore.

+ NAStore produces ANSI standard formatted tapes.

- UniTree is very difficult to measure. Using FTP for all transactions, makes measurement of the individual components of a store almost impossible. Although FTP reports the time to restore a file, it does not begin measurement until a file is resident on disk cache. Therefore the numbers FTP reports are suspect. Many times we were reduced to watching the UniTree log files.

- Although UniTree does give the user access to data through NFS, it is not possible to determine disk residency using NFS. File read and write performance using NFS, even on the local system, is poor (significantly lower than FTP) and is discouraged.

- UniTree uses a proprietary tape format.

All of the packages were hit for a proprietary tape format except NAStore. Many people seem to agree that this is important, however few have implemented using a standard.

System Aggregate vs. Time Plots give a picture into the system of how performance changes over time. These were produced on the Convex because of an existing utility called *syspic* and access to the source code. A utility exists on the Cray which could be modified to provide this data, but there was insufficient time prior to testing.

From these plots (included as an appendix), it is clear that there is large variation in performance over time. One performance goal for these packages could be that the average system tape performance during load approach (number of data paths * peak drive performance). On all of the systems tested, this number is 4 * 2.7 MB/s = 10.8 MB/s. Tape mounts and dismounts will always reduce the system aggregate performance from a theoretical peak, but with 16 drives, it should be possible to keep the data paths well utilized during periods of high load. A more aggressive goal would be to approach (number of data paths * data path rate) = 4 * 4.5 MB/s = 18 MB/s.

Scores
Misc [7]: **DMF: 6** **FileServ: 4** **NAStore: 6** **UniTree: 3**

7 Conclusions

Summary Scoring

Category	Weight	DMF	FileServ	NAStore	UniTree
Functionality	11	11	11	10	8
Performance	23	11.04 (18.4)	21.16	21.16	4.6
Reliability	23	23	23	23	23
Stability	23	23	21	21	15
Ease of Use	9	7	9	7	5
Outstanding Issues	4	3	4	4	0
Miscellaneous	7	6	4	6	3
Total	100	84.04 (91.4)	93.16	92.16	58.6

FileServ is the most well rounded product, based on all of the factors considered. NAStore is a strong second. DMF comes in third unless the filesystem performance is factored out. Removing the penalty for a slow filesystem, DMF still falls 3rd just behind NAStore. UniTree loses many points in the performance area on tests which were not completed due to a bug. If this bug had been fixed, UniTree would be a stronger contender but would still place fourth due to lower performance and stability.

The decision of which Storage System to put into long term production at NAS is a judgment call which will involve the technical comparison, cost information and other factors. A discussion of costs would have restricted the availability of this report and was therefore dropped. One of the other factors that will be considered in the production decision is the ability to influence or make change in the Storage System in response to local requirements. NAStore is certainly the easiest product for NAS Program to influence and change. There is no major internal development required to use NAStore in production service. There are some features which can and will be added, but these can be accomplished with a sustaining level of effort.

