System-managed storage

by J. P. Gelb

In early 1988, IBM announced the Data Facility Storage Management Subsystem (DFSMS™), comprising functions in MVS/DFP™, other products in the Data Facility family, and RACF. This announcement constituted a major step in the realization of system-managed storage. The need for system-managed storage was established in the late 1970s and early 1980s, through growing customer requirements in the management of external storage space, performance, availability, and device installability within and across systems in these customers' installations. The concept of system-managed storage is an evolutionary one, culminating in a resource manager for external storage that separates the logical view of data from physical device characteristics, simplifies interfaces for the use and administration of storage, integrates the functions of storage management products, and provides a synergy of hardware and software functions to effect complex-wide management of external storage resources, as discussed in this paper.

On February 15, 1988, IBM announced the Data Facility Storage Management Subsystem (DFSMS™). This is a major step in the realization of a concept termed *system-managed storage.* This announcement focused primarily on the Multiple Virtual Storage/Data Facility Product (MVS/DFP™) and its component, the storage management subsystem. A subsequent announcement on April 19, 1988, described functions within the Data Facility Hierarchical Storage Manager (DFHSM), Data Facility Data Set Services (DFDSS), and Resource Access Control Facility (RACF) products that, together with MVS/DFP (and DFSORT), comprise DFSMS.

This paper explores the rationale for system-managed storage, the concepts and facilities that it comprises, and the significance of the solution in the management of data and external storage. We focus on the problems that exist in manual management of data and storage, the elements of the solution, and the staged introduction of that solution over time.

IBM began examining the problems associated with the future of storage management in the fall of 1979. Essentially, the questions that were asked were the following:

- What critical problems will exist in the areas of data and storage management over the next ten years?
- How do they affect IBM customers and products?
- When do these problems become critical?
- What factors contribute to the problems?
- What should IBM's goals be to solve these problems?
- What are the elements that any solution must include?

Working groups were established to address these questions and determine the direction of a solution. They identified the mid- to late-1980s as the critical period in the management of storage, and predicted the IBM environment would become one that would exhibit high I/O (input/output) bandwidth and MIPS (millions of instructions per second), with the potential for increased I/O bottlenecks. New direct-access storage device (DASD) technologies were predicted to be emerging, thereby producing the need for coexistence between new and old devices. Additionally, IBM was predicted to be supporting multiple operating systems, a large number of new (and old) program products, and a proliferation of data and storage management facilities that would contribute to high maintenance costs. These factors would make it increasingly difficult for IBM to support customer storage management requirements effectively.

The median customer in that time period could correspondingly be characterized as experiencing complex installation management in a loosely coupled (or distributed) multihost environment. There would be significant increases in I/O bandwidth and processing power (MIPS). Job loads would involve both interactive and batch processing, with an emphasis on shared local data for real-time use. End users would become increasingly involved in the management of data (and the storage on which the data would reside) incurring increased data-management costs (for control, programming, availability, and maintenance) and the need for centralized data control. A typical customer would exhibit more concern about data security and integrity and would have a large investment in storage products to support the installation.

Based on this analysis, the following problem areas were identified:

- Device support, migration, and exploitation—the need for timely support of new devices, tools to simplify the movement of data to those devices, and the ability to fully exploit the device capabilities
- I/O performance—the need for enhanced data delivery to and from applications
- Storage sharing—the need to optimize the use of storage resources
- Data storage technology flexibility—the ability to explore and exploit new storage technologies with minimal user impact
- Usability—the need to simplify the processes of managing data and storage for end users and system administrators

In addition, it was determined that any solution IBM put in place had to have several basic attributes with regard to storage, data, and use. For storage, the key attributes include hiding physical device characteristics from application (or user) awareness, thereby enabling device, controller, or storage subsystem modifications to be transparent to the host; providing increased DASD connectivity; and improving storage space utilization. For data, it is important to maintain support for existing data interfaces, increase data access performance, and provide a single focus for data sharing, integrity, security, authorization, and commit. Key to data and storage use are the reduction or elimination of user involvement in the specification and respecification of data attributes (with respect to storage types, device access patterns, performance requirements, etc.). This could be accomplished through a single, simple data-attribute-definition user interface and by maintaining data about data, and positive human factors (user-friendly diagnostics, error handling, and recovery).

The problem areas and solution characteristics were independently explored (and corroborated) by the GUIDE and SHARE user groups.

These analyses pointed to the need to establish a storage- and data-management strategy and direction in which the system—rather than the users—managed the storage. Some of the characteristics of this solution include the following:

- Elimination of device characteristics in applications and transparency for data and device migration/conversion
- Localization of device-support code
- Exploitation of storage-device capabilities
- The spreading of data across volumes
- Data management on a logical basis to provide user-selectable data performance and availability and policy- and command-driven data maintenance concurrent with use
- Optimization of data transfers
- Centralization and consolidation of control and information for data and storage, with dynamic modification of control structures
- Provision for establishment tracking and auditing

In particular, the strategy had to provide system-managed solutions to meet the requirements of *space management* so as to increase space utilization capacity and simplify the establishment's involvement in space allocation, reorganization and defragmentation of space, and data migration. The strategy also had to provide for *performance management* to enhance the storage system's data delivery capability, provide relative logical levels of I/O performance criteria to users rather than absolute physical levels, and improve I/O response consistency on an establishment basis. Provision was needed for *availability* *management* to support continuous operations, enhance multisystem data access, and simplify data maintenance processes (backup, recovery, archiving, and disaster backup/recovery). *Device install management* was needed to reduce the proliferation of device support, decrease the amount of work required by the establishment to install and effectively use new or additional devices, and eliminate application involvement in device-dependent optimizations. *System-wide management* was required to centralize the administrative control of external storage, provide a single focus for the management of storage throughout multisystem installations, and enhance access and resource control mechanisms.

