N94-33815

Storage System Architectures and Their Characteristics

Bryan M. Sarandrea

Advanced Archival Products, Inc.
6595 S. Dayton Street, Suite 1200
Greenwood Village, CO 80111
Phone: (303) 792-9700
Fax: (303) 792-2465
bryan@aap.com

1. Abstract

Not all users storage requirements call for 20 MBS data transfer rates, multi-tier file or data migration schemes or even automated retrieval of data. The number of available storage solutions reflects the broad range of user requirements. It is foolish to think that any one solution can address the complete range of requirements. For users with simple off-line storage requirements, the cost and complexity of high end solutions would provide no advantage over a more simple solution. The correct answer is to match the requirements of a particular storage need to the various attributes of the available solutions.

The goal of this paper is to introduce basic concepts of archiving and storage management in combination with the most common architectures and to provide a some insight to how these concepts and architectures address various storage problems. The intent is to provide potential consumers of storage technology with a framework within which to begin the hunt for a solution which meets their particular needs. This paper is NOT intended to be an exhaustive study or to address all possible solutions or new technologies, but is intended to be a more practical treatment of todays storage system alternatives.

Since most commercial storage systems today are built on Open Systems concepts, the majority of these solutions are hosted on the UNIX operating system. For this reason, some of the architectural issues discussed focus around specific UNIX architectural concepts. However, most of the architectures are operating system independent and the conclusions are applicable to such architectures on any operating system.

The problem:

The explosion of information storage requirements is being driven by more on-line data collection, data intensive applications such as imaging, government regulation over data availability and maintenance, and other needs. As this explosion takes place more and more users are realizing that disk (magnetic disk, DASD) is not an ideal solution for many reasons. Relative to other technologies, disk is more expensive, more prone to mechanical failure or data loss, requires increased administration, and other limitations.

Organizations are continually looking for solutions which meet their individual storage and retrieval needs which solve some of the problems associated with disk. However, disk has many advantages such as high transfer rates, random access, readily available file system interfaces and others. These advantages mean that disk almost invariably plays some role in meeting the storage requirements. The architectures discussed in this paper are all built around systems where disks play a primary role in the architecture. These are the most common solutions available and they leverage the good attributes of disk while utilizing alternative technologies to minimize the less attractive aspects of disk.

2. Criteria

This paper is forced to limit the scope of parameters discussed to only a few. In keeping with the stated goals of providing high level guidance, these parameters will be dealt with in general terms and as such should be useful as guidelines in evaluating or selecting technologies.

The parameters we will try to address are:

cost
performance
transparency of data access
administrative burden
distributed access

The conclusions reached on any given architecture will be arguable. For example, it is impossible to provide detailed performance information for tertiary storage systems. Nearly all systems discussed handle multi-user or multi-tasking environments where system throughput will vary by access patterns. A system with no contention and pre-staged media can provide nearly instantaneous response while the same system could require 2 minutes to begin providing data under other circumstances. Instead of providing details, an attempt will be made to present the issues. Only a detailed analysis or even evaluation period can determine actual performance under a given situation.

Costs will be given in relative terms along with some insight into the elements driving system pricing.

3. Terminology

The terms defined below are useful for understanding this paper. In no way do these definitions attempt to resolve the confusion over the correct use of these terms in the industry. Other industry terms will be introduced in context.

On-line:

Data which is on-line is accessible without human intervention.

Off-line:

Data which is off-line is inaccessible without human intervention.

Tertiary storage:

Storage which is accessible without human intervention but which is not RAM or directly addressable hard mounted media such as a magnetic disk drive. Tertiary storage devices in this paper will typically refer to removable media auto-changers such as optical disk jukeboxes or tape libraries.

Transparent Access:

Transparent access implies that a file can be accessed using the standard file system calls of the native operating system. Under transparent access, an application written to be capable of creating, reading, writing, etc. on the operating systems standard magnetic disk file system could perform these same functions on the tertiary storage without modification.

4. Non-Transparent Access Systems

While most storage solutions offered today are stressing transparent access, there are still many non-transparent access solutions on the market.

4.1 Backup Systems

Backup is only mentioned here for completeness. While backup systems are typically used in conjunction with any storage solution to protect from data loss in the event of failures in the storage system, in and of itself backup would not typically be considered a storage solution. Backup is, however, often used as simple archival mechanisms as referenced under "Simple Archive" below.

4.2 Simple Archival

For many applications and users, the usage characteristics of files (data sets) is either well known whereby a specific determination is made of which files should be archived, or specific user or application control over the process is desired. This environment could most easily be characterized as manual or demand archival. Typically the device used is a removable media device such as a cartridge tape drive which would require manual loading of the tape for retrieval. Some newer systems integrate the archival software with tertiary storage devices, providing unattended access to the files.

In some systems, certain data types are always considered archival data and are only retrieved to on-line devices when accessed. Not only is automated file/data management not necessary, it is often undesirable in these systems.

In order for these systems to operate effectively, mechanisms need to exist which support the storage and retrieval of these files to and from the archival system storage. Older systems accomplished this by providing specific archival functions which enable applications or users to copy file data from its current location to the archival system. Those systems typically would either provide an archival name or allow the user to specify an archival name. With these systems, an archived file could not be accessed in place by an application. Access required the file to be copied from the archival media to on-line storage. In addition, the archived files did not appear in the standard system file name space (file system). This required that the application or user remember the "archival name" of the file and learn new access methods.

Nearly all modern archival solutions will at a minimum, manage and track media volume allocation. This allows the non-transparent access simple archival process to be minimized to as few as two commands equivalent to "store the file" and "retrieve the file". Manual systems will typically then interface to a system administrator or the user to satisfy the media load function. The identification of the correct media is provided by the archival software which tracks the file to volume relationships.

