Beyond backup toward storage management

by M. Kaczmarski
T. Jiang
D. A. Pease

The IBM Tivoli Storage Manager, a client/server product providing backup, archive, and space management functions in heterogeneous distributed environments, performs extensive storage management after client data have reached the server. Beyond minimizing the amount of data that a client needs to send on successive backup operations, Tivoli Storage Manager optimizes data placement for disaster recovery, for restore operations, and for fault tolerant access. It also adapts to changes in device technology. The original design points of the product in research have been expanded to provide a comprehensive set of functions that not only facilitate backup but also support content managers and deep storage applications. The design points and functions are described in this paper.

The proliferation of distributed computing and Internet usage together with continually falling storage prices, greater disk capacities, and tremendous data growth, challenge storage administrators to adequately provide nonintrusive backup and proper recovery of data. Enterprise computer system data protection is subject to operational demands that are driven by varying business requirements and continual advancements in storage technology. A number of factors lead to inherent complexity in the seemingly mundane task of recovering data.

All data are not the same. Information that supports important business processes may be distributed across multiple applications, databases, file systems, and hosts—intermixed with data that are easily recreated and clearly less important to the enterprise. Data elements that share the same file system or host containers have varying levels of importance, depending upon the applications they support or the rate at which they are changed. The management complexity in dealing with this environment often leads to inefficient backup practices because all data are treated at the level required for the most important elements. Differentiated data requirements need to be recognized and managed in an automated way to control network and media resources and the administrative expense involved in backup processing.

Disaster recovery can involve many dimensions, ranging from simple user errors that cause the loss of word-processing files or spreadsheets, to hard drive failures that impact entire file systems or databases, to tragic losses of buildings and assets that include large-scale information technology infrastructure and storage subsystems. Backup management is usually tuned to one of these possible disaster situations at the expense of efficient recovery should the other occur. Automated management of backup storage helps administrators move from operational monitoring to developing strategies and practices for handling each of these situations.

0018-8670/03/$5.00 © 2003 IBM

New storage devices and technology come and go, creating a struggle in migrating massive amounts of data from one device type to another while maintaining application availability. Failure to keep up with advances in storage technology can expose an enterprise to long-term support problems should its existing devices fail. These exposures directly affect the ability of the organization to provide proper data protection.

These factors provide the motivation to go beyond backup processing to a more comprehensive storage management paradigm—one that controls costs and automates common tasks by providing a means to map the underlying storage to the requirements of a business.

The IBM Workstation Data Save Facility (WDSF) was developed in the late 1980s at the IBM Almaden Research Center to meet customer requirements for distributed network backup. The product underwent significant redevelopment and became the ADSTAR Distributed Storage Manager (ADSM) in 1993. It was later renamed the Tivoli Storage Manager (TSM). The need for network backup emerged from distributed client/server computing with the proliferation of personal computers and workstations. The goal was to centralize the protection of distributed data in an environment where information assets were no longer restricted to controlled mainframe computer environments. Backing up individual computers to locally attached devices was, and still is, costly and error-prone and often did not meet requirements for disaster recovery. With TSM, clients can back up their data to central servers. The servers store the data on a variety of media and track the location of the data for later retrieval.

Tivoli Storage Manager is a client/server application that provides backup and recovery operations, archive and retrieve operations, hierarchical storage management (HSM), and disaster recovery planning across heterogeneous client hosts and centralized storage management servers. Support has been made available for over 15 client platforms, 7 server platforms, and over 400 different storage devices as illustrated in Figure 1. Specialized clients, represented as green cylinders in the figure, supply backup and restore or archive and retrieve support for specific applications such as DB2* (Database 2*), Lotus Domino*, Microsoft Exchange, and SAP R/3**. A client application programming interface (API) is also provided for those customers or business partners who wish to store and retrieve data directly into TSM. Data are transferred between the clients and the TSM server over the communications network or across a storage area network. A Web-based administrative interface and a coordinated distribution of shared management policy provide a common control point for multiple storage management server instances.

Advanced design points were established for TSM in an environment where many network backup applications evolved from simple single-host backup utilities. The primary influences were the need to deal with relatively slow network speeds, scalability in handling a large number of clients and platforms, and the desire to manage data with policy constructs borrowed from systems-managed storage (SMS) of mainframe computers. This paper describes these design points with a survey of functions and features that demonstrate storage management capabilities in TSM. Today, these capabilities provide management for active data as well as backup copies.

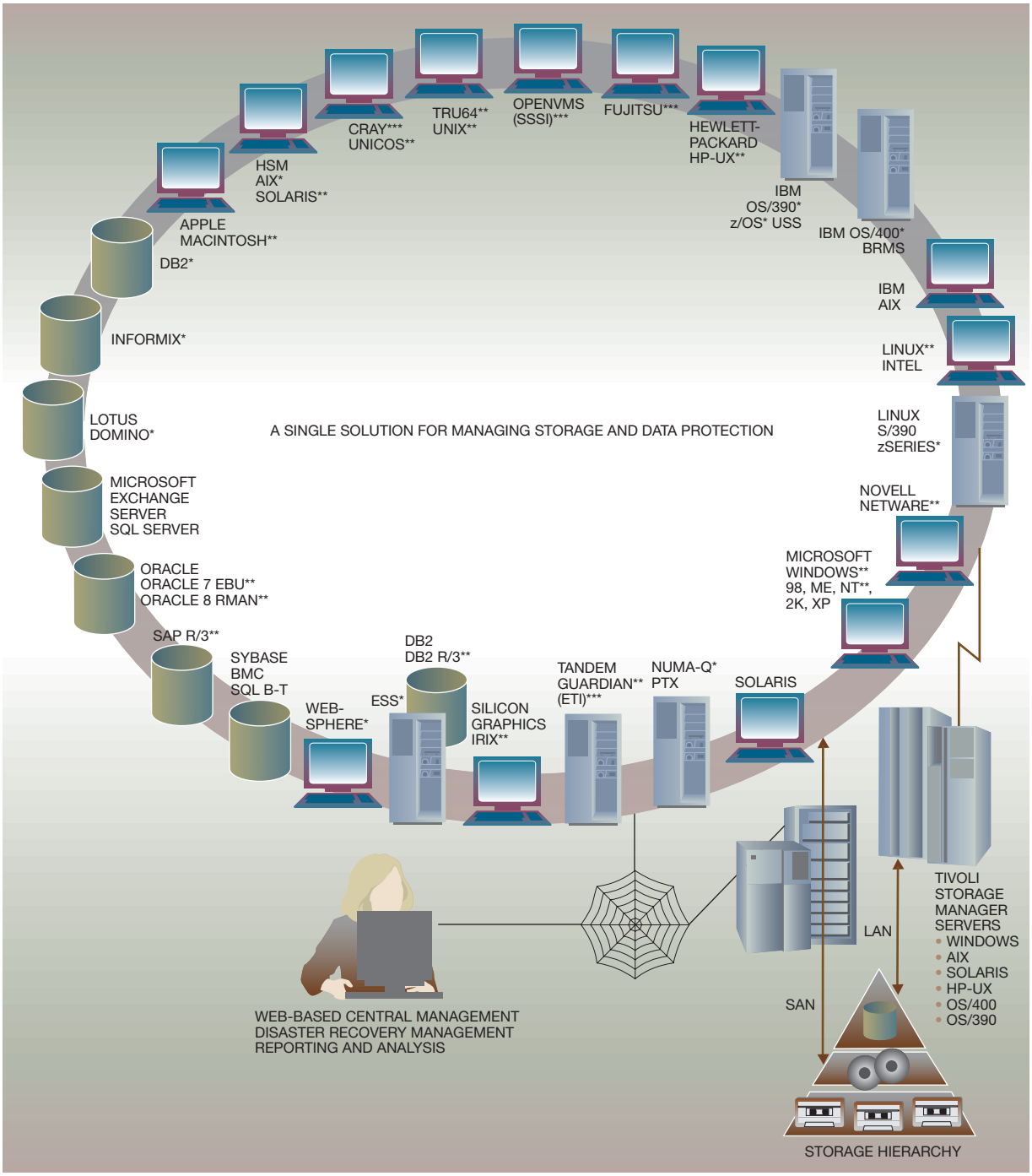
Minimizing network traffic: Progressive incremental backup