Acknowledgments

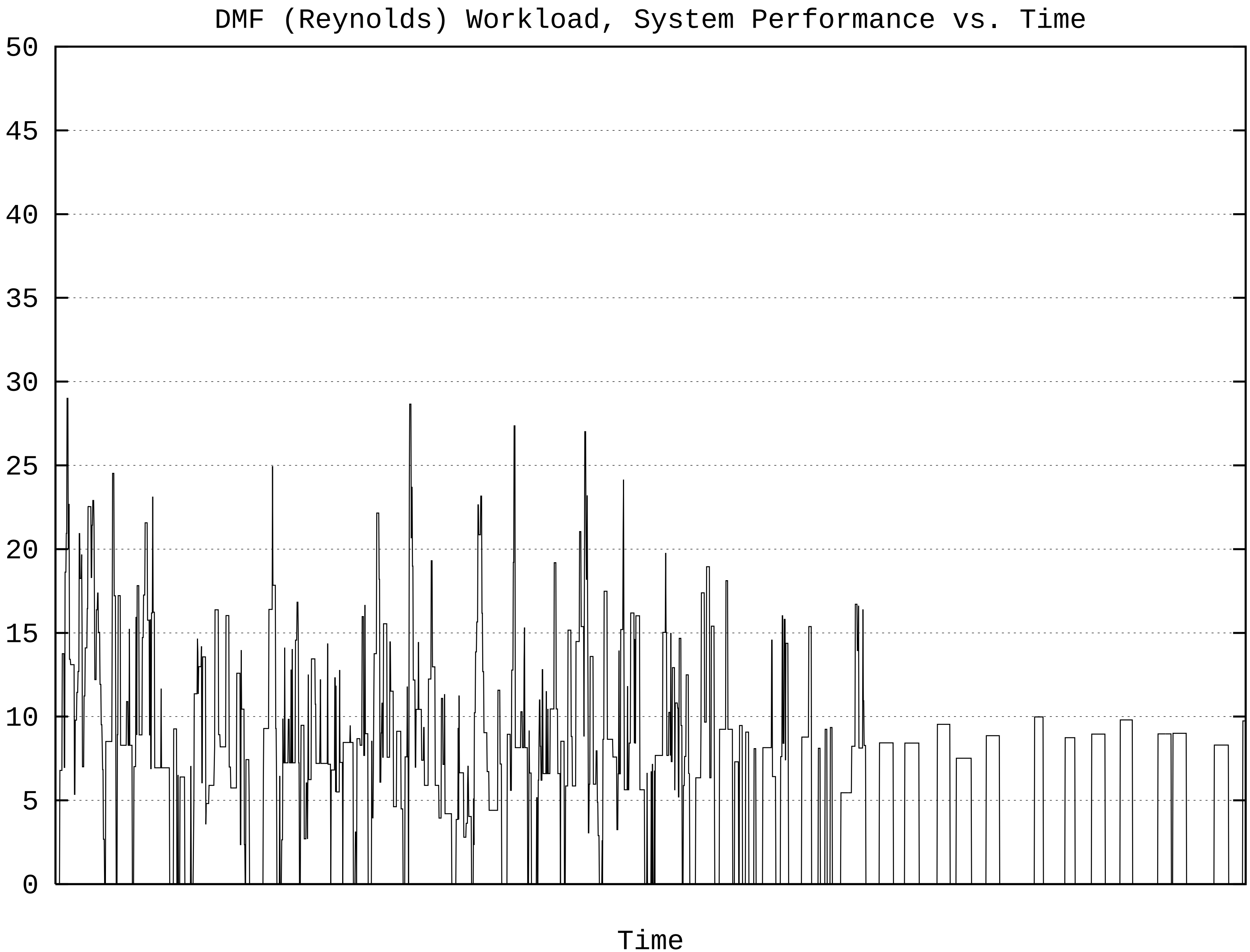
This report would not have been possible without the efforts of quite a number of people. I would like to thank the following people who played some part in data gathering or support with any of the packages:

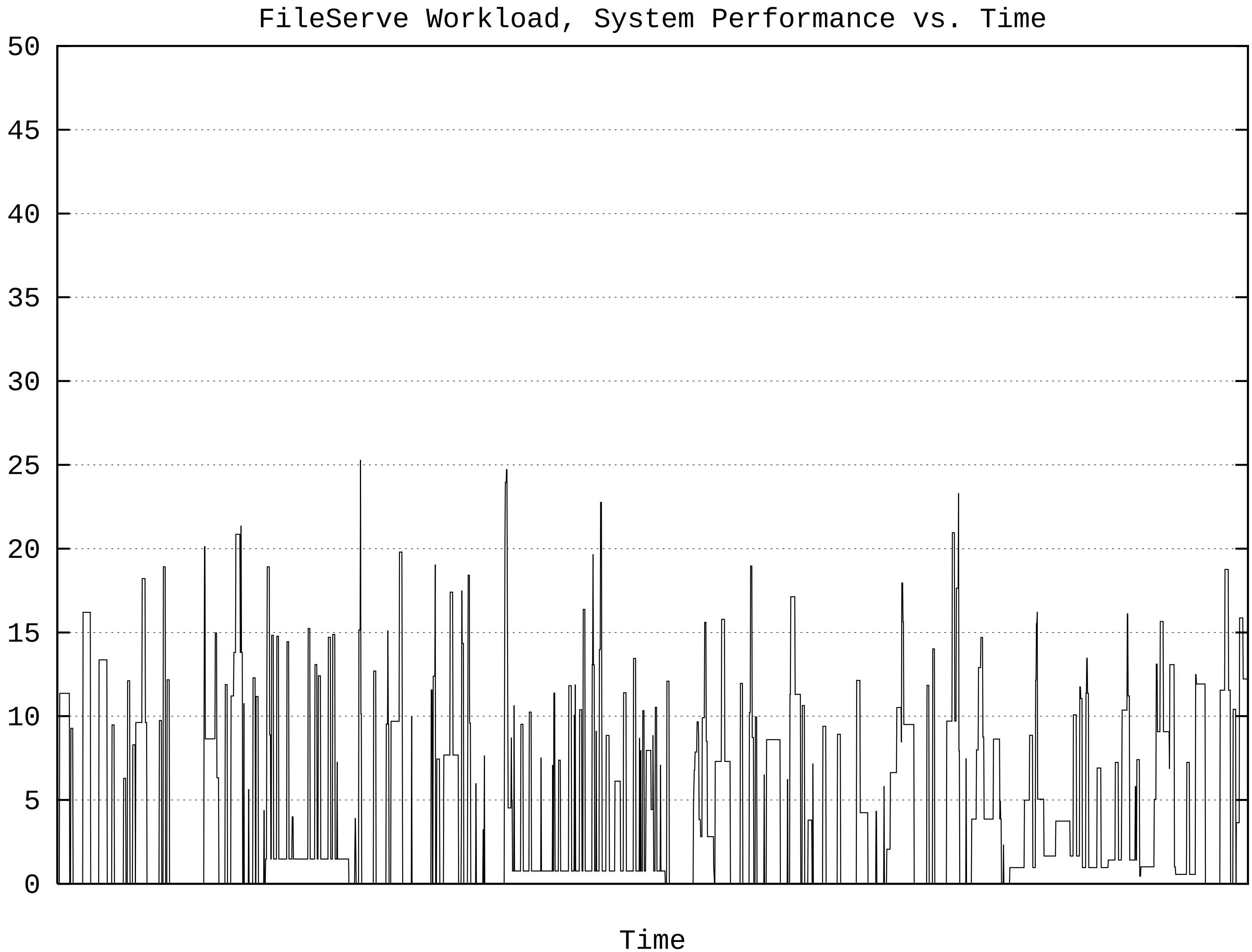
Alfred Nothaft for composing the workload code and running it
Eugene Miya with testing on various packages and platforms
Mike Selway and Eric Powell of E-Systems with FileServ
Larry Spoo and Sharon Bayne of Convex with UniTree
Nick Cardo, Winston Lew and Alan Powers of Sterling with DMF
Ruth Iverson, Alan Poston, Tom Proett, Bill Ross of CSC with NAStore