In addition to the solution itself, three observations were made. First, the solution elements cross problem and requirement boundaries because the problems were not themselves independent. Thus, the solution should not result in creating individual managers for space, performance, availability, and device installation. Second, a total solution would take time to implement and would result in major, dramatic changes to the way establishments process and control data and storage. Finally, the complexity of the solution required more computing power than System/370 could provide.

These led to the cornerstones of the strategy for a system-managed solution: a single, integrated storage resource manager to optimize the aggregate storage characteristics of space, performance, availability, and installability in a consistent manner; the storage resource manager to be coordinated with other system resource managers; and evolved in stages over several releases, beginning in the mid-1980s. Additionally, the solution would move in a direction to integrate software storage and data management functions into a coordinated system. It would separate the logical and physical aspects of data and storage specification and control. The solution would simplify the interfaces for the administration and use of data and storage. It would also utilize hardware support within both the host and the storage subsystem to make data and storage management more effective.

Data and storage management problems

The problems of managing data and external storage within customer installations began reaching significant proportions in the late 1970s and early 1980s. The growth rate of external storage and processor capacity (MIPS) in support of business and application growth, together with hardware technology advances, provided increasing pressure to manage storage more effectively. Additionally, the complexity of performing storage management increased with application growth, operating system complexity, and the need for 24-hour-per-day operation. This section examines these problems from customer and environmental viewpoints.

The complexity of performing storage management increased with application growth, operating system complexity, and 24-hour-per-day operation.

Customer perceptions. Customer concerns in the area of data and storage management have been expressed through direct interviews, surveys, Authorized Program Analysis Reports (APARs), Product Application and Support Requirements (PASRs), and formal requirements and white papers submitted through user groups. Some of the key areas addressed include the following:

- Automatic data migration and recall, based on device characteristics and data requirements
- System-controlled backup, based on installation-supplied attributes and only when the data have changed
- Automated recovery with damage-assessment aids
- Data sharing, including concurrent read/write with multisystem support and integrity
- Migration and coexistence with user-transparent device conversion and new-device exploitation
- Dynamic tuning of I/O performance with monitoring facilities
- Reliability, availability, and serviceability via nondisruptive failure isolation with integrity
- Security and data protection through owner-directed authorization of all resources with auditing

- Storage resource utilization involving automatic, geometry-free use of space with administrative control of limits and threshold monitoring
- System management of storage resources through existing facilities with feedback to and control by administrators
- Usability and productivity via a user-friendly logical view of storage with controlled defaults and increased system involvement to reduce complexity

Both the SHARE and GUIDE user groups expended considerable effort analyzing the current and future storage management environment from a customer perspective. A SHARE white paper[1] discussed storage management needs for large systems. Specific elements of the paper addressed device-independent input/output, data access control, the use of data-oriented definitions rather than foreign hardware terms, a hierarchical storage manager, and other requirements. A key aspect to providing these elements was "... (an) evolutionary plan (that would) ... lead to a simpler, more efficient, and more usable operating system."

Both the SHARE and GUIDE user groups expended considerable effort analyzing the storage management environment from a customer perspective.

In 1981 SHARE also drafted a position paper that lists a set of requirements that focus more directly on storage management. These requirements include the following:

- Installations should be able to define defaults for common storage parameters, such as space, unit, and block-size attributes.
- At most, only one widespread change of job control language (JCL) or time sharing option control lists (TSO CLISTs) should be required to obtain these improvements rather than each time a new device or default is established.
- Allocation of space should be viewed logically by records or by byte count, instead of physically by tracks or cylinders.
- The user should be able to mix datasets in a concatenation with different block sizes and attributes, independent of storage device characteristics.
- IBM should provide reasonable defaults for the requested parameters, based on an "average" installation. These defaults should be easily modified by the installation, but should be suitable for most installations.
- IBM program products and distributed JCL should make use of these defaults wherever possible.

In March 1983, GUIDE concluded a multiyear project through the publication of a strategy paper titled *Requirements for Futures of Storage Management.*[2] The paper set forth the following requirements:

- A revolutionary solution is presented in an evolutionary manner.
- Productivity of support personnel must exceed storage growth rate.
- Users must be aware of only data attributes and not aware of physical attributes.
- The subsystem must be self-adjusting to a changing environment.
- Data must be accessible across multiple execution environments.
- The storage subsystem must be capable of recovery from failure.
- The security interface, a universal interface to a standard system security manager, must be available in the storage subsystem.
- The storage subsystem must be flexible to allow the addition and removal of components without service interruptions.