The rate of growth in the amount of data stored in computer systems has traditionally outpaced growth in network bandwidth. The use of traditional communication lines for backup processing suggests that indiscriminate backup loads can clog or disable communication networks. Control is needed to determine when backup processing takes place and to ensure that backup communications traffic is minimized when it does occur.

The goal behind the progressive incremental backup of TSM is that once backed up, unchanged data should never have to be resent (or rebacked up) to the server. Most methodologies for open systems have been developed to optimize data placement for tape media reuse and not to minimize data transfer and optimize scalability in a client/server environment.

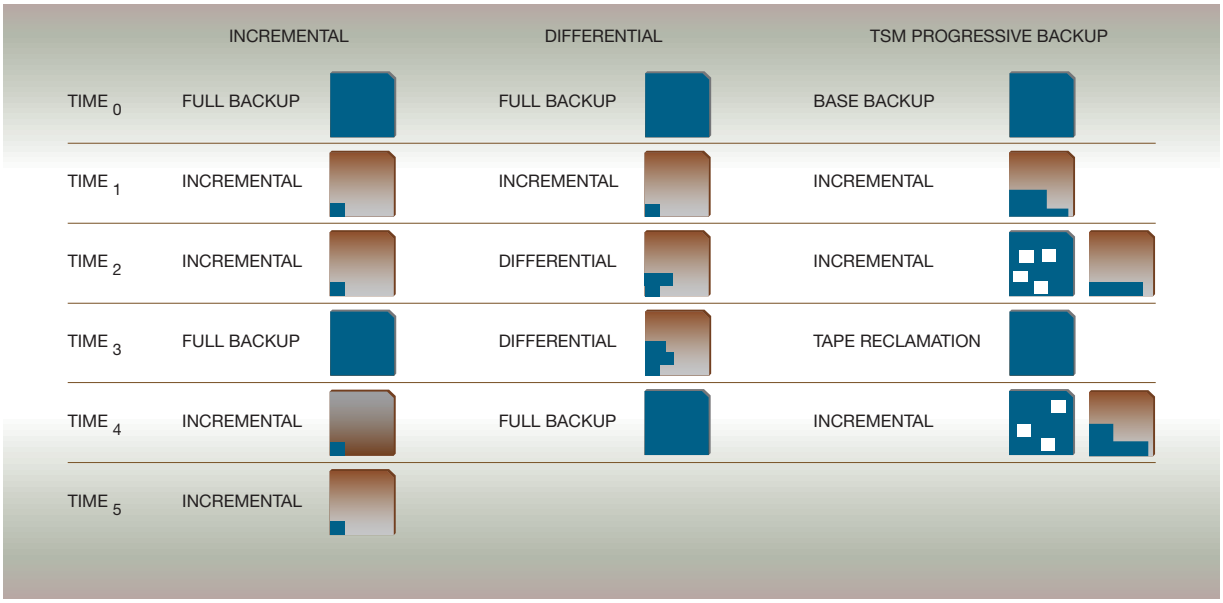
The use of tape as a backup medium requires that periodic consolidation of data be performed. Tape differs from disk in that once a tape is initialized (or "labeled"), data can only be appended to the tape until it is full, after which time it must be reinitialized before it can be reused. Tape consolidation is required because files change at differing rates; subsequent backup operations that only copy changed files will store the new copies on additional tapes. On existing tapes this leaves logical "holes" occupied by file copies that are no longer current. Over time, these operations fragment the regions of use-

Figure 1 Tivoli Storage Manager topology and supported platforms



* Trademark or registered trademark of International Business Machines Corporation.
** Trademark or registered trademark of their respective companies.
*** Third-Party Vendor using TSM API or Service offering. TSM clients supported by V5.1 server, V4.2 and V5.1 clients.
Source: F. Saldana, "Tivoli Storage Manager Overview," *SHARE Proceedings*, Nashville, TN (March 3–8, 2002).

Figure 2 A comparison of different backup techniques



Source: F. Saldana, "Tivoli Storage Manager Overview," *SHARE Proceedings*, Nashville, TN (March 3–8, 2002).

ful data on the original tape volumes and spread backup data across many tapes, requiring more time and mount activity to perform a restore operation and requiring more media for storing backup data. Traditional *incremental* or *differential* backup methods achieve consolidation by periodically performing a new full backup of all of the data to a fresh tape. This method frees the original tapes so that they can be reused but has the side effect of resending all (unchanged) data across the network. This method not only wastes network bandwidth, processing cycles, and tapes, but it also leads to having to manage more data.

Figure 2 illustrates the most common consolidation methods in use in comparison with the *progressive* (or *incremental forever*) methodology of TSM. Increasing points in time are displayed on the left as times T_0 through T_5. Each column in the figure represents a different backup technique, with use of tape for backup storage depicted as square tape cartridges. The dark areas on the cartridges represent used portions of tape, whereas lighter regions represent unused tape or regions of tape that are no longer valid because the file copies in these areas are no longer needed.

Incremental backup processing is shown in the first column. The technique involves periodic full backup operations that copy all data (at times T_0 and T_3), interspersed with "incremental" backup operations that only copy data that have changed since the last full or incremental backup operation (times T_1, T_2, T_4, and T_5). Although relatively efficient for backup processing, the technique can require the highest number of tape mount operations when restoring data. A full restore operation needed shortly after time T_2 but before time T_3, for example, would have to restore the data from tapes created at times T_0, T_1, and T_2.

Differential backup processing, column 2 in the figure, is similar to incremental backup processing. Periodic full backup operations (times T_0 and T_4) are interspersed by "differential" backups (times T_1, T_2, and T_3), which copy all data that have changed since the last *full* backup operation. Restore operations are more efficient than with incremental processing because fewer volumes need to be mounted, but differential backup still requires unchanged data to be repetitively copied (or backed up) from the client to the server. A full restore operation after time T_2 in the differential model would need data from tapes

created at times T_0 and T_2 (since the tape at time T_2 also contains the data copied to the tape created at time T_1).