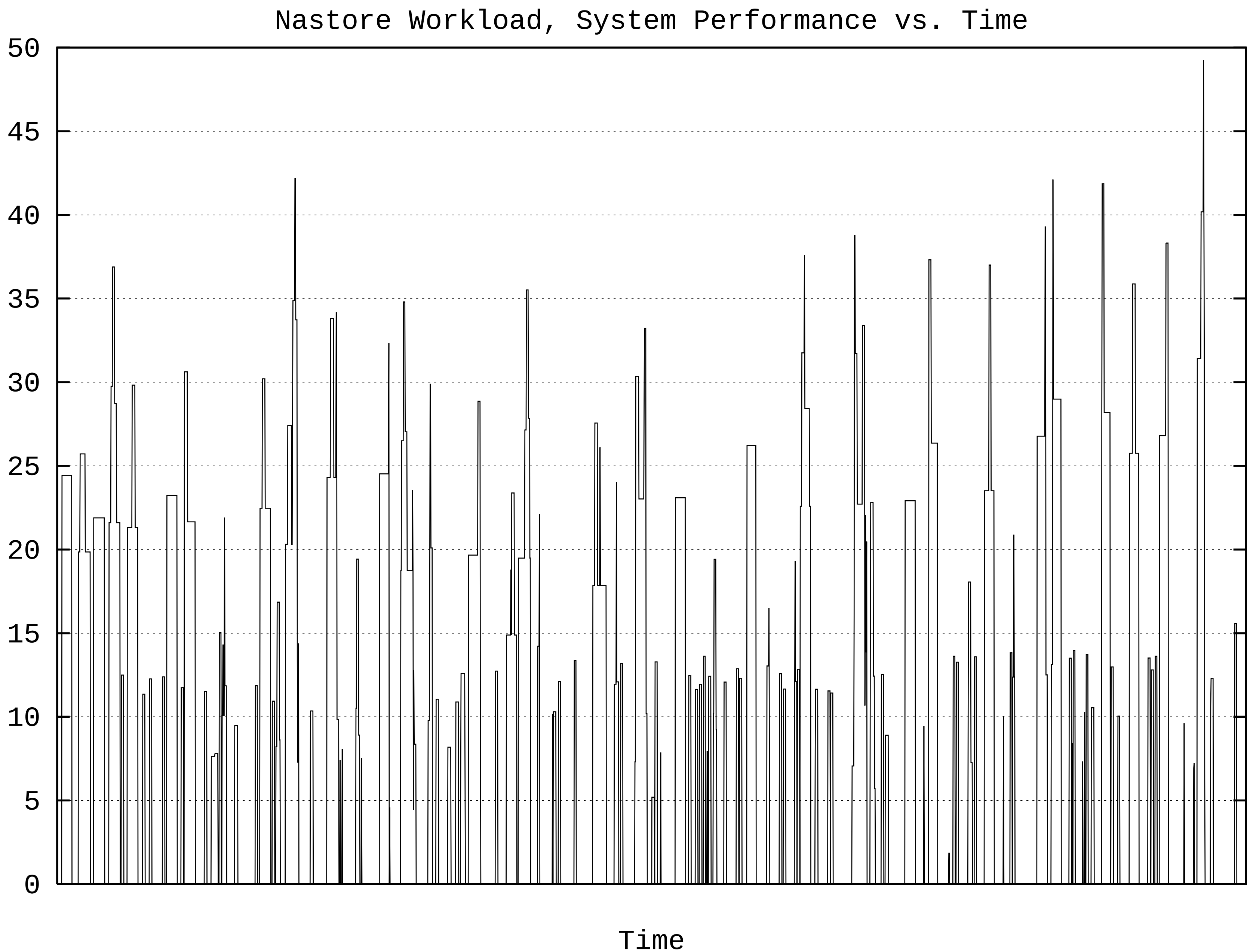
Contacts

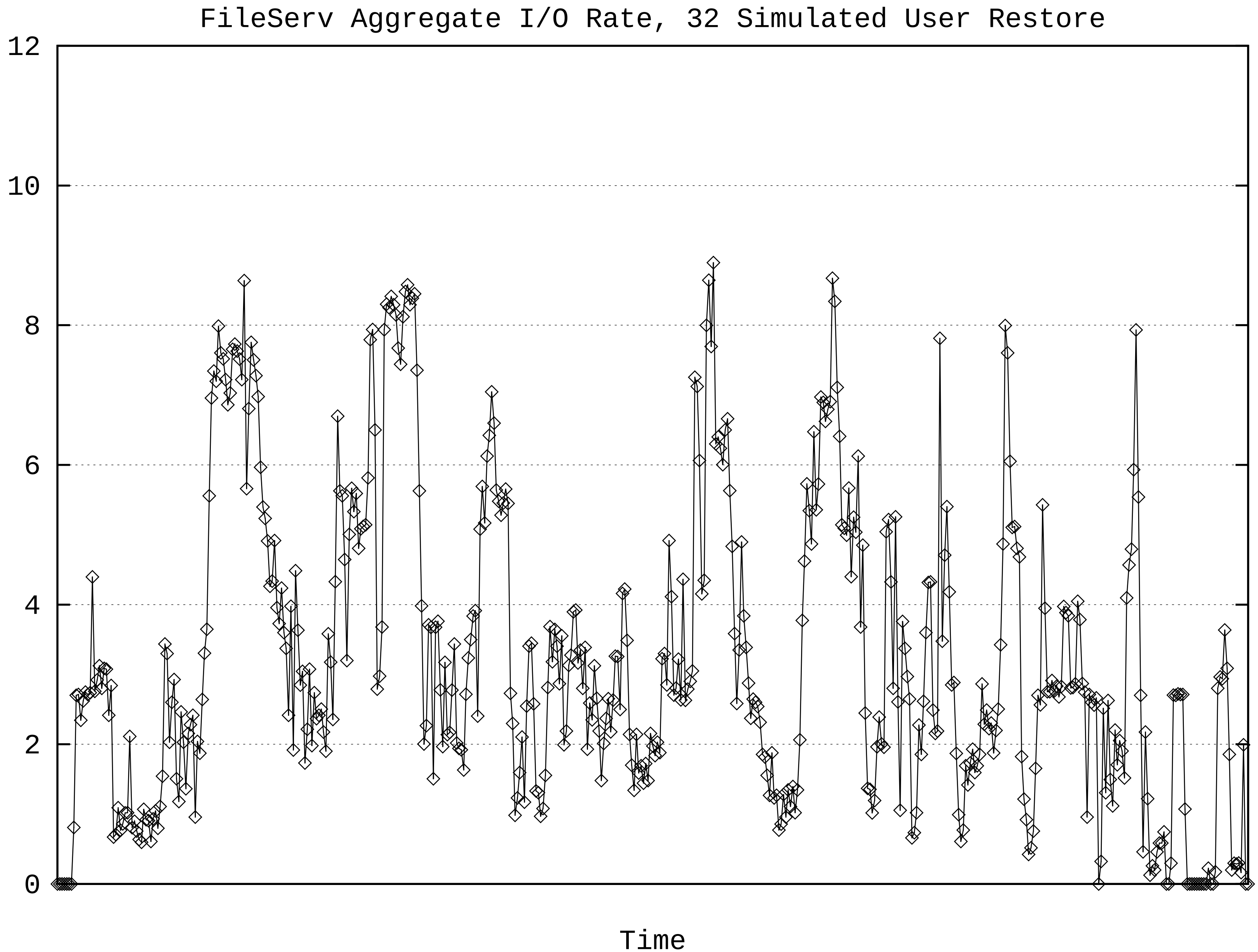
Several individuals are listed below for follow up or additional information.