There are many similarities between these archival solutions and backup solutions. The advent in recent years of simplified identification and retrieval of individual files from backup volumes, combined with the ability to specify individual files for backup, has all but duplicated the functionality previously provided by archival software. In fact, many backup vendors sell their solutions for both backup and archival purposes.

4.3 Automated Archival

Automated archival in these systems provides a function which automatically identifies which files should be archived. Typically such a function would be used to groom the on-line disk file systems for files which have not been accessed for long periods and/or which are large files. These procedures are often un-popular since the lack of transparent, automated access

combined with the lack of user control over what is archived can create various problems for both users and their applications.

4.4 Characteristics of Non-Transparent Access Systems

Clearly these solutions do not provide the benefits of transparent access to files. However, they can be very cost effective solutions when used with stand-alone tape drives or low cost autoloaders.

Performance of these systems ranges from essentially off-line to low performance since even automated systems will require restoration of the file from the archival media before access to any data can take place. These systems provide no ability to access the data without restoring the entire file, thus sufficient on-line storage capacity must exist in order to get access to the data. If sufficient space does not exist, it is up to the user or system administrator to move other files to create space or to find an alternative location for the file.

The administrative burden of these systems is placed on the user or application to determine which data should be archived. In addition, manual load system will require manual interaction to handle media load functions. Clearly data archived using these systems should be data which is very infrequently accessed.

5. Transparent Access Systems

The remainder of this paper discusses systems which attempt to provide transparent access to files held in tertiary storage. By transparent access, we mean that the access methods used to read or write these files is identical to those used in accessing the operating systems standard magnetic disk resident files. In order for this to happen, files on tertiary storage must be available in the file system name space and the standard operating system calls must be supported for access. There are numerous architectures available providing this functionality. Each architecture has been designed to provide certain features and functions. We will try to identify these as well as any trade-offs made by a specific implementation.

5.1 Virtual File Systems Interface (VFS)

In tertiary storage systems implemented on UNIX systems, the most common approach to achieving transparent access is to utilize the Virtual File System Switch as a mechanism for inserting new file system types or for extending the functionality of existing file systems. The VFS provides a standard interface to the internal OS calls dealing with file data. The purpose of this interface is to allow multiple file system types to co-exist within a UNIX kernel. Because the calls are well defined and adhered to by all underlying file systems, it is theoretically impossible for the users, applications, and network protocol packages to identify which file system is being accessed. This allows file systems to be added to the kernel which are designed to deal with the idiosyncrasies of the underlying hardware, transparently to the users.

FIGURE 1 shows how the VFS hides the file systems from the users, applications, and network protocols. The system calls to the operating system and the VFS call layer provide a set of standard interfaces which allow the introduction of new file system types for handling new functionality. The VFS is the facilitator which allows many of the architectures on the market today to provide the transparent access that is so desirable. In addition, since the underlying file systems are isolated from and invisible to the network protocols, systems implemented using this layer can be compatible with existing network software, allowing the vendors to utilize the standard protocol packages available from the OS and network software vendors. This minimizes the development efforts of the storage system vendors, reducing price, complexity, and cost.

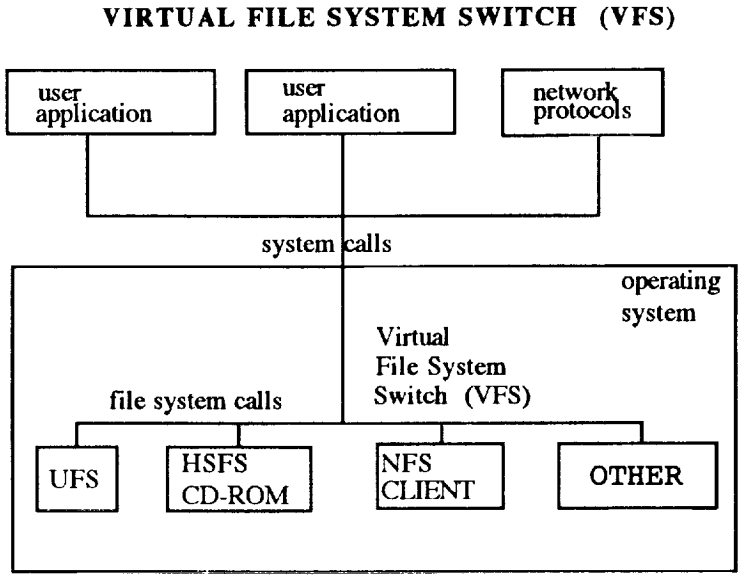

FIGURE 1

While the VFS depicted above is specific to UNIX, new operating system designs, including Microsoft's NT and most micro-kernels such as MACH are incorporating these concepts.

The transparent access systems discussed below can also be broken down into two classes of systems, those which try to provide automated data management and those which do not. Automated data management has come to be known as Hierarchical Storage Management (HSM). In HSM, the system attempts to manage the data, keeping it on the appropriate class of storage for its type and access patterns. A simplistic model of this is to move the least frequently accessed data from magnetic disk to tape as the magnetic disk fills up and more space is required. For vendors providing HSM, the goal is to keep the data management function hidden from the user. This is only possible if file retrieval is truly automated and the users access methods remain unchanged. Any solution will only be successful if the user population accepts it. Users are less likely to subvert the data management solutions if the solutions are designed well, make good selections, provide good performance, and do not interfere with their applications.