The paper concluded with the statement: "Implementation of solutions to the problems outlined meeting the requirements stated, will likely be a series of evolutionary steps or new features introduced in the one- to five-year timeframe with a complete solution in the five- to ten-year timeframe."

Since 1979, both user groups have submitted numerous requirements in the area of data and storage management. These requirements have focused on both specific near-term solutions for immediate "relief" and long-term directions for a more general solution.

The changing data processing environment. The results of surveys and customer visits over the last several years have made it apparent that the data processing environment is changing and growing dramatically. In 1966 the median account had the processing power equivalent of 0.2 MIPS using the primary control program (PCP), an early version of Operating System/360. By 1984 that median account had grown to 29 MIPS under the Multiple Virtual Storage/Extended Architecture (MVS/XA™) operating system, and projections indicate growth to over 300 MIPS by 1990. This growth in processor capability, coupled with the increasing trend toward operating-system resource managers and away from manual control, has placed increased demand on storage and data management.

As processing power and operating system sophistication have grown, so too have the demands for storage, data, and data access.

As processing power and operating system sophistication have grown, so too have the demands for storage, data, and data access. In 1966 the demand for data was primarily from the batch environment. An average of 100 datasets, representing 200 000 000 bytes (200M bytes) of storage, were involved, and the application programmer could move the data to balance out the system load fairly easily.

By 1984 a median account had 44 000 datasets on 102 billion bytes (102G bytes) of storage. The workload on the system was largely database and interactive-processing oriented, with a smaller number of jobs evenly divided between batch and system data. The application programmer could no longer tune the system. Rather, system administrators were responsible for balancing and controlling data placement and migration. A study of seven accounts showed that, on average, one full-time person was needed to manage every 10G bytes of storage. Thus the 1984 median account required about nine people to manage storage, either as part of a single storage management group or spread among a larger number of people on a part-time basis. The actual number could vary greatly, depending on the account and the tools available.

Projecting observed growth into 1990, a median account could reach 250 000 datasets and over a trillion bytes (1.2T bytes) of storage. Although tools and techniques for assisting in storage management do exist and will probably proliferate without additional system assistance, increasing numbers of administrators (or application programmers in the more common decentralized environment) may be required to manage the storage. This assumes, of course, that manual management is even possible on such a large scale.

The numbers just given are based on a 45–60 percent compound growth rate for storage. They are also figures for a median account. Thus there are expected to be customers whose usage is well beyond these numbers. The requirements for DASD are growing faster than the requirements for MIPS, due to such factors as the need for more data about data and the requirements to maintain historical information for legal or business reasons.

It is clear from this growth that system control for the more efficient use of storage and the automation of storage management is necessary for the 1990 environment to exist.

Cost of storage and storage management. Along with the growth requirements for data and storage, there is another trend in the data processing environment that concerns the cost of storage and its management. The raw cost of storage, in dollars per megabyte per month, is decreasing. In 1978 external storage cost $2.61 per month for a megabyte (on a lease basis). By 1984 that cost had been reduced to $0.94, with improved price/performance and capacity, and that lowering cost trend should continue into 1990. This figure, however, reflects only the absolute cost of hardware and not its effective use.

In a GUIDE survey comprising 90 MVS customers and 639 000 datasets, it was determined that 55 percent of the storage capacity was unused as follows: 6 percent for gaps, 24 percent allocated but unused (e.g., reserved to reduce out-of-space errors), and 25 percent unallocated to increase perceived device performance. Only 45 percent of the storage devices (on average) was actually being utilized for data. The cost of this unused capacity effectively doubles the cost of the raw storage.

On top of these direct costs for effectively using storage for data are the indirect costs of the people needed to manage the storage. These are the people involved in space and performance management, dataset backup, and capacity planning. Though the cost of storage is decreasing, people costs are increasing, and the net effect of adding these costs shows a leveling of aggregate cost for DASD on a dollars-per-megabyte-per-month basis.

There is still another indirect cost factor associated with storage, and that is the cost of data availability: the cost per usable megabyte. This cost includes the dumping, mounting, and storing of backup data to insure against critical outages, normalized to the same dollars per megabyte per month. The aggregation of all these costs indicates that in 1978, the cost per usable, managed megabyte of storage was three and a half times the cost of the raw storage itself. In 1984 that figure was approximately four and a half times the cost. In 1990 the projection shows that the cost (per month) of usable, managed storage will be ten times the cost of the storage itself. (See Figure 1.)

Consequences of user-managed storage. In a user-managed storage environment, users of the system manage their own application data (the logical domain) and the storage devices (the physical domain) on which that data reside. Existing externals (JCL, TSO, etc.) specified by these users interweave characteristics from both domains, as do device dependencies within applications. This hinders the separation of the domains for the users, and ensures maximum impact when the configuration changes or data movement must be done for performance tuning.