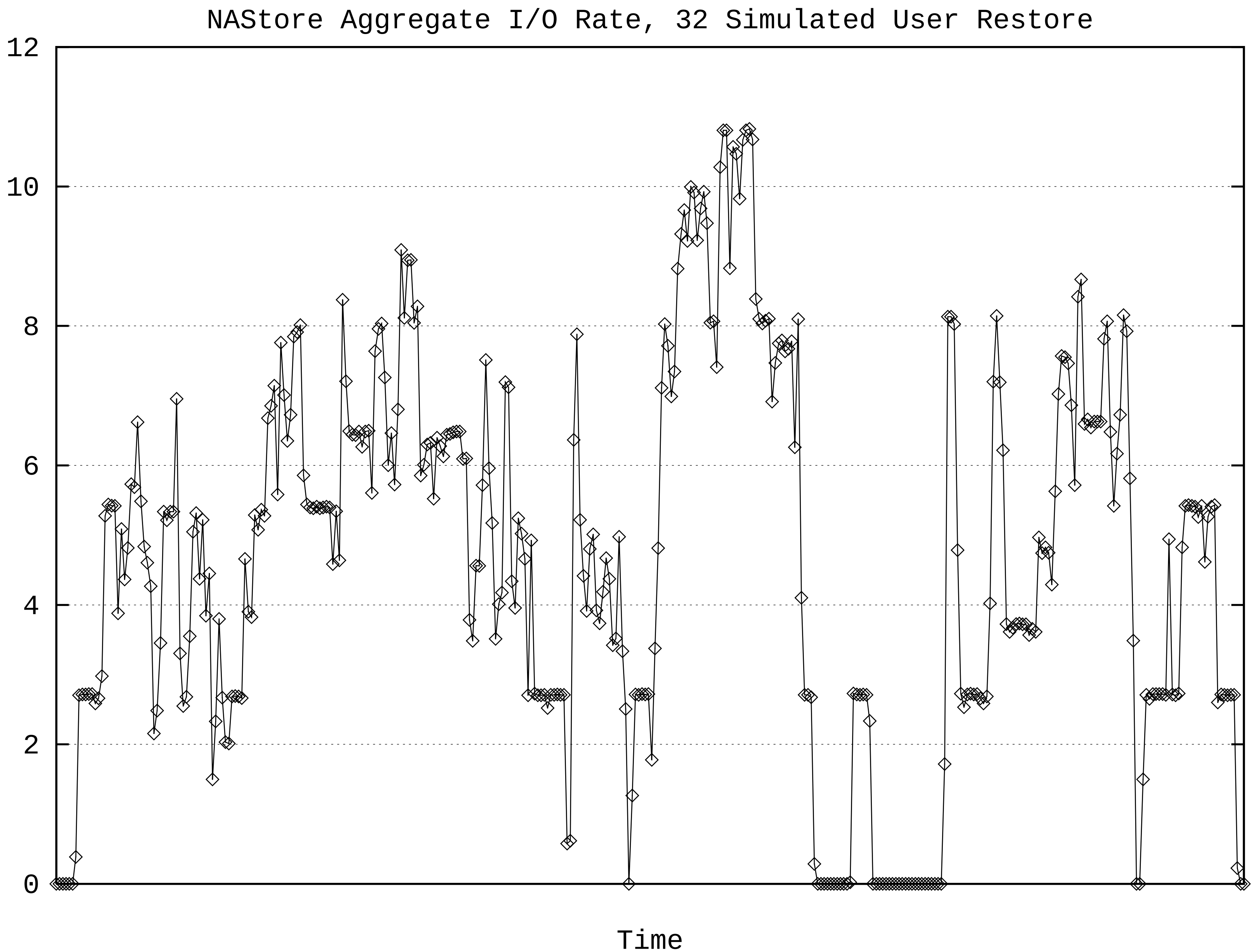
Data Migration Facility	Steve Banks Cray Research Inc. (408) 745-6466 banks@renaissance.cray.com	
FileServ	Steve Frazier E-Systems Corp. (404) 980-6685	John George Convex Computer Corp. (408) 453-5700 jgeorge@convex.com
NAStore	John Lekashman NASA Ames Research Center (415) 604-4359 lekash@nas.nasa.gov	Alan Poston CSC - NASA Ames (415) 604-4307 poston@nas.nasa.gov
UniTree	Max Morton Open Vision (510) 426-6452	John George Convex Computer Corp. (408) 453-5700 jgeorge@convex.com


DMF (Reynolds) Workload, System Performance vs. Time
50
45
40
35
30
25
20
15
10
5
0
Time


FileServe Workload, System Performance vs. Time
50
45
40
35
30
25
20
15
10
5
0
Time


Nastore Workload, System Performance vs. Time
50
45
40
35
30
25
20
15
10
5
0
Time


FileServ Aggregate I/O Rate, 32 Simulated User Restore
0
2
4
6
8
10
12
Time


NAStore Aggregate I/O Rate, 32 Simulated User Restore
0
2
4
6
8
10
12
Time