Systems which are not trying to attempt data management typically will utilize the native operating systems ability to specify the device as part of the file name specification. In UNIX this is done by allowing the device to be "mount"ed into the file system tree. From then on, data can be created or accessed on that device by using the path name which includes the mount point of the device. Through these methods, users and applications can control the device on which their data resides while maintaining system usability by having identical access methods to those of the primary data storage devices. While no industry accepted terminology exists for this class of tertiary storage systems, we will call them "direct access" systems in this discussion.

Even within these sub-classes there are a number of possible architectures which can be found. These architectures will be discussed below in order to see how they impact the criteria which we are trying to address here.

6. Direct Access Systems

These systems share many attributes with archival systems. The user or application of direct access systems makes the determination of which device(s) the data should reside on. Unlike

archival systems, however, the data files can be created directly on the tertiary storage (or copied there) using the standard file access methods of the host operating system. Some vendors call these systems "Direct Access Secondary Storage" since the data is created and accessed in place on direct access devices with random access characteristics.

Direct access systems exist which will cache the most frequently used data on high speed devices in order to provide better system performance. Such systems will typically allow repeated accesses to data to be satisfied from the cache. These systems differ from data management systems in that the primary copy of the data is on the specified tertiary storage device, while a copy of the data may exist on a high performance cache device only when it has been recently accessed. Data management systems, on the other hand, would have the primary copy on the high speed device and would only create a copy of the data on the tertiary storage device when the primary copy is about to be deleted. The sophisticated caching methods, of these direct access systems, blur the distinction between direct access and data management systems.

Direct access systems are particularly useful in environments where users or applications need control over data location or where the best location of data can be better determined by the user or application. Such situations may include:

Secure environments

The applications or users require deterministic location for data integrity, performance, transportability or other reasons.

Large file situations where the files or the aggregate size of the files being accessed at once do not fit on the available magnetic disk storage.

Situations where large data streams would force a data management system to purge all recently accessed data to handle the creation of the resulting large data files thus overriding the benefits of the data management algorithms.

System contains primarily small files where the overhead of tracking the data management function would mitigate the savings potential of tertiary storage.

Environments where the cost of storage is an overriding concern over the performance of repeated file access.

6.1 Server Based Direct Access Architectures

In a server based system, FIGURE 2, the tertiary storage is connected to a system which is responsible for allocation of drives, loading of media, and movement of data to/from the drives.

In these systems the data for all storage or access functions flows through the server architecture. Thus this architecture can create certain bottlenecks in the system. Careful consideration of the software architecture must be given in order to allow the simultaneous operation of the library and each of the drives to maximize total system throughput. In addition, the server must be closely matched to the performance requirements of the application and the hardware devices. A server handling 4 optical disks capable of 1 MBS each is clearly a different class of machine from the server which would be required to handle 4 tape drives capable of 20 MBS each.

In some environments all data processing, at least for a given set of data, is done on a single machine and no distributed access to the data is required. A direct access tertiary storage system for such an environment would be typically be configured similarly to a Server Based.

6.1.1 General Characteristics of Server Based Architectures

Server Based systems implemented as shown in FIGURE 3 have the benefit of providing the transparent access discussed in relationship with the VFS. Such systems can support local file access or remote file access using the OS vendors protocols or third party network protocols available for the host computer.

Some Server Based systems are available which provide transparent access by re-implementing specific network protocols. These implementations will typically limit the flexibility, since most only support a limited number of protocols.

A Server Based system will usually provide acceptable performance and functionality in any environment where file servers are now used to provide distributed access to a single shared name space. However, since the data stored on tertiary storage is often less frequently accessed or it is archival data, the applicability of this architecture can be far greater than those served by file servers alone. Often it is the data rate performance that prevent this type of architecture from being used. For example, if the system was required to support multiple simultaneous 10 MBS data streams to/from high performance helical scan tape drives, it may be difficult, or too expensive, to configure a single server to handle the requirements.

This architecture is particularly cost effective. Since direct access is being used, the large front end disk farm usually associated with HSM systems is avoided. Further, since the data rates of many of the devices used in tertiary storage systems is not particularly high, an inexpensive server class machine is usually quite effective as the underlying host hardware.

Administrative burden is variable with the specific implementation and is treated below where specific implementations are addressed.

The distributed access, server approach used, particularly for VFS based systems, allows these servers to be utilized in a very broad range of applications and heterogeneous network environments.

SERVER BASED DIRECT ACCI

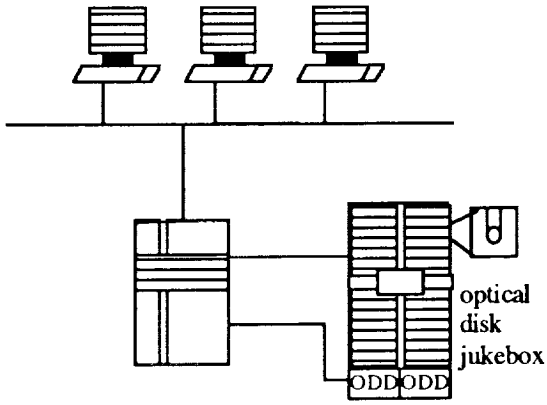

FIGURE 2

6.1.2 Direct Access Using a Magnetic Disk File System

Since a number of the devices used in removable media storage libraries have similar characteristics to magnetic disks, many vendors provide system solutions which use magnetic disk file systems to provide identical access methods for the tertiary storage. FIGURE 3, below,

shows a UNIX system architecture which implements this approach and a alternative file system approach. As shown in this optical disk jukebox system, the standard magnetic disk file system can be used to write the on-media format of the optical disks in the system. An additional device driver, the jukebox device driver, is layered between the file system and the device driver which operates the optical disks. This allows the block I/O requests to be intercepted so that the correct media for the given request is loaded before actually issuing the I/O request to the optical disks drive(s).