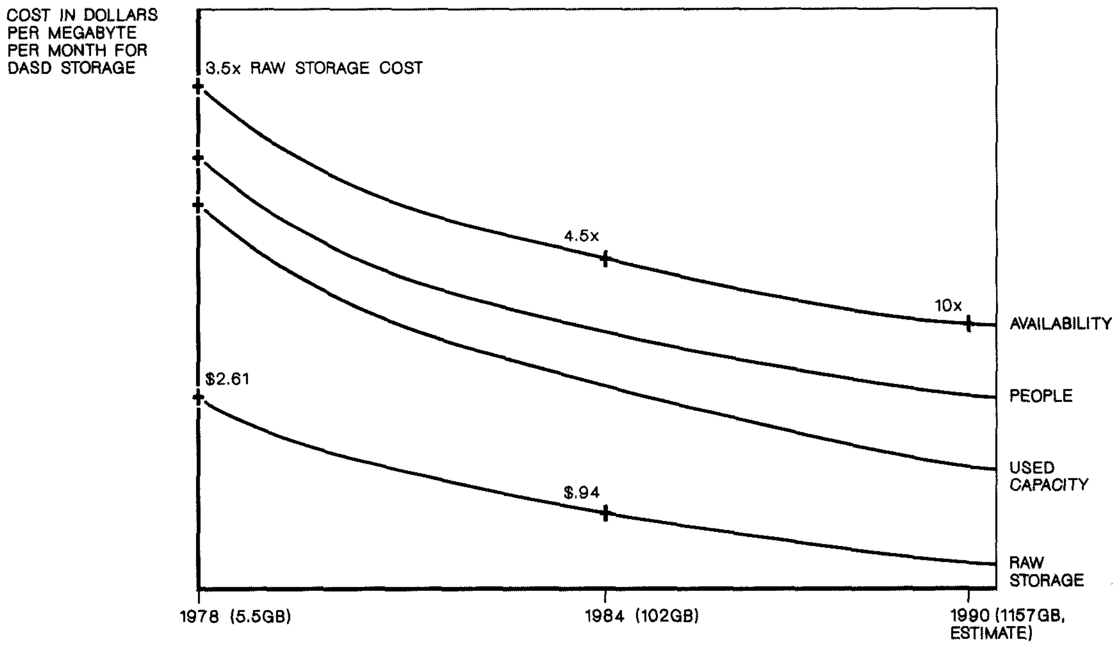
In this mode of managing data and storage, users may be characterized as having too little information and too much control. The user specifies data storage in explicit device terms and is consequently affected when devices are reconfigured or replaced by new device types. To preclude out-of-space abnormal job endings (ABENDS) due to lack of sufficient storage space, datasets are over-allocated, thus wasting space and increasing fragmentation problems when datasets are deleted. It is also the user's responsibility to match the data requirements to storage capabilities. Thus, users must be sensitive to the kinds of devices available (and to their characteristics) to meet the needs of their applications. To add to the problem, users perform these tasks with the single focus on their application. A user does not or cannot take into consideration all the other applications that may

Storage administrators have tremendous amounts of information as to the storage and data status, but little control.

be run concurrently, or whose data may reside coincidentally. The less sophisticated the user, the more likely the user will select unnecessary space parameters and device types for data, thus preventing more knowledgable users from getting what is really needed. Without the education and information of an installation storage administrator, the user can (and does) exercise excessive control over the precious storage resources, with potentially disastrous consequences on the user's application, modification of externals, and the installation as a whole.

An installation's storage administrators, on the other hand, are somewhat in the opposite situation. They have tremendous amounts of information available (volume table-of-contents listings, catalog listings, resource measurement facility data, system management facility data, etc.) as to the storage and data status, but little control over the environment on an on-going basis. At best, they must wade through copious amounts of data (largely a clerical task) and then publish guidelines and policies to govern future use of the storage. In many installations, the administrator can do little more than suggest procedures for users to follow in allocating to and using the physical devices. Although administrators may be responsible for tuning the system via data movement and data deletion, their hands may be tied by the users' power to override and lack of sophistication. Several installations have instituted controls via job scheduling systems, JCL builders, installation exits, etc. Although valuable as aids in storage management and control, these tools often suffer as a con-

Figure 1 Trend in cost of storage versus cost of usable, managed storage



sequence of major system modifications or new installation standards.

In addition to the user and administrator role, consider the role of the system. It uses very little of the information at its disposal, and exercises little control over data and storage. The system is constrained, on the one hand, by the user's explicit specification of storage, and on the other hand by physical device geometry and architecture (tracks, cylinders, extents, etc.). Reconfiguration of devices, or installation of new devices, often is disruptive because of device dependencies and the scattering of device and configuration information. Thus, the system is relegated to only following orders, with little freedom to decide for itself or override what it has been told.

The roles of users, administrators, and the system in a user-managed storage environment create complexity for the management of data and storage. There are other factors, however, that contribute to that complexity and apply pressure on data and storage management. The continuing growth in the area of data processing directly affects data and storage management, as more and more data are created for on-line use. The increased data are not only a consequence of business growth but are also the result of the compounding effect of requiring more data about data for reporting, database, and interactive use, and for the growing amounts of inactive data for archive, vital records retention, and backup.

In addition to the amount of data created, there is increasing need for better data access in terms of performance and multisystem availability. The concept of single-system image requires that any user running any application on any system must be able to access the necessary data at any time. Both the data growth and its accessibility place increased pressure on data and storage management.

Processing power and capability are growing to meet the data processing demand. Systems are becoming faster and larger as are their number, and installations are growing and becoming more complex as well. More jobs run concurrently, with increased need for data access, which places more pressure on I/O subsystems to deliver the data in a timely manner. Performance tuning becomes increasingly important and complex in such an environment, and the windows for doing data and storage maintenance become smaller as the need for 24-hour-7-day-per-week operation grows to support the larger data processing demands.