Incremental and differential processing require that all client data be resent periodically so that new tapes can be created and those written on in earlier operations can be made available for reuse. TSM backup processing, illustrated in column 3 of Figure 2, is incremental in nature for every backup operation after the first full backup (T_0). As changed file copies are sent to the server, earlier versions are no longer needed. These older copies represent logical empty spaces in the server tapes and are represented by white squares in the figure. Instead of resending all client data to consolidate space on tape, the TSM server automatically reclaims the tape volumes written in earlier operations by copying valid (unexpired) data to new volumes (shown at time T_3). The emptied tapes are then made available for reuse. The reclamation and co-location process (described later) continually reorganizes data on tape volumes so that restore operations are optimized and a minimal amount of data needs to be sent from the client during backup operations.

Network-attached storage appliances are another example of hardware requiring new backup methodologies.

The adaptive subfile differencing technology of TSM extends the philosophy of sending only changed data for mobile user connections in extremely slow communications environments such as phone lines. The differencing technique is applied to certain files (as specified by the user or the administrator) so that only changed portions of updated files are sent. After an initial backup of the entire file (the *base*), changes are calculated by comparing the base file with the current version of the file.[1] If the changed portion is small enough (less than 40 percent of the total file size), only the changed data (a *delta*) are sent, along with information on the base file on which the changed portion is dependent.

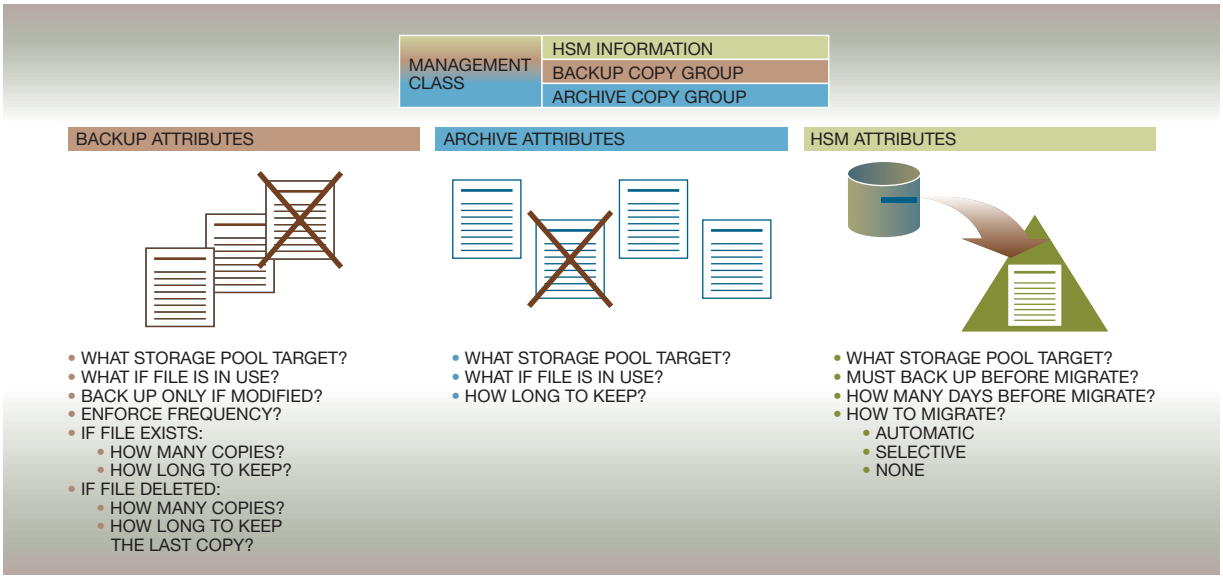
The server tracks the relationship between file deltas and their dependent base versions so that the proper file image can be reconstructed correctly during a restore operation. TSM also ensures that the base version of a file exists when a delta is backed up. If the base is not found (because of inadvertent deletions or damage caused by volume handling, for example), the server forces the client to resend the entire file.

Backup and restore traffic on the communications network (local-area network, or LAN) can be minimized through the use of a storage area network (SAN). When a SAN connects server storage devices and clients, TSM can be configured for direct client I/O operations through the SAN to server-owned devices for backup or restore operations. The communications network is only used for communication of meta-data about data placement or media mount or dismount operations. If SAN problems are experienced, backup or restore operations simply resort to using the communications LAN. This "LAN-free" technique not only frees the communication network, but also removes processing overhead from the TSM server.

The SCSI-3 (Small Computer System Interface 3) extended copy command provides another means to off-load data movement from both the TSM client and the server.[2] "Server-free" backup and restore operations rely on an intermediary in the SAN fabric to perform data movement outboard from the host machines involved in a backup or restore operation. The intermediary, typically a Fibre-Channel/SCSI bridge device, must be SAN-connected to both the client and server storage devices. An IBM SAN Data Gateway, for example, can be used by TSM to read the data from clients' disks and write the data to the tape drives utilized by the TSM server. Logical snapshot technology is utilized to obtain data copies while client I/O operations remain active. Client application or file system I/O is momentarily suspended in a consistent state to start the logical snapshot. After this point, copy-on-write techniques retain the original contents of updated file system or database blocks in another location. The original block images and their associated addresses are stored with the volume image so that they can be used at restore time to reconstruct the file system or database image as it existed when the logical snapshot was made.

Network-attached storage (NAS) appliances are another example of hardware requiring new backup methodologies. TSM addresses NAS appliance backup and restore operations through support for the Network Data Management Protocol (NDMP).[3] TSM uses NDMP to back up large NAS appliances to tape drives

Figure 3 Management class attributes



directly attached to the NAS appliance, or to tape drives shared with the TSM server. These methodologies also prevent the large amounts of data that are stored on a NAS appliance from having to be sent over the LAN.

Policy management: All data are not created equal

Mainframe storage management demonstrated that an automated means could be utilized to identify and manage data according to different business needs.[4] Critical application data have backup requirements different from those of work files or utilities. Manual management of the differing needs for data retention and versioning is expensive and ineffective. Likewise, maintaining copies of all of the data according to the most restrictive requirements (of some of the data) is also prohibitive because of the need to store all the unneeded copies. If the most important files in the file system need to be maintained for nine months, in this model, then the tapes on which the entire file system backup is written must be maintained for nine months.

The need to manage individual files[5] at varying levels of importance has not been met by simple backup mechanisms. TSM is designed to overcome the expenses in manual management and data retention through the application of customer-defined policy. The policies of TSM specify the number of versions of a backup copy to keep and the amount of time to keep them. The user or administrator binds (or associates) individual files (based on path name or file name wild-card specifications) to a specific versioning and retention policy. The container for these criteria is the management class, illustrated in Figure 3. It contains a number of attributes that govern the frequency at which the associated file should be copied, how many versions are to be maintained, and how long the versions should be maintained. If the business needs for the data change, the management class can easily be updated or new management classes created. This method represents a major change from the full plus incremental/differential backup paradigm that retains sets of tapes for a certain period of time.