In particular, this method is used by a number of vendors for optical disk devices. The implementation of the device drivers is relatively simple to the implementation of a special file system and some compatibility with the standard OS's file system is achieved. However, this approach can not be used for media types which are not substantially the same as magnetic disk, e.g. WORM optical disks or tape devices since the file system is designed for random access of re-writable media.

6.1.2.1 Characteristics of Magnetic Disk File System Implementations

As mentioned above, one of the limitations of this approach is that certain types of devices and media types cannot be used. This stems from the fact that by re-using a file system written for magnetic disks, we are restricted to operating with device which can emulate the magnetic disk.

These systems also suffer from restrictions in the design of the file systems used as follows:

> The file system typically cannot cross media boundaries. This means that a minimum of one physical file system will exist per media volume. For two sided media, this will usually create two physical file systems per volume. This can be a burden for the system administrator who must now allocate space in much smaller fragments and potentially move data sets around the volumes as volume space becomes scarce. This will also typically limit the maximum size of files and will prevent files from being created in file systems and directories when media volumes become full.
>
> The file system does not know that the underlying media is removable and will schedule block I/O in random fashion, potentially causing thrashing of the jukebox. In addition, since the media is removable, standard file system sync mechanisms will not function correctly and system crashes may cause extensive file system damage.
>
> The caching mechanisms are also designed for non-removable media and may be inadequate for the long delays associated with loading and unloading media in a multi-user or multi-tasking environment.

These factors are typically weighed against system software cost. Systems which are implemented around custom file systems designed to solve these problems require significantly more R&D and typically carry a higher price.

Performance of these systems is typically acceptable if the above problems can be avoided. Thrashing in particular will drive system performance down. In this situation, the random block I/O patterns keep the jukebox constantly changing media volumes and very little I/O actually gets done.

6.1.2 Direct Access Using Alternative File Systems

As seen above, it is possible to use the embedded standard file system when the devices used have characteristics similar to a magnetic disk. But what if the vendor is interfacing non-standard devices or wishes to solve some of the problems pointed out in the discussions above. This is where new file systems have been introduced by several vendors. The VFS layer is called into play to allow a new file system to be added into the architecture where "alternative file system" is identified in FIGURE 3. This approach has allowed some well known file

systems to be added, as shown in FIGURE 1, which are considered to be a "standard part" of the UNIX kernel, but which in fact are really add on file systems for dealing with different device types or even networks. For example:

> High Sierra File System (HSFS). Most UNIX systems implement this CD-ROM file system under the VFS layer allowing the CD-ROMs to be mounted as a separate file system type.
>
> Network File System (NFS). As a network file system connecting two hosts, the NFS implementation consists of two pieces. The NFS client software which makes a remote file system appear as local is implemented under the VFS as a unique file system type. The NFS server software is implemented as "application code" which makes calls into the file system services of the VFS just as does any other protocol. This allows the server code to make any file system under the VFS umbrella available to the network.

These are just two examples of well known VFS implementations. Just as the HSFS implementation was designed to handle the differences associated with CD-ROM, third party tertiary storage solutions using this implementation are free to implement whatever mechanisms are appropriate for the device being integrated. The designers of these systems have much more flexibility in handling the idiosyncrasies of the various devices and can add custom scheduling algorithms, add additional caching, adjust the on-media formats, etc. as needed to provide good solutions for these devices. Packages are available to deal with re-writable and WORM optical disks as well as tape devices.

DIRECT ACCESS USING VFS

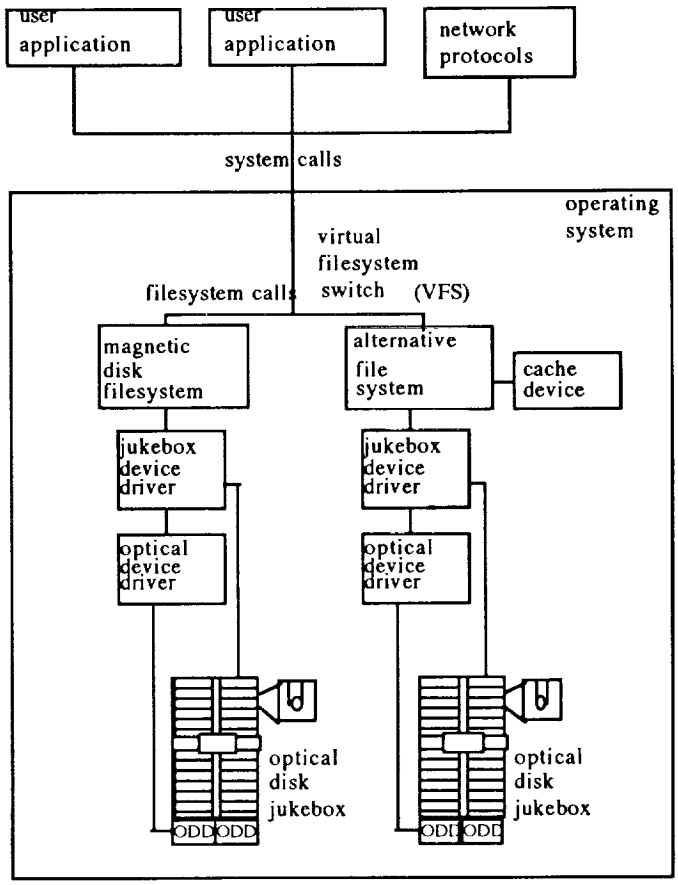

FIGURE 3

6.1.2.1 Characteristics of Alternative File Systems Implementations

The virtual file system approach has allowed vendors to directly address the unique characteristics of removable media devices while providing a standard file system type interface, making these devices appear as magnetic disks. The flexibility of this approach is born out by the availability of UNIX style file system interfaces for tape libraries as well as optical disk jukeboxes with WORM media.