Another factor that increases the complexity of and pressure on data and storage management is the evolution of storage devices themselves.

Another factor that increases the complexity of and pressure on data and storage management is the evolution of storage devices themselves. New generations of DASD offer increased space and performance characteristics, in conjunction with smarter control units offering cache control and multiway reconnect. This growth in pathway, capacity, and performance, coupled with the increasing numbers of attached storage devices, requires greater sophistication in the way data and storage management accesses and utilizes the data and devices.

The limits of current DASD technology, however, are being pushed. New DASD technology, coupled with a spectrum of device capability, will be necessary to meet the demands of data processing. Yet, existing device geometry characteristics are often built into applications and procedures, which will require major conversion time and energy to modify, if a substantially new architecture is to be unveiled. This can limit the ability to explore and exploit new technologies by forcing the engineering community to pursue technically expensive solutions in order to maintain compatibility.

Solution requirements. The increasing pressures on storage management to respond to business data processing growth on one hand and DASD, CPU, and memory evolution on the other, can potentially lead to higher cost and complexity in the management of both data and storage. Specifically, the problems introduced by these pressures can be categorized into five areas of data and storage management: space management, performance management, availability management, device install management, and system-wide management.

We now discuss each of these management areas as to its associated problems, the requirements a solution must satisfy, the relationships among the areas, and the elements of a solution.

Space management. Space management refers to the allocation, manipulation, and control of physical external storage space. In the past, awareness of physical space—tracks, cylinders, extents, etc.—was propagated throughout externals, interfaces, access methods, program products, operating system components, and applications. This global infiltration often resulted in the logical manipulation of data to avoid physical space problems and placed the duty for space management on the application programmer.

Problems associated with physical space management include out-of-space failures, fragmentation, poor capacity utilization, single-volume constraints, and device-type constraints. To solve these problems, applications and software should be isolated from dependencies on physical device characteristics. Also, control of the storage resource should be centralized, and the system should be empowered to manage that storage resource. To that end, the space management requirements are for the system to do the following:

- Allow an installation to utilize the real capacity of storage devices more completely, without performance or operations penalty
- Eliminate constraints on the ability to use storage space (e.g., number of extents per volume and volume-related limits)
- Simplify the human involvement in space allocation, reorganization, conversion, and migration
- Remove device dependencies from software interfaces and applications
- Provide space tracking, reporting, and limiting facilities

Performance management. Performance management refers to the placement of data on physical storage such that the logical access requirements of the data are matched to the access capabilities of the storage devices. Datasets have varying requirements for access performance. Typically, there is a mix of requirements, such that there are many (small) datasets with high-access needs and several large datasets (e.g., archive data) with low-access needs. In addition, there are many datasets whose access patterns vary, depending on the application that is referencing them and the time of day or year they are accessed.

Complicating this environment is the basically uniform sustained access capability of individual storage device types. The determination of which datasets are to be placed on which devices is often a difficult and time-consuming manual process. In addition, the placement is frequently performed after-the-fact. That is, data are moved because a problem has already occurred, which often creates new performance and contention problems. It is virtually impossible for manually driven or application-driven procedures to optimally place data in a timely manner.

In order to reduce performance bottlenecks due to contention for external storage resources, the performance management requirements on the system are the following:

- Improve the ability of the system to deliver data; that is, minimize the effects of seek, rotational delay, transfer time, and device and path contention
- Provide specifiable service levels for data delivery
- For the system as a whole, reduce the number of physical I/O operations to do the equivalent amount of work of a predecessor system
- For the system as a whole, increase the concurrency of physical I/O operations taking place
- Allow storage specifications to imply data placement that will result in localization of reference
- In system-determined instances, delocalize sequential data in order to get higher gross byte throughput
- Improve the utilization of system resources
- Provide tracking and tuning facilities

Availability management. Availability management refers to the process of maintaining the timeliness of data and its disposition, and the accessibility to data in a multisystem environment. Managing data availability has been a manual process, for the most part. Otherwise, it has been the responsibility of individual applications or of the installation as a whole. Typically, the preparation for logical outages (invalid data) and physical outages (media/device failures) has been accomplished during a *backup window*, which is a point in time (usually of low data activity) when storage volumes and individual datasets are dumped. Dumps are most commonly made to shelf media (e.g., tape), and duplicate copies of the data are often made to other on-line storage.

Eliminating the backup window, minimizing physical outages, minimizing data outages, and automating recovery are key goals to making data available.

The process of validly recovering data after an outage can be complex and time consuming, and an outage can be very costly to an installation. In addition, less time is being allowed for the backup window because of the growing requirement for 24-hour-per-day operation. Eliminating the need for the backup window, minimizing the effects of physical outages, minimizing data outages, and automating the recovery process are key goals to making data available.

There are also growing requirements for maintaining inactive copies of data (i.e., data that are not expected to be needed again). This includes archival data, vital records, disaster recovery, retired data, and older versions of data. These types of data are addressed more completely later in this paper.