Declarative attributes that govern the management of backup copies are grouped together in a *backup copy group*. Conceptually, multiple backup copy groups can be supported in the same management class, each outlining different backup criteria (e.g., one backup copy group might be used for daily backup operations, whereas another specifies the attributes for backups that are to be taken on a quarterly basis). TSM currently supports only a single

backup copy group in a management class. The following is a list of these attributes.

- Retention attributes specify how long, in days, that associated files are to be kept.
- Version attributes specify how many backup copies of a file are to be kept.
- Backup copy retention and version attribute values can be specified differently for backup copies of files that still exist on the client and those that have been deleted from the client.
- A special retention attribute can be specified to determine how long the last backup copy is to be maintained for a file that has been deleted from the client.
- Serialization determines how backup processing should handle files that are in use during the backup operation. If a file is in use and modified during backup, TSM can optionally retry the operation to attempt to obtain a copy that has not been modified while being backed up.
- Copy frequency and mode attributes are used to determine how often a file should be backed up and whether the backup should occur only if the object has been changed since the last backup operation.
- The copy destination is a symbolic name that identifies the storage pool in which TSM should first attempt to store the file copy. Storage pools are described in detail later.

Policy conflicts can be experienced when versioning and retention attributes are interpreted by the system. The versioning attribute might specify that a certain number of copies have to be maintained, but some of those copies may be older than the value specified for retention. To resolve this conflict, retention and version values work together to determine when backup copies should be marked for deletion from the storage repository of a server. Versions are deleted when they have been retained for a period longer than the retention value, or when there are more file copies than specified in the versions attribute. When performing versioning, the oldest versions are removed first. An inactive[6] file copy is deleted, then, when eligible by either the retention value or the versioning values specified in its bound management class.

Archive operations are used to keep point-in-time copies of important data for regulatory or bookkeeping requirements. The files are retained for the period specified in the management class to which they are bound and carry no notion of versioning. A common description, or package name, can be used to group sets of related archives together for later lookup and retrieval. Archive attributes controlling retention, copy serialization, and the initial storage pool for data placement on the server are contained in the management class called *archive copy group*.

Tivoli Storage Manager support for client space management is also controlled by management class policy. Space management, or hierarchical storage management (HSM),[7] is a technique used by specialized TSM clients to allow over-commitment of the space in a client file system. Over-commitment is achieved through transparent movement of the data associated with some of the files to the TSM server. To the user of the file system, it appears as though all files still reside locally. File size and access dates are used to determine which files are migrated from the client file system to the server. When a file is migrated, the file name is maintained via a "stub file" in the file system directory. Migrated files are automatically recalled from the TSM server when an application accesses their stub files.

Management class attributes govern the handling of migrated files and determine the kinds of space management activities supported (e.g., automatic migration by a client daemon, explicit migration by a client administrator, or no migration), whether or not migrated files need to have backup copies in the server before they are eligible for migration, and the initial storage location into which migrated files should be placed on the server.

A policy domain groups management classes and associated client nodes, acting as a scope of administrative control (an administrator may have control over one or more policy domains). Every TSM client node and management class is associated with one and only one policy domain. Management classes are bound to, or associated with, client files by using path and file name wild cards during backup, archive, or migration operations. End users or administrators specify include and exclude rules that control which files are eligible for certain operations and the management class to which they are to be bound when the operation is performed. Explicit ordering protocol in the application of these rules results in each file being associated with one and only one management class. Possible conflicts in applying wild-card operations are resolved through the ordering protocol in the application of include and exclude rules.

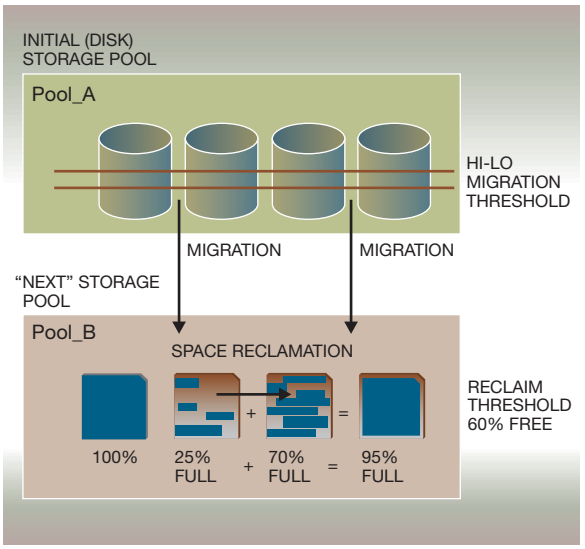
Providing the means for storage management: Storage pools

The server must record where individual files are placed in order to be able to relocate them, consolidate tapes, and maintain different file copies based on management class retention and versioning attributes. The server must scale, potentially managing thousands of clients with hundreds of simultaneous backups, without requiring hundreds of tape drives to be simultaneously available at the back end of the server. Requirements such as these motivated the implementation of a storage hierarchy within the TSM server with removable or fixed media drives organized into storage pools. Pools of storage provide a level of abstraction to the large number of storage volumes and devices required to store terabytes of data. The abstraction provides simplification in organizing the hierarchy and in specifying how certain storage is to be used. These systems-managed storage pools are at the heart of TSM automation.

A TSM server storage pool consists of one or more volumes of the same device type (e.g., disk, IBM 3995 optical drive, DLT (digital linear tape), and IBM 3590 tape drive). Nondisk (or nonrandom-access) device types are called *sequential storage devices* and are typically associated with automated libraries. The number of volumes in a storage pool is known so that total capacity can be calculated. Attributes associated with each storage pool govern automated actions for managing the files that are stored in the pool. Following are example attributes.

- A *maximum size* limitation for objects to be stored in the storage pool
- Storage pool *migration criteria*, consisting of thresholds that determine when migration should be performed and the name of the target storage pool into which files should be migrated (the "next" storage pool)
- Whether or not migrated files should be *cached* in the storage pool after being moved (disk storage pools only, as explained later)
- The number of *migration tasks* that should be used to copy data from the storage pool to the next pool for the data
- Storage pool *reclamation criteria*, consisting of a threshold at which reclamation should occur and an optional target storage pool for the reclamation
- The maximum number of *scratch* volumes that can be used in the storage pool (sequential storage devices only, typically tape)

Figure 4 Storage pool migration and reclamation



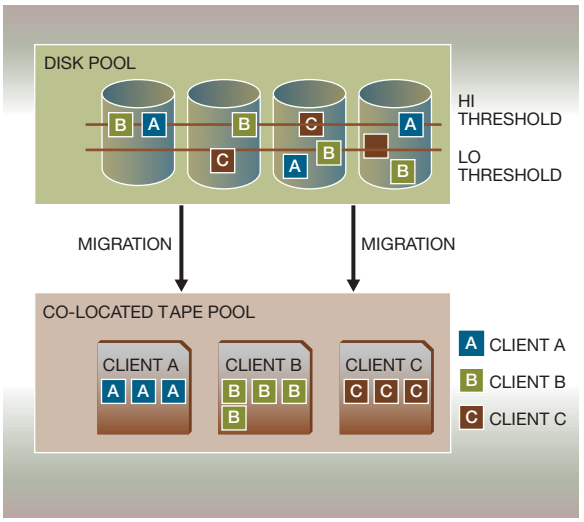
Source: F. Saldana, "Tivoli Storage Manager Overview," *SHARE Proceedings*, Nashville, TN (March 3–8, 2002).