The cost of these solutions is typically higher than others discussed thus far. This is due to the extensive software development effort required to write a complete UNIX file system. In addition, many of these file systems provide features not found in traditional file systems and invest a great deal in tuning the file system designs to obtain the best possible performance.

Additional caching is available in most of these systems. This caching allows the systems to provide a higher overall system performance, particularly in mutli-user and multi-task access environments. Here the cache can be used to implement read ahead and write behind algorithms which reduce wear on the auto changer hardware and increase overall system throughput by minimizing the number of volume exchanges required.

Some of these solutions also provide the ability to concatenate the media, thus providing a single file system view of the entire system. This can greatly alleviate the system administrators burden when dealing with space allocation. Single file system views also typically provide full file size support and eliminates the problems associated with full media volumes discussed above.

6.2 Direct Access with Network Attached or Switched Peripherals

In order to provide high data transfer rates and avoid the potential bottlenecks of server based systems, several vendors provide access to the tertiary storage peripherals through a high speed network, e.g. fibre optics, or high speed switch such as HPPI. Such a connection allows the data to be transferred direct from the data storage device direct to the requesting station. These systems are only employed in environments where distributed processing and very high performance distributed data access is required. There would be no need for such a system if all data was accessed or processed on the server.

WITCHED OR NETWORK ATTACHED PERIPHERALS

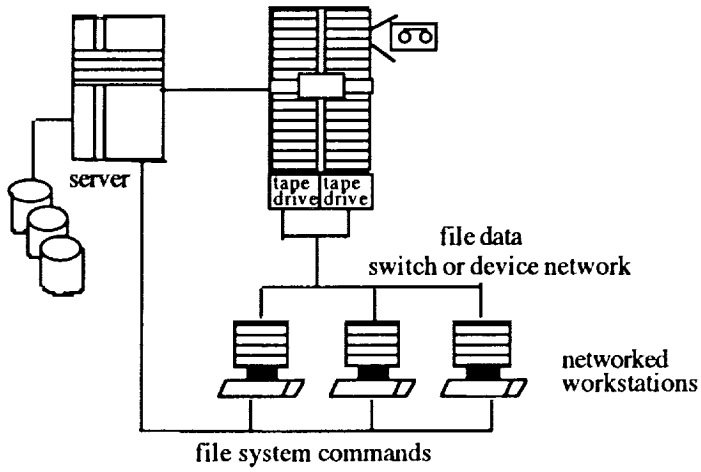

FIGURE 4

In these systems, in order to support shared access to the files stored on the tertiary storage, a central file system name space must still be maintained. This central processor is then also responsible for allocation of the drives, and operation of the library. In a system like this caching can get very complicated if shared simultaneous access across multiple processors is supported. In fact, the problem becomes identical to the caching problems addressed in such distributed file system implementations as DFS and NFS. Since the tertiary storage system has now become not only a storage subsystem but also a complete distributed file system implementation, the products tend to be incompatible with other network protocols and much more complex to administer.

If the files are not shared, each system on the network may maintain its own file system information and provide specific volume load requests to the library controller. Since each system tracks its own files, the files are not available to other users on the network. This model looks much like the server based implementation, without other network users, except for the fact that the library and devices become shared resources.

The benefits of these systems are that once the data is located, the transfer takes place directly between the peripheral device and the users processor. With a high speed connection, this configuration is much more capable of achieving the high transfer rates that are available on todays latest tape drives. Typically these systems are found handling the high speed data streams associated with todays super-computers. Since todays workstations are incapable of driving a 20 MBS tape drive at full speed, using these configurations on a network of workstations will not typically provide enough benefit to justify the increased complexity and cost.

6.2.1 Characteristics of Network/Switched Peripherals

Since direct peripheral device access and control is required to implement this architecture, custom software must be loaded onto both the server and each of the systems requiring data access. The architectures presented thus far have avoided this by utilizing exiting network solutions to provide the data access. This add a high degree of administration and support overhead to these systems. It also requires the implementation of custom data transfer protocols and control protocols by the software vendor.

Since the peripheral device control is turned over to the remote machine once the media volume is loaded, it is also not possible to share the peripherals. Once allocated, a drive will be dedicated to the remote machine until all of its I/O is complete. Therefore, although the I/O is most likely happening at a much higher rate, there is no ability to use caching as a solution to providing mutli-user access to the tertiary storage.

The extra software components, hardware components, and complexity of these systems relegates them to environments where very high performance transfers are required. This is usually associated with real time data streams or super computer centers in combination with very fast peripherals such as D1 or D2 tape drives. These same issues also mean that these systems are perhaps the most expensive to configure.

The requirement of dedicating access to drives and media volumes in this configuration makes comparison with cached systems difficult. Since contention can create significant delays in allocation of the drives or access to a particular media volume, the time saved in data transfer must more than compensate for the ability of a cached server type system to provide shared interleaved access and cached data. This implies that these systems also work best in environments with very large data sets.

7. Hierarchical Storage Management (HSM) Data Management

HSM or data management systems attempt to provide automated administration of data such that data is safeguarded against loss, shared access is provided where necessary, and large data repositories are managed in a performance/cost tradeoff fashion which minimizes the costs of

keeping the data available. Most HSM systems provide a view of the managed data that makes all data appear as though it is magnetic disk resident. Thus, the magnetic disk systems act as the primary storage device and, in fact, tertiary storage resident data which is accessed is typically moved back to the disk in order to provide access.