The system should do the following in order to meet these availability management requirements:

- Simplify the storage maintenance process by creating point-in-time copies for backup, recovery, disaster backup and recovery, and for the archive
- Automate the process of making point-in-time copies, according to policy and through explicit instruction

- Standardize the process of making point-in-time copies across all applications
- Ensure that data are available for both read and write access, while point-in-time copies are being made
- Recover damaged data within some maximum amount of elapsed time, based on selected availability options
- Provide automatic repair of some classes of data, according to policy
- Allow and enhance data sharing within the same system complex and between systems via shared DASD

Device install management. Device install management refers to controlling the impact of attaching new or additional storage devices and making them available for use. In the past, the addition of storage to a system has often been disruptive and required changes to applications with device dependencies and modifications to several parts of the system itself. Effective utilization of new devices has been limited because of hard-coded or historical references to old, often obsolete device characteristics.

The need to maintain geometric compatibility at an external level has affected the ability to exploit new technologies and to market new devices with totally different geometries or attributes. The cost to an installation of converting existing data and storage to new devices has been high, which had the effect of reducing the usefulness of the devices.

Dynamic modification of the physical storage configuration without affecting logical data requirements and the automatic usage of new storage are of paramount importance. Thus the device install management requirements on the system are to do the following:

- Reduce the human effort required to install and effectively use new storage devices and to remove old devices
- Provide data access independently of device type or device characteristics
- Allow the addition or removal of devices or device types without shutting down the MVS system (for example, neither system generation [SYSGEN] nor I/O generation [IOGEN] should be required in order to add or remove devices)

System-wide management. System-wide management refers to unified control of data and storage across all systems in an installation. In the past, storage was reasonably confined to a single system. Facilities to manage that storage could be unique to the individual system and tailored to the application requirements of that system. With the increased need for shared storage associated with multisystem applications, as well as availability and capacity constraints, the need for integrated, cross-system tools and a single focus for control becomes evident.

Attempts to meet the individual management requirements must take into consideration all the management requirements.

To manage storage effectively across all systems in an enterprise requires system-wide support to do the following:

- Automate the storage management process
- Centralize control over storage resources
- Provide functional integration of storage management facilities with simplified, interactive maintenance capabilities
- Allow larger, centrally managed collections of storage or storage pools
- Enable data with differing service requirements to reside in the same pool
- Provide monitoring, reporting, and accounting information for all storage resources

Integrated approach. Finally, we note that these individual management requirements are not independent. Managing storage space affects the deliverable performance of the devices containing the space. Managing availability (e.g., by multiple copies of data) affects the amount of useful space. Any solution that attempts to meet the individual management requirements must take into consideration all the management requirements. The result is an integration of the various solution elements into a single system manager. The elements just described are summarized in Figure 2, and a solution that

Figure 2 Requirements versus solution elements

REQUIREMENTS					SOLUTION ELEMENTS
S			I		ELIMINATE DEVICE CHARACTERISTICS IN APPLICATIONS
S		A	I		MANAGE DATA ON A LOGICAL BASIS
S	P			W	SPREAD DATA ACROSS VOLUMES
S	P		I	W	EXPLOIT STORAGE DEVICE CAPABILITIES
	P			W	OPTIMIZE DATA TRANSFERS
S	P	A	I	W	CONSOLIDATE CONTROL INFORMATION
		A			SUPPORT DATA MAINTENANCE CONCURRENT WITH USE
S		A		W	PROVIDE POLICY-DRIVEN/COMMAND-DRIVEN DATA MAINTENANCE
		A	I	W	CENTRALIZE THE CONTROL OF DATA AVAILABILITY
	P	A			PROVIDE USER-SELECTABLE DATA PERFORMANCE AND AVAILABILITY
			I	W	LOCALIZE DEVICE SUPPORT CODE
S	P	A	I	W	ALLOW THE DYNAMIC MODIFICATION OF CONTROL STRUCTURES
S	P		I	W	BE TRANSPARENT FOR DATA/DEVICE MIGRATION/CONVERSION
S	P	A	I	W	PROVIDE ESTABLISHMENT TRACKING AND AUDITING

S=SPACE
P=PERFORMANCE
A=AVAILABILITY
I=INSTALL
W=SYSTEM-WIDE

included those elements would be an integrated solution that met the requirement areas.

Revolution within evolution. To meet the requirements described above implies the need for a revolution in the way data are stored and managed on storage media. The necessary changes would fundamentally affect storage devices, I/O processing, data placement, data and storage administration, and the mechanisms for relating data requirements to storage capabilities. A single-step solution that would satisfy most of these concerns would entail a major redesign of the data to storage relationship over a long period of time, precluding the timely introduction of new function and interim solutions, and necessitating considerable re-education of customers and their users. However, growth in data and storage must be accommodated, while a long-term solution is being created. A more effective procedure is to create the long-term solution over time, by providing a staged introduction of function and facilities while moving toward and culminating in the desired future solution. This is the approach that IBM is taking to produce the system-managed storage solution.

The system-managed storage solution

System-managed storage is the synergistic relationship among software facilities, hardware facilities, administrators, and users that enables the operating system to direct the effective use of external storage. One of the basic principles of system-managed storage is the separation of logical and physical views of storage. The logical domain pertains to the management of data, and includes requirements such as data intrinsics (e.g., record format and record length), data performance (e.g., fast access), data backup (e.g., weekly and number of versions), data retention needs (e.g., one year up to forever), and data security needs (e.g., who has what type of access). The physical domain is concerned with the storage devices themselves and includes characteristics such as capacity, performance, pathing, hardware availability, system connectivity, tuning, physical location, allocation, and free-space maintenance.