- Storage pool *co-location criteria*, used to determine how client data are consolidated or spread across volumes

Storage pools are linked together in a hierarchy, typically structured with the highest-performing, most expensive devices, such as disk, at the top, and lower-cost, slower (sequential access) devices below. Extensive data copy and automated movement operations between storage pools are provided. The use of disk at the highest level supports a large number of simultaneous backup operations without requiring a large number of tape drives to store the data as the data enter the TSM server.

Storage pool migration is an automated process that moves files from one storage pool to another when the utilization of the first storage pool reaches a threshold, illustrated in Figure 4 as the "hi" migration threshold. The "next" storage pool attribute determines where migrated data are to be placed. Data are migrated to the target storage pool until the utilization of the first pool falls below a minimum threshold (also shown). The next storage pool is also used when files are larger than the maximum allowed for the pool or the estimated size for a file is larger than the remaining free space in the pool.

Figure 5 Data co-location



Source: F. Saldana, "Tivoli Storage Manager Overview," *SHARE Proceedings*, Nashville, TN (March 3–8, 2002).

Storage pool reclamation automatically consolidates sequential volumes, also illustrated in Figure 4. The amount of usable data on sequential storage can decrease over time as individual files expire through versioning or as a result of retention policy. When these files reach their expiration date, "virtual" empty spaces begin to appear on the tape volume. This fragmentation wastes space on the tapes and slows the restore process because of the time required to skip over empty spaces. When the amount of reclaimable or unused space on a volume reaches the reclamation threshold, the remaining usable (or valid) data on the volume are copied to a new volume, and the original volume becomes empty and available for reuse. The reclamation process consolidates tape data and makes traditional full plus incremental/differential operations obsolete as discussed earlier. In Figure 4 the contents of two tape volumes with 25 percent used space and 70 percent used space are copied to a third volume, which will be 95 percent used after the copy operations. The two partially empty tapes are then reclaimed as empty volumes.

Although migration and reclamation operations are shown together in the same figure, they are actually not linked together and may execute at different times.

The storage pool co-location attribute determines how sequential volumes are filled. By default, client data are placed on any available volume in a sequential storage pool. This operation causes client data to be spread across a large number of volumes over time, requiring many volume mount operations during restore processing, significantly degrading performance. Co-location corrects this behavior by grouping files from a client, or a single client file space, on the fewest number of sequential volumes possible. In Figure 5 we see that although data from clients A, B, and C are comingled in the disk storage pool, they are copied to distinct storage volumes when migrated to a co-located tape storage pool.

The multithreaded implementation of the TSM server supports multiple migration, reclamation, and backup or restore tasks executing concurrently. Manual data movement is provided through the use of administrative commands. The storage pool, together with individual file information tracked in the server database, supports minimal data transfer and granular file management.

Storage pool data placement

The TSM server database is a central catalog that maps client file names to location information on storage media in the storage hierarchy.[8] Client file names are mapped to surrogate keys called object IDs (identifiers). The 64-bit object IDs are mapped to tables in the storage subsystem that record the storage volumes and position information on the volumes containing the file data. Figure 6 illustrates the relationship between file names, object IDs, and the placement of the files in the storage hierarchy.

The relationship between object IDs and files is used to relocate or replicate files without renaming them. The technique extends the notion of location independence in a manner currently not supported by most file systems, where position in the namespace implies a dependent location of the data (e.g., all files in a file system are located on the same set of volumes in a volume group).

Location independence allows management of the data of a file to occur transparently to the application or user. This transparency is important in that files stored on the server can remain accessible to the clients, yet fulfill the intent of management policy for replication, movement, and intelligent data placement. Physical storage location is hidden from upper levels of processing and client functions that continue to access objects by their unchanged names.

When the TSM server migrates a file from a disk storage pool to tape, it can optionally continue to cache the copy on disk by not deleting its object ID and allocation information from the disk storage pool in the server database. This file, illustrated in Figure 7, resides in two places simultaneously: the disk storage pool and tape. The client is not aware of this fact and simply requests the file for a restore or retrieve operation by name. The server automatically returns the file stored on disk when receiving such a request because a tape mount is not required; the operation is automatically optimized.

The server database and recovery log work together with a transaction manager to implement two-phase commit processing.[9] Database and storage changes from aborted transactions are automatically rolled back so that partial file references are never created. The database and log reside on top of a logical volume manager that virtualizes access to disk volumes or file system files beneath it. This virtualization supports simultaneous three-way mirroring of the underlying storage. All database and recovery log updates are written up to three times on different volumes so that no data are lost should one of the volumes fail. Read operations can access any of the mirrored copies. A mirror can be dynamically taken off line for replacement or correction and later placed on line and resynchronized without affecting the availability of the TSM server.[10] The transaction-consistent operation of the server database and recovery log provides the integrity required for reliable, fault-tolerant operation.

Exploiting the architecture

The server storage pool hierarchy and data location abstraction enable a number of features that automate data placement, movement, and replication to comply with a different management policy conceived by the administrator.

Copy storage pools, as their name implies, contain copies of files that are stored in primary storage pools (the original storage pools described earlier). The copy storage pool provides a customer with the means to create duplicate copies in case of media failure or for off-site disaster recovery purposes. Copies are created in either of two ways: asynchronously through the scheduling or execution of the backup storage pool command, or synchronously during backup or archive operations that store new data on the server.

Figure 6 Decoupling file names and storage locations in the server database

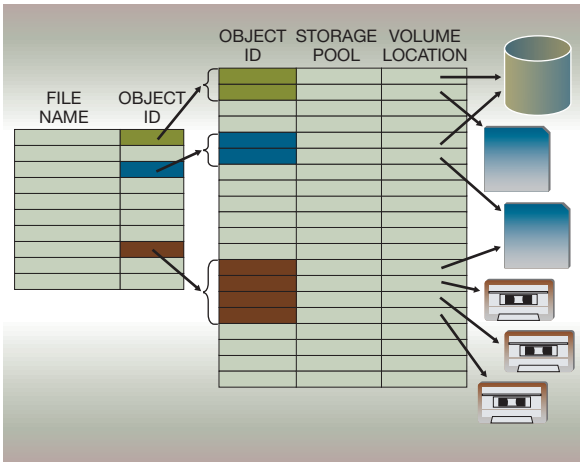
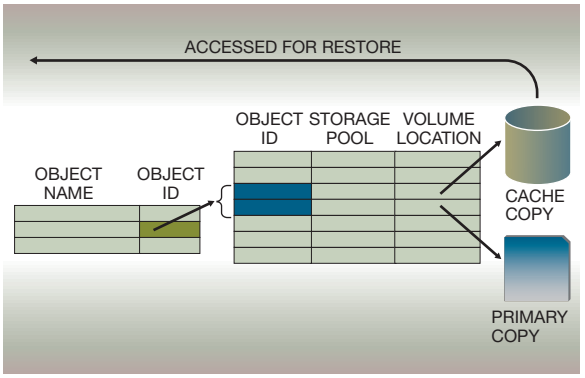


Figure 7 Transparent access to cached object copies for restore and retrieve



Copy storage pools are not structured in hierarchy-like primary pools. Files can be backed up automatically from any primary storage pool to any copy storage pool. The backup is incremental in nature: Files with IDs that are recorded in the server database as already residing in the copy storage pools are not copied to the pool again during a backup storage pool operation. Because the operation is applied to the file level and not to the storage volume, the primary storage pool and copy storage pools can use completely different devices. An entire storage hierarchy can be backed up to a single copy storage pool. A backup storage pool operation can be executed against the disk storage pool and the tape storage

pool to which the disk storage pool migrates. Files that are backed up while residing on disk are not backed up again after migration to tape.