Unlike Direct Access systems, HSM systems will nearly always provide a first tier of storage which is a magnetic disk file system. The HSM then controls whether, and when, files are moved from the primary storage to tertiary storage. This is often referred to as "file migration", another name used to specify an HSM system. Users are given some level of control over the factors used to determine which files are migrated, however, it is the migration software which will perform the move automatically. Typically files are moved from the primary magnetic disk storage to the tertiary storage in order to maintain free space for new files on the magnetic disks. When a file which has been moved is accessed, the file is moved back to the magnetic disk for access.

Much like direct access systems, HSM systems can be configured in a large variety of architectures to meet various needs. A typical HSM system today would have only two tiers of storage, being magnetic disk and an optical disk or tape library. However, more vendors are now offering multi-tier systems with three or more tiers of storage. For example a system might move a file from magnetic disk to an optical disk jukebox and then if still not accessed after an additional time period move the data again to a tape library.

7.0.1 General Characteristics of HSM Systems

HSM systems can be very complex. The analysis of HSM systems requires an in-depth look at the architecture used and some general issues concerning HSM and how the systems and data are used. However, it can be generally stated that HSM systems can provide significant savings in storage costs when used in the right environment. In addition, the automation of the data management function frees system administrators from the chores of managing disk space and data archives.

Much like Direct Access systems, the performance, flexibility, network compatibility and other issues are determined by the specific packages architecture. We will try to address these below.

In evaluating HSM systems there are several hidden aspects to be aware of. First, HSM performs its function by being aware of which files have been used recently and which have not. For this reason it is imperative that nothing on the system destroys this information or the HSM system will not operate correctly. However, it is common for backup software to routinely go through file systems "looking" at if not accessing all files. This can cause the HSM system to see all files as recently accessed.

Some vendors solve this problem by providing special local and client/server backup packages which can work with their HSM solutions to solve the problem. With these vendors, the customer may be locked in to that backup solution, in which case it is imperative that the backup package also be evaluated as to functionality, cost, performance, etc. since it is the only backup package which can be used.

Other vendors solve this problem by making their HSM software compatible with third party backup packages. This is usually done by providing a framework within which to run the third party package such that it does not destroy the vital information needed by the HSM system. With such a system, the entire market of available backup packages can be used, leaving more flexibility for the consumer.

In addition, there are other backup issues;

How do you prevent the backup package from migrating in all files on a "full" backup.

How do you restore a migrated file.

What happens if you restore an old backup tape on a client. Are the references to migrated files still accurate.

Select an HSM package which has addressed these issues in its design.

HSM systems also impose additional storage overhead on files they migrate, and sometimes even on files which are not migrated. An HSM system which puts a file on tertiary storage will potentially require the following storage:

An inode to track the migrated file,
A data block for information about the file,
A local database entry for additional information,
A database entry on the tertiary storage system to track the files location,
An inode on the tertiary storage system for identifying the data file,
and of course the data file itself.

Look at how much storage is used for migrated files and how much disk is required just to set up the file migration. For most systems it is not economical to migrate small files.

It is also important to determine whether there are functions on your system which will defeat the data management system. For example, does anything periodically sift through your file systems reading files which would cause all files to migrate back. If users use the "file" command, does the HSM system allow for that without causing all files to migrate in.

For systems that are composed primarily of very large files, HSM may also be a bad choice. If each file access causes some other file to migrate out to make room to migrate this file in, then a direct access system may be a better choice. For example, a 2 GB disk HSM system where all files are 200 MB and which has 11 simultaneous users will thrash the system trying to get all 11 200 MB files read in at the same time.

The following are some features to look for in HSM systems.

File data is available as it is migrated in, as opposed to waiting for the complete file to be resident on the magnetic disk.

It is possible to determine if a file is resident or non-resident.

Manual migrate in & out is possible.

The system supports high and low watermarks for controlling available disk space.

The system supports pre-migrated files which can have their space freed up quickly.

The system catches "out of space" and begins "demand migration" creating space instead of returning an error.

Files migrate in both directions, some system migrate out and only provide Direct Access to these files after being migrated.

The software can manage pre-existing file systems. File systems are compatible with the native operating systems file system software.

7.0.2 VFS Versus Non-VFS HSM Implementations

The implementation approach taken implies a great deal about the functionality of the package. In HSM systems we find, as we did in the Direct Access discussions, that some

implementations utilize a VFS approach while others re-implement specific network protocols. These later system suffer from the same flexibility issues, in that all protocols or access methods supported must be re-implemented and cannot build on the existing system capabilities. These systems may also suffer from performance degradations associated with trying to implement file system type functions as application level software. In the UNIX community, the non-VFS implementations are nearly always Server Based HSM only and do not support the Client/Server HSM model. One benefit of non-VFS systems is that the opportunity exists to use mechanisms other than traditional inodes to track files. This could alleviate some of the duplicated overheads and prevent file systems from becoming full by running out of inodes.

Those systems implemented as VFS code will typically either overlay data management onto an existing file system or insert a file system with data management capabilities. The former approach allows the use of the vendors standard magnetic disk file systems. By layering the data management function over the existing file system, it is possible to maintain complete compatibility with all file system utilities and other software.

The approach of adding a unique file system, whether through the VFS or not, means that the resulting on-media formats will be incompatible with the vendors and a complete set of separate file system maintenance utilities will have to be used.

7.1 Server Based HSM

In a server based HSM configuration, FIGURE 5, all managed storage is centralized on a file server. These systems utilize file transfer protocols such as FTP or distributed file systems such as NFS or DFS to provide decentralized file access services. Any data not resident on the server cannot be managed by the HSM solution.

For network environments where centralized storage is already in use with network file servers and where the network clients are usually diskless, this type of system fits well. The HSM system will continue to manage storage as a central resource and the tertiary storage is shared as a function of data management of the shared disk farm.