If a file that is to be retrieved is found to have a primary copy that is inaccessible (e.g., the volume is marked "destroyed" or the copy cannot be read), the next-best candidate is automatically sought from a copy storage pool. The algorithm determines the best candidate by considering the following:

- Whether the volume on which the file resides is already mounted
- Whether the volume resides in an automated or manual library
- Whether the volume is on site or off site (e.g., resident in a local library)
- Whether the volume has been marked as damaged

The volume for the best candidate is automatically accessed or mounted (preferably from an automated library). If the volume is not available (e.g., is stored in an off-site location), a message is issued indicating that the volume has to be retrieved for the restore process.

As tape volumes in copy storage pools fill, the TSM Disaster Recovery Manager (DRM) function can be configured to eject those tape volumes from their automated libraries so that they can be taken off site for safekeeping. TSM continues to track the location and contents of volumes that are off site. As files on off-site volumes expire, virtual empty space begins to appear on the volumes. Normal tape reclamation would require the copy storage pool volumes to be brought back on site so that the remaining valid data could be consolidated on new tapes. Bringing volumes on site, however, would expose them to the very disaster situation that they are being taken off site to guard against.

Reclamation processing for off-site volumes takes advantage of the fact that the TSM database maintains explicit information about volume contents. Because these off-site volumes are copy storage pool volumes, primary copies of all files on the volumes still reside in primary storage pools on site. To reclaim an off-site copy storage pool volume, TSM automatically creates a new tape volume in the copy storage pool by recopying the files that reside on the off-site storage pool volume from their primary copies in their primary storage pool on site. Once these files have been copied to a new copy storage pool volume, the off-site storage pool volume is effectively emptied and may be brought back on site on the return trip that takes the new copy storage pool volume off site. This procedure ensures that there is always a valid off-site copy of a file. TSM tracks which volumes are on site and off site and generates information that tells administrators which off-site volumes can be brought back on site.

In addition to managing volume movement for copy storage pools and allowing for the entry of the off-site vault, row, and column location of the tape, the DRM feature automates the development of a recovery plan. The plan includes scripts to allocate and restore the TSM server from database backup tapes and configuration information that automates recreation of the appropriately sized database and recovery log volumes. Together with copy storage pools, the DRM automates much of the arduous task of planning for recovery from a significant disaster. The method can also be presented to auditory and certification committees as proof of a disaster recovery plan.

The embedded database backup function of TSM protects the server database by providing recovery should a disaster occur. The database backup can be scheduled to take full or incremental copies of the database while the server remains operational. Periodic image copies (called "snapshot" backups) can also be taken, consisting of logical full backups intended for use off site. The DRM tracks database backup volume location and manages the off-site and on-site transition of the volumes.

Whenever the TSM server database has to be restored to a prior point in time, a referential integrity problem can be introduced between the database and storage pool volumes. Database entries from earlier points in time can indicate that files reside on volumes that have since been overwritten by migration, reclamation, or move data operations. The server provides a means for reconciling this situation by tracking sequential volume usage in a volume history file. When restoring the database to an earlier point in time, the administrator can refer to the volume history file to see which volumes have been overwritten or reused since the time of the database backup to which the server was restored. Using this information, the administrator can selectively audit storage volumes to correct or remove database entries that refer to files no longer residing on those volumes.

A policy setting, the *reuse delay*, can be specified to retain data on sequential storage pool volumes when the server database is restored to an earlier point in time. The setting determines how long volumes remain empty prior to being overwritten, returned to scratch status, or reused for additional server operations. The use of this setting in concert with retention policy for database backups ensures that most data can still be accessed on sequential storage pool volumes even when the database has to be restored to an earlier point in time.

During a large-scale restore operation, the TSM server notifies the client whether additional sessions may be started to restore data through parallel transfer.

Device migration

Device migration is a task that storage administrators perform to take advantage of advancements in storage hardware technology. This task involves moving all data stored on one particular technology (e.g., 8-mm tape) to a new technology (e.g., DLT). Without a method to migrate the data, a customer will be subject to loss of data stored on a particular hardware technology when that technology is no longer supported.

Storage pools utilizing new storage technology can be defined in the storage hierarchy. Manual MOVE DATA commands can be executed or scheduled to move the data from existing storage pools to the new technology, or migration and reclamation can automatically move the data over time.

The export/import function of the TSM server provides for movement of data in a platform-independent manner between TSM servers on different operating system and hardware platforms, so that a TSM server can be moved from one platform to another, or be split into multiple instances should scale require.

Recent enhancements have eliminated common media as the transport between the two TSM servers. Instead, the source server can open a communication session directly to the target server, and client data and policy information are exchanged directly. This server-to-server export/import provides direct movement of client data between servers and replication of data so that clients can have data restored from more than one server. A server can incrementally export its data to another server, keeping both synchronized with the same client data. For those customers who require additional protection, this method eliminates any real-time single point of failure for accessing or restoring data.

Optimizing the restore operation

An administrator can stage data to a new location in anticipation of client access. The *move data by node* command is used to move data belonging to a specific client to disk in anticipation of a large-scale restore operation, to selectively migrate data for a particular client from one storage pool to another, or to selectively co-locate client data on as few media volumes as possible.

Co-location processing is intended to optimize performance by minimizing tape mount activity, but advances in multisession client support have highlighted some limitations. If a client's data are stored on a single tape volume, only one session can be used to restore the client, because tape cannot be simultaneously read from multiple locations. By spreading client data across multiple tapes, restore processing can utilize multiple parallel sessions to restore the client, with each session restoring files from a different tape. For large-scale restore operations, a less restrictive form of co-location will optimize throughput because it supports parallel sessions and is under consideration for development.

During a large-scale restore operation (e.g., entire file space or host), the TSM server notifies the client whether additional sessions may be started to restore data through parallel transfer. The notification is subject to configuration settings that can limit the number of mount points (e.g., tape drives) that are consumed by a client node, the number of mount points available in a particular storage pool, the number of volumes on which the client data are stored, and a parameter on the client that can be used to control the resource utilization for TSM operations. The server prepares for a large-scale restore operation by scanning database tables to retrieve information on the volumes that contain the client's data. Every distinct volume found represents an opportunity for a separate session to restore the data. The client automatically starts new sessions, subject to the afore-

mentioned constraints, in an attempt to maximize throughput.

Individual object tracking in the server database provides an opportunity for multiple choices in restoring data. Users can restore individual files, directories, or entire file systems. The restore operation can bring back the latest version backed up, a previous version of the files, or the data in the form in which the data appeared at a specific point in time. A point-in-time restore operation will utilize the server database to determine which files existed and which had been deleted or did not exist at the time specified.

Full volume (or full file system image) backup and restore operations are also supported. A volume-level backup or restore operation is faster than individual file-level backups and restores for all objects that reside on the volume because it does not have to process and track the individual files. Occasional volume-level backups are ideal for hardware or disaster recovery failures. File system image backups can be used in conjunction with progressive incremental file-level backups to provide the speed of a volume-level restore operation in certain situations and the granularity of a file-level restore operation in others.

The backup set function allows the creation of a stand-alone, self-sufficient copy of the active file system data of a client node. The server can generate a set of media containing backup data from a specified client node. This media set is then transported to the client node. The node can restore data directly from the backup set media, without having to contact the TSM server, and without having to move any data over a network. This technique is used to restore remote mobile users or, in disaster recovery situations, to restore high-priority server machines.

Enterprise management

A TSM server can support the needs of thousands of client machines. As the amount of stored data or number of client nodes continues to grow, however, administrators may find that backup operations for all nodes cannot be completed every night or that the expiration, reclamation, migration, and storage pool backup operations consume so much time and resources that they cannot extend backup windows to meet the expanding demand. The typical solution to this dilemma is to install additional TSM servers and storage hardware, but the challenge is to do so without greatly increasing the administrative cost for expansion.

The enterprise configuration functions of TSM address the total cost of ownership for users requiring multiple servers. Although additional processing and storage hardware is necessary to support multiple server instances, much more of the cost is associated with storage administrative staff required to maintain those servers.[11] Lowering the total cost of ownership for multiple server instances, then, is associated with ease of administration.

Enterprise configuration enables automated communication across multiple TSM servers to perform the following functions with the intent of appearing to the administrator as a single manageable, but distributed, entity:[12]

- Sharing, through subscription, consistent administrative definitions across server instances so that policy only needs to be defined in a single place, yet still be applicable across the enterprise
- Performing tasks on multiple servers concurrently through parallel command routing and enterprise log-on to all servers
- Centralized event (message) reporting for all server and client operations
- Storage of data managed by one server in the storage hierarchy of another server

The TSM server configuration manager provides for the definition and distribution of configuration information to managed TSM servers. Managed servers subscribe to profiles defined on the configuration manager. These profiles include administrative definitions, management class policy and bindings, and administrative and client schedules.

The configuration manager can be a server that is dedicated to this task or a fully operational TSM server that is also providing data storage for other clients. One or more definitions are grouped into one or more named profile containers. Managed TSM servers subscribe to named profiles to automatically receive the definitions that they contain whenever they are changed.

Subscribed profiles are distributed both in a push model and through passive polling by the managed servers. Administrators can push profile contents to subscribing servers after making a change with the *notify subscribers* command. The subscribing servers poll for updated definitions on a periodic basis, or

every time that the server is restarted. Administrator definitions, passwords, and authorizations are distributed through encrypted sessions so that commands can be easily routed across servers and the administrators can log on to any server using the same credentials. Regionalized control can also be implemented by selective subscription to very few administrators.

Command routing provides a mechanism for issuing commands simultaneously on all servers in a specified list or group. Server and server group definitions are maintained in the TSM server database (and propagated through the technique described above). Any administrative command can be directed to one or more servers from the administrative command line session of a single server. Returned results are organized so that the administrator can match them with their source.

The server maintains a log of all messages (events) that are issued in a database table that can be queried through defined query commands or SQL (Structured Query Language) select statements. Enterprise configuration provides the administrator with the ability to route server events (which include TSM client messages) to other TSM servers for centralized storage or to other event receivers such as the Tivoli Event Console, or to drive pagers or e-mail notices through "user exit" programs. The function can provide a centralized location from which to view all TSM messages regardless of the location from which they originated.

Servers can use each other's storage hierarchies in a manner similar to the way in which application clients use TSM. The technique utilizes one or more stored files on a target server to emulate a storage pool or other sequential storage volume on a source server. A source server, for example, can back up its server database information directly to another TSM server. Copy storage pools can be defined using these "remote volume" device classes so that a source server can back up its primary storage pools to the storage hierarchy of another server. Although the storage pool volumes reside on the other server, their contents are indexed by the database on the source server.

The TSM library-sharing feature enables sharing of storage hardware such as tape libraries in SAN environments. Library sharing allows two or more TSM servers to share the same automated library so that new TSM servers do not always require additional library hardware. One TSM server is configured as the library manager, performing all mount and dismount activity for the library devices for its own needs as well as those communicated by other TSM servers called library clients. In addition to performing this activity on behalf of other servers, the library manager tracks volume ownership in the library so that one server does not inadvertently overwrite volumes belonging to another server.

Active data use: Deep storage for content management

Placement and control of different types of data with transparent access to a hierarchy of storage devices distinguishes true storage management architecture from that of the backup and recovery application. These features, together with an open API, led to the use of TSM as an active data repository for content management applications.

Other "deep storage" management systems, such as UniTree**, originally developed at the Lawrence Livermore National Laboratory[13] in conjunction with the High Performance Storage System (HPSS),[14] were developed to meet the needs of institutions experiencing rapid data growth, demanding storage beyond traditional disk-only file systems. Similar notions of location independence and hierarchical pools of differing storage types were implemented as well.

Where and when possible, TSM has been enhanced to better support these kinds of applications, further extending its use into the management of active data as well as backup copies.

Partial object retrieve allows external applications to access parts of a large file without requiring its complete retrieval. After storing a large file in the storage hierarchy of TSM, the application can access (read) portions of that object by specifying offsets and lengths of the sections to be returned. A check imaging application, for example, might store large files consisting of multiple check images, but can retrieve individual images from each large file without retrieving the entire file.

Enhanced migration criteria enhance control over which files are migrated from disk to tape. When migration is required, TSM normally determines which client node is utilizing the most space in the disk storage pool and migrates all of that client's data to tape. This migration continues in an iterative fashion until enough data have been migrated so that space uti-

lization in the disk storage pool falls below the low-migration threshold. For content management applications, however, service level agreements require data access performance rates that are tied to the age of the data. The content management application might have to guarantee, for example, that objects less than 30 days old remain stored on disk for high-speed access.

Support for these service level criteria is provided with the migration delay parameter on primary storage pools. This parameter is used to specify the number of days from the time that files were inserted on the TSM server, or (optionally) last retrieved by the client, before being eligible for migration to the next storage pool. The migration continue parameter determines the behavior of migration processing should the low-migration threshold not be realized in migrating all objects stored longer than specified with the migration delay parameter. The specified value determines whether migration continues with objects that do not qualify for migration through the delay parameter to reach the low-migration threshold, or whether migration should stop.