7.1.1 CHARACTERISTICS OF SERVER BASED HSM:

The performance of Server HSM varies with the implementation. A good implementation should have a minimal impact on the performance of the magnetic disk file system on non-migrated files. In fact, the managed file system should perform with less than 1 to 2 % degradation.

It is usually possible to add Server Based HSM to an existing file server and obtain acceptable performance, assuming that the performance was already acceptable. These systems should place a minimal amount of load on a functioning system.

The cost of Server Based HSM should be close to that of Server Based Direct Access systems of the VFS style. The level of complexity of the two products is roughly equivalent. However, prices will reflect the type of tertiary devices being supported as well as the type of server.

The administrative load of these systems should be quite low. In fact, most vendors claim the reduction in system administrator load as one of the cost justifications of HSM systems. However, there is certainly an initial setup and learning curve on systems this complex.

SERVER BASED HSM

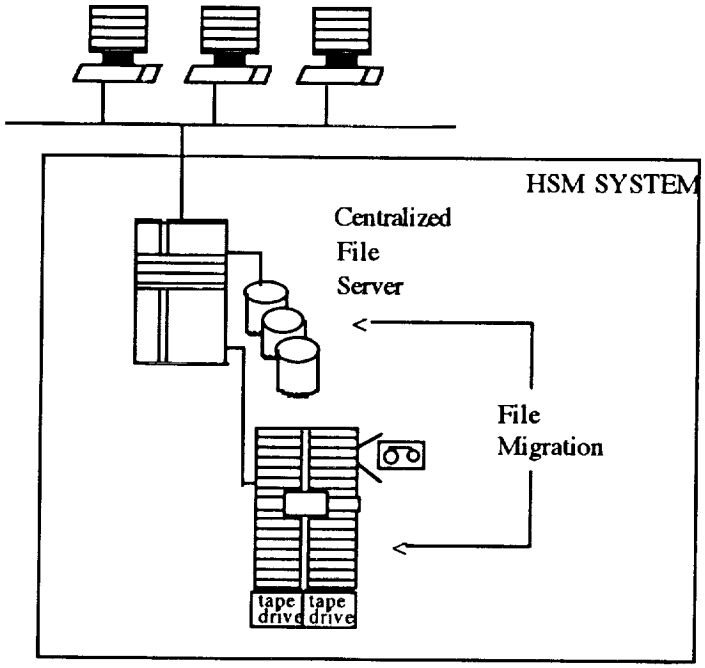

FIGURE 5

7.2 CLIENT - SERVER HSM:

Client Server HSM, FIGURE 5, provides the ability to manage data at both the client level as well as at a server level. By creating a client-server network interface, the client HSM software can manage local data repositories and utilize the storage capacity of centralized tertiary storage devices. This allows the most recently accessed file data to be moved close to the client by moving it to the clients local disk while sharing the high capacity lower cost storage devices for the data which has been migrated from the primary storage.

CLIENT - SERVER HSM

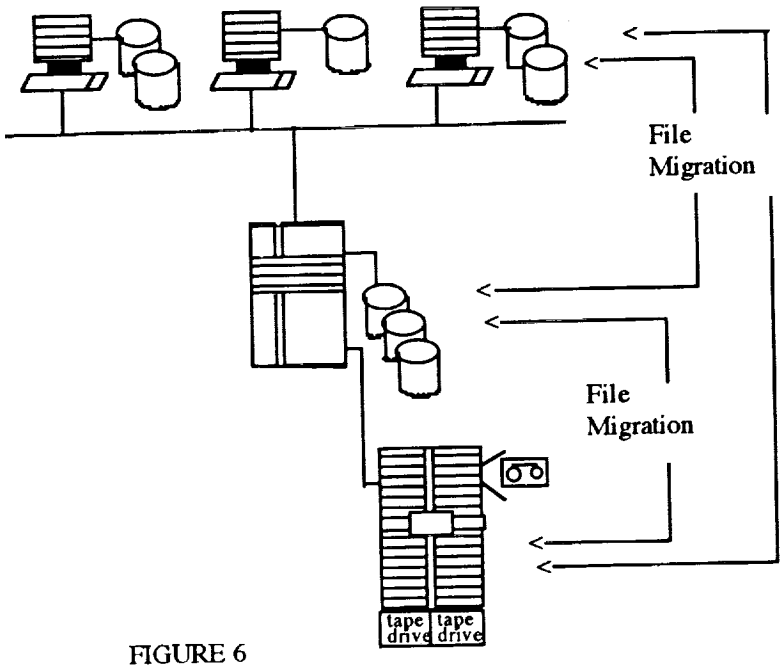

FIGURE 6

The primary consideration when deciding on a Server Based or Client - Server configuration is the impact on accessing the file systems. Most HSM solutions, in order to provide high performance in combination with transparent access, keep some type of reference to the migrated files on the disk, and the file system, where the file was originally created. When using Client - Server configurations this implies that in order to get distributed access to each of the diskfull clients files, their individual disk file systems must be "exported" to be made available for access from other clients. This in effect makes each client a server as well if shared access to each clients files is required. However, for environments where shared access is not required, this method can provide data management to each diskfull machine, high local file system performance, and access to shared inexpensive tertiary storage.

Since the Client - Server architecture typically supports the client software running on the server as well, data management of the servers disks is also provided as depicted in FIGURE 6. This allows shared access to the servers disk file systems combined with data management. Some systems will also allow the servers disks to be the first "tier" of storage used for migrating files from the clients. This provides two tiers of high performance storage before files migrate to tertiary storage.