Conclusions and future work

Data protection is not the process it once was, when simple backup tools could be used to manage the data stored on stand-alone machines. Data growth has given rise to complex new environments such as storage networks and an ever-increasing range of storage devices. Content-rich applications require support for a deeper level of the storage hierarchy, with policy attributes that can be tuned to guarantee levels of service in individual object retention and access.

The notion of namespace location independence and storage pool hierarchy of Tivoli Storage Manager provides a management system that can be configured to meet storage needs. Declarative policy attributes are associated with individual files to control retention and versioning without imposing operational procedures on the administrator. After time has been invested in identifying and planning for business needs, the software provides policy-setting capabilities to let the administrator articulate requirements. The data are then managed automatically and protected on multiple levels for recovery from various types of disasters. If business needs change, a central update can be made to TSM policy, affecting change in how storage objects are managed. Automation in TSM employs storage management techniques that support requirements imposed by active data, and not just backup copies.

Future work will likely include further exploration of the use of TSM technology as deep storage for content-rich applications involving massive amounts of data. The IBM CommonStore family of applications, for example, utilizes TSM for archive and document storage with HSM support for e-mail attachments.[15] Policy constructs may include higher-level business goals such as line-of-business or application recovery point and recovery time objectives. These constructs can be used to determine whether the proper infrastructure and backup schedules are in place to support recovery needs. The automated configuration and monitoring of backup and recovery and replication operations and infrastructure with respect to service-level agreements is a desirable goal to further reduce cost of ownership.

Efforts in the IBM Storage Tank* file system project,[16] a heterogeneous, shared SAN-based file system, leverage experience and technology gained from TSM. TSM server database concepts have been restructured and clustered to better support dynamic load balancing, failover, and recovery for the metadata controller that maps the file system namespace to individual disk locations for files. The TSM client API has moved into a standard native file system interface with caching and locking to provide high-performing, ubiquitous, and direct access to storage through techniques similar to LAN-free data transfer. Pools of homogenous storage devices with differentiated levels of service are utilized through new policy constructs to govern data placement and movement in a location-independent namespace. Each of these areas extends design points originally established for TSM, beyond backup, toward storage management.

Cited references and notes

1. R. C. Burns and D. D. E. Long, "Efficient Distributed Backup with Delta Compression," *Proceedings of the Fifth Workshop on I/O in Parallel and Distributed Systems*, ACM, San Jose, CA (November 1997), pp. 26–36.
2. *Information Technology—SCSI Primary Commands—1 (SPC3)*, T10 Working Draft, R. O. Weber, Editor, T10 Technical Committee, International Committee for Information Technology (September 17, 2002); see ftp://ftp.t10.org/t10/drafts/spc3/spc3r09.pdf.

3. Network Data Management Protocol (NDMP), Network Data Management Protocol Working Group, at http://www.ndmp.org/.
4. J. P. Gelb, "System-Managed Storage," *IBM Systems Journal* **28**, No. 1, 77–103 (1989).
5. "File" is used as a generic term in this paper to represent a directory, file system image, database, mail attachment, document, audio or video image, file, or any other data object.
6. A file copy is called the "active" file copy when it is the most recent backup copy of a file that is known to still exist on the client. All other backup versions (older copies) of the file are known as "inactive" copies. When a file is deleted from the client file systems, all of its backup versions become inactive copies. Active file copies are never deleted from the server by policy unless or until they become inactive copies.
7. D. Simpson, "Does HSM Pay? Be Skeptical!" *Datamation* **41**, No. 14 (August 1995).
8. The database also stores policy (management class) attributes and associations between management classes and files.
9. L.-F. Cabrera, R. Rees, S. Steiner, W. Hineman, and M. Penner, "ADSM: A Multi-Platform, Scalable, Backup and Archive Mass Storage System," *IEEE COMPCON 1995*, San Francisco, CA (March 5–9, 1995), pp. 420–427.
10. This is true as long as a valid mirror remains on line and working correctly during the operation, of course.
11. The cost of managing storage has been estimated at three or more times the actual purchase price of the storage hardware itself.[7]
12. *Tivoli Storage Manager for Windows Administrator's Guide*, GC35-0410-00, IBM Corporation.
13. W. Schroeder, "SDSC Enhancements to NSL UniTree," *Cray User Group (CUG) 1995 Spring Proceedings* (March 13–17, 1995); paper at http://www.sdsc.edu/projects/Systems_soft/UniTree/enhancements.html.
14. *High Performance Storage System (HPSS)*, IBM Corporation, http://www4.clearlake.ibm.com/hpss/about/HPSS-Brochure.pdf.
15. Content Manager CommonStore, IBM Corporation, at http://www.software.ibm.com/data/commonstore.
16. *IBM Storage Tank—A Distributed Storage System*, IBM Corporation (January 24, 2002), at http://www.almaden.ibm.com/cs/storagesystems/stortank/ExtStorageTankPaper01_24_02.pdf. See also J. Menon, D. Pease, R. Rees, L. Duyanovich, and B. Hillsberg, "IBM Storage Tank—A Heterogeneous Scalable SAN File System," *IBM Systems Journal* **42**, No. 2, 250–267 (2003, this issue).

General references

Achieving Cost Savings Through a True Storage Management Architecture (facilitated by Andrea Strahan), Tivoli Software White Paper, IBM Corporation (January 2002); available at http://www.tivoli.com/products/documents/whitepapers/.

L.-F. Cabrera, R. Rees, and W. Hineman, "Applying Database Technology in the ADSM Mass Storage System," *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB)*, Zurich (September 11–15, 1995), pp. 597–605.

Accepted for publication December 24, 2002.

Michael Kaczmarski *IBM Tivoli Systems, 9000 S. Rita Road, Tucson, Arizona 85744 (kacz@us.ibm.com).* Mr. Kaczmarski is a Distinguished Engineer working as a software architect for Tivoli storage products. Deeply involved in bringing TSM technology to market from its birth in IBM Research, he led the development of product functions and features over the last 10 years. He has a master's degree in computer science from the National Technological University and a bachelor's degree in management information systems from the University of Arizona.

Tricia Jiang *IBM Tivoli Systems, 5600 Cottle Road, San Jose, California 95193 (tricia@us.ibm.com).* Ms. Jiang is an advisory software engineer. She is currently working as a technical attaché to marketing. She has a bachelor's degree in computer science engineering from Northern Arizona University and a master's degree in management information systems from the University of Arizona.

David A. Pease *IBM Research Division, Almaden Research Center, 650 Harry Road, San Jose, California 95120 (pease@almaden.ibm.com).* Mr. Pease is a Senior Technical Staff Member in the Almaden Research Center and manager of the Storage Software department. Since 1996, he has managed the Storage Tank research project. He began working on storage systems research projects at Almaden in 1990. In addition to Storage Tank, he has worked on projects relating to the Tivoli Storage Manager product formerly known as IBM ADSM and the Universal Disk Format (UDF) file system for DVDs. For 12 years prior to joining IBM, Mr. Pease ran his own business, consulting and teaching software development and operating systems. In 2000, he completed his work for a Master of Science degree in computer engineering at the University of California, Santa Cruz, where he is currently a Ph.D. degree candidate.