It is easy to see the flexibility of Client - Server architectures. If the workstations shown in FIGURE 6 were configured as servers, and each was serving a network of diskless workstations, we would have a configuration which would support a number of decentralized department or work group servers where each server's local disks had HSM software managing the local data and moving less frequently used data to a shared tertiary storage system.

7.2.1 Characteristics of Server Based HSM

When the client software is used in conjunction with the server based tertiary storage components, this system is identical to a Server Based HSM.

For true Client/Server configurations, this system provides the performance benefits of local disk file systems combined with the benefits of HSM. The client software can also be written to

be consistent with the native operating systems capabilities. On clients with such features as access control lists (ACLs) or typed files, the client software can retain such functionality while implementing the data management function. Under Server implementations, the clients may only see those file system characteristics available on the Server.

The single biggest factors affecting the performance of access to migrated files will be the network interface used to send commands and data between the components and the performance of the tertiary storage system. The network interface should be flexible enough to support a wide range of client implementations. No standard exist at this time for these interfaces and vendors have all gone their own way. However, standards are expected in the future, in the hope that the components from various vendors will be able to work together.

System pricing for client server varies widely. Some vendors price by client, others by GB of tertiary storage and yet others by GB of disk managed. (I'm sure I missed some.) These systems will typically cost more for equivalent servers and storage devices because of the higher complexity involved in supporting remote clients. However, the current market forces have created a broader selection of these systems than high end Server Based HSM systems and competition seems to be quickly driving prices lower.

7.3 Multi-Tier HSM

In multi-tier configurations, data can be moved from the primary storage through multiple levels of tertiary storage according the migration parameters. The first tier might be an optical disk jukebox providing fast random access, fast media load times and a lower cost per MB than the primary storage. Once resident on the optical disks, data may later become eligible to be moved again to less available storage. The next tier might be a tape library system. This tier would feature a very low cost per MB but at the expense of much longer file retrieval times should the data be requested.

Some systems will have off-line media as the lowest level of managed storage. In such a system, data moved to the off-line media will require operator assisted media loads for any requested data. While this is certainly the least expensive, it does require an operator, and data access is no longer automated.

7.3.1 Characteristics of Multi-Tier HSM

Multi-tier HSM provides a very sophisticated ability to tune the cost of data storage to the access patterns of the data itself. However, setting up, managing, and tuning such systems can be a significant effort. In addition, the cost of these systems is higher due to the added functionality and flexibility. Look carefully to see that the added savings on storage capacity of the lowest tier(s) justifies the administrative effort and initial purchase differentials.

These systems should be capable of moving data from any tier direct to the user systems primary storage. The performance impact of having to move the file through multiple tiers to retrieve the file would create severe performance penalties.

7.4 Distributed HSM Architectures

While we have demonstrated some of the configurations available in HSM solutions from a file system perspective, we have yet to discuss the flexibility some vendors offer on the tertiary storage side. Whether or not the system consists of multiple tertiary storage devices as discussed in the mutli-tier configuration above, it may be desirable for the tertiary storage to be remote from the primary storage or from other tertiary storage components. FIGURE 7 shows a distributed storage HSM configuration.

The ability to distribute the storage elements of an HSM system can spread the processing, data transfer and the network loads. This type of capability creates a more scalable system and one

where additional capacity can be added in smaller increments. The power of any given component does not need to be scaled to match the total systems capacity.

DISTRIBUTED MULTI-TIER HSM

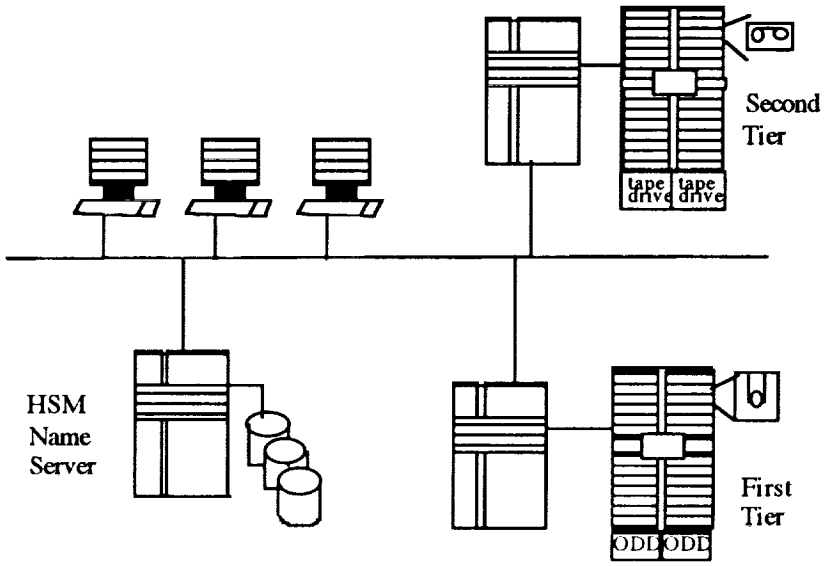

FIGURE 7

8. Combined Architectures

It is possible to configure many different system architectures from the individual concepts presented in this paper. For example, a multi-tier HSM system consisting of a RAID as the primary storage device coupled with an optical jukebox could be used in a network attached peripheral system in order to provide the high performance advantages of both architectures and the central name space attributes of a server based implementation.

9. Summary

It is not possible to address all of the possible architectures. I hope that this paper has, however, introduced the basic architectural concepts.

By having the basic understanding of the various type of implementation available, it is possible to analyze ones requirements with respect to performance, cost, transparency, complexity, etc. and to be able to evaluate the competitive offerings on the market with a more objective set of criteria. No one vendor offers all of the configurations discussed, it is therefore important to understand the technology to determine if a given vendor's proposal meets your requirements.