Another of the basic principles of system-managed storage is the simplification of the user interfaces. Simplification is the process by which users (e.g., administrators and application programmers) control external storage and the data that reside there. This entails isolating less sophisticated users from system internals and esoteric externals, providing a coherent view of facilities, and making the process of specifying, controlling, and managing both the logical and physical domains easier.

Another of the basic principles of system-managed storage is the simplification of the user interfaces.

Integrating the functional relationships among data and storage management products (e.g., MVS/DFP, DFHSM, DFDSS, RACF) is a principle by which a coordinated approach to the management of external storage is provided. This includes establishing a clear, evolutionary product direction, providing a common data and storage management focus, consolidating control information, and building on existing facilities.

Utilizing hardware function support in the host and in the storage subsystem is a principle by which function is streamlined. This includes improving data availability, optimizing data transfers, and supporting the dynamic modification of control information.

These principles form the basis for system-managed storage.

Key concepts and facilities. In a user-managed storage environment, physical storage awareness pervades data management, as suggested by Figure 3. This creates a major impact on data management facilities when the physical storage environment is modified. It also encourages users and applications to become dependent on the physical environment to satisfy data requirements. The consequences lead to underutilization of storage devices and contribute to a widening gap between processor demand for access to data and the storage subsystem hardware's capability to deliver the data. The consequences also impinge on an installation's ability to correct the situation.

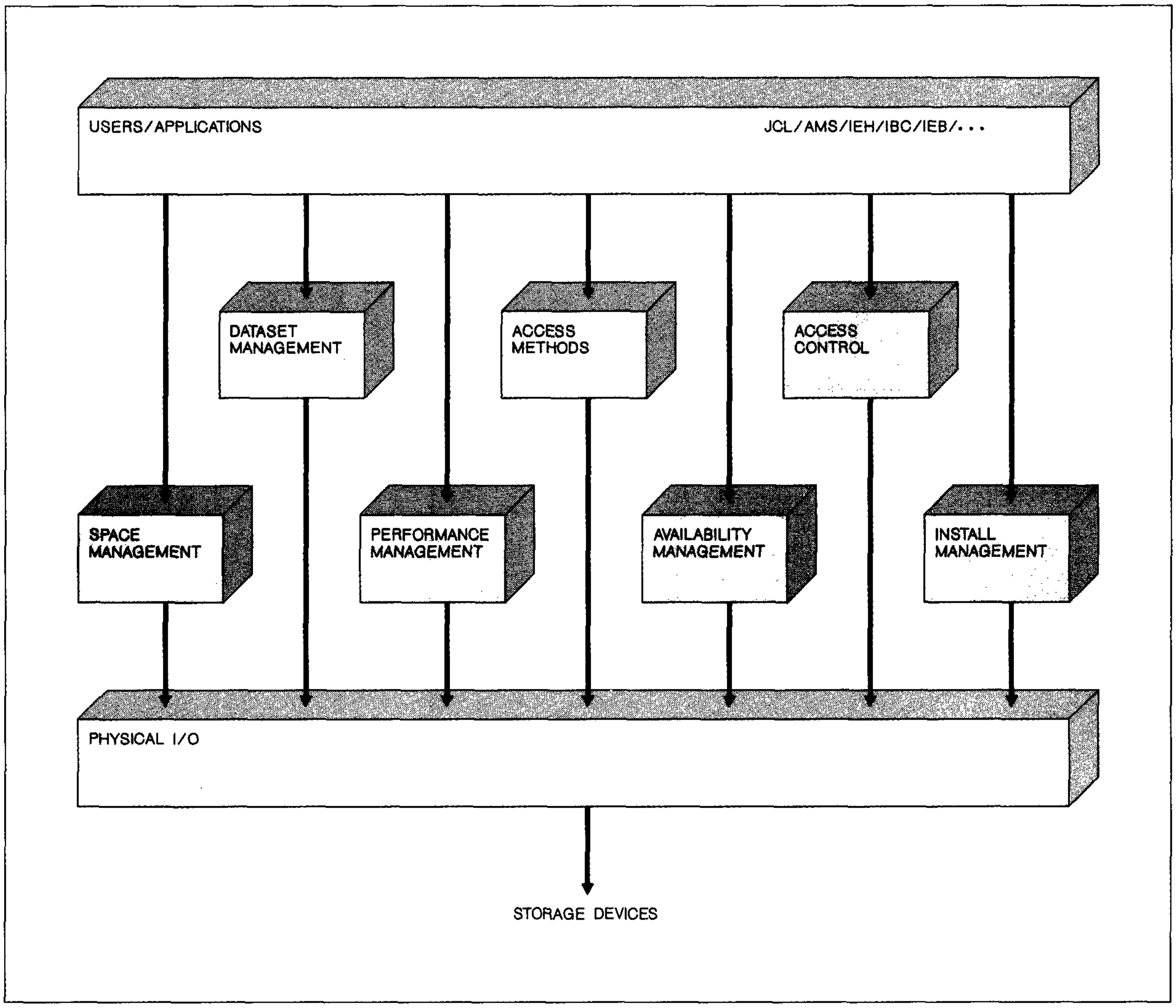
Figure 3 shows the relationships among data and storage management facilities and physical storage in a user-managed storage environment, and the arrows indicate storage awareness. The various facilities shown include the following functions:

- Dataset management—allocation, catalog, and associated services
- Access methods—record and buffer management
- Access control—identification and security
- Space management—physical allocation, utilities, aids, migration and recall
- Performance management—measurement, reporting, utilities, aids, and monitors
- Availability management—backup, recovery, error detection and repair, and reporting
- Device install management—storage conversion, data migration, utilities, and aids

Storage management subsystem. Two distinct facilities are required in the design of a system resource manager for external storage. One is an external storage manager to manage the physical storage, and the other is a logical-data manager to manage the data requirements. The *external storage manager* (referred to here as ESM) is the system resource manager for physical storage. It is the owner of the physical storage-oriented domain, and is the focus for system management of physical storage with respect to space, performance, availability, and device installation. It also provides support for local device attachment (and attachment exploitation), cache management, control over device configuration, media refresh (rewriting data as the media age), multisystem storage control, and a common storage format across devices.

Figure 4 shows that the ESM is the only facility with awareness of the physical storage. It subsumes the externalized functions associated with space management, performance management, availability

Figure 3 User-managed storage environment



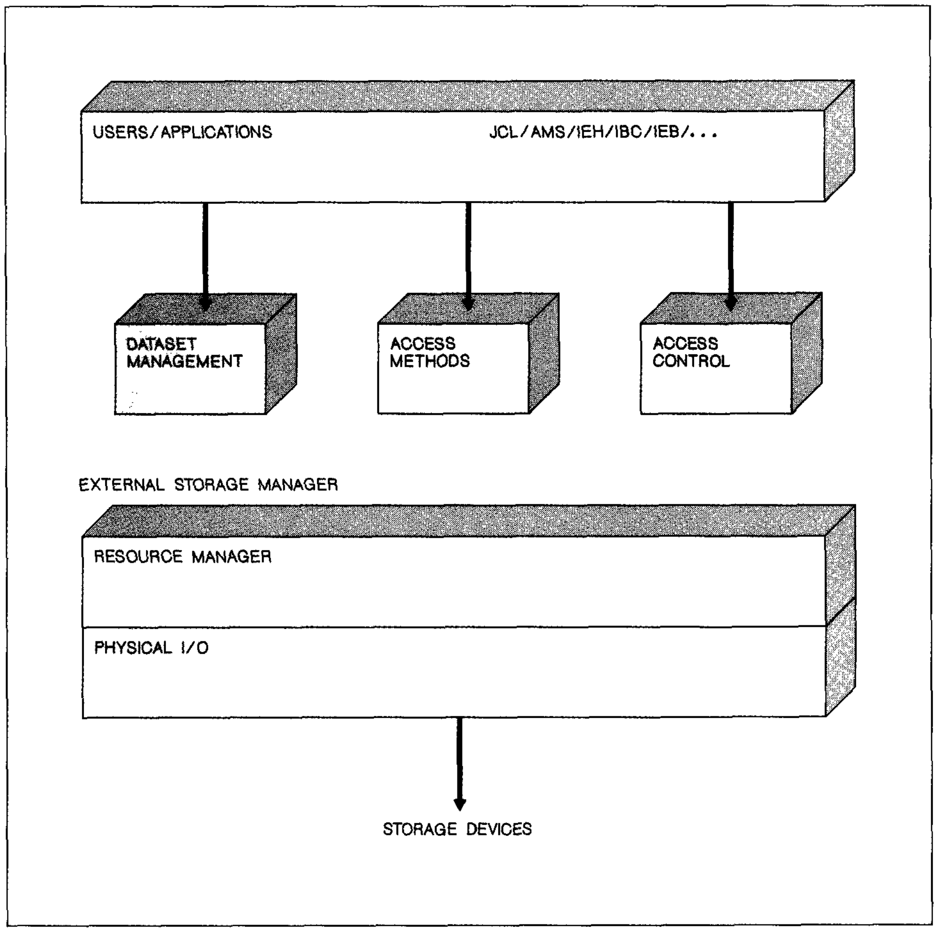
management, and device install management that are shown in Figure 3. These functions become system managed and controllable by definitions and policies established by a storage administrator.

To produce an external storage manager that can control physical storage, it is necessary to remove volume awareness from the user domain and from the facilities and applications used to manage data. These facilities and their interfaces constitute the domain of the *logical data manager* (referred to here as LDM).

The LDM isolates the access methods and dataset management functions from the physical devices and is itself shielded from device characteristics by the ESM. The LDM is the focus for the logical view of dataset format and content, record management, authorization and security, retention and disposition, and manipulation and reporting. It also provides facilities for the following storage administration functions: the definition and maintenance of logical views of storage, the reporting of logical storage status, the limiting and tracking of logical storage, and the access to ESM functions for physical storage monitoring and control.

User applications should be unaware of the existence and function of the LDM. That is, applications and

Figure 4 Role of external storage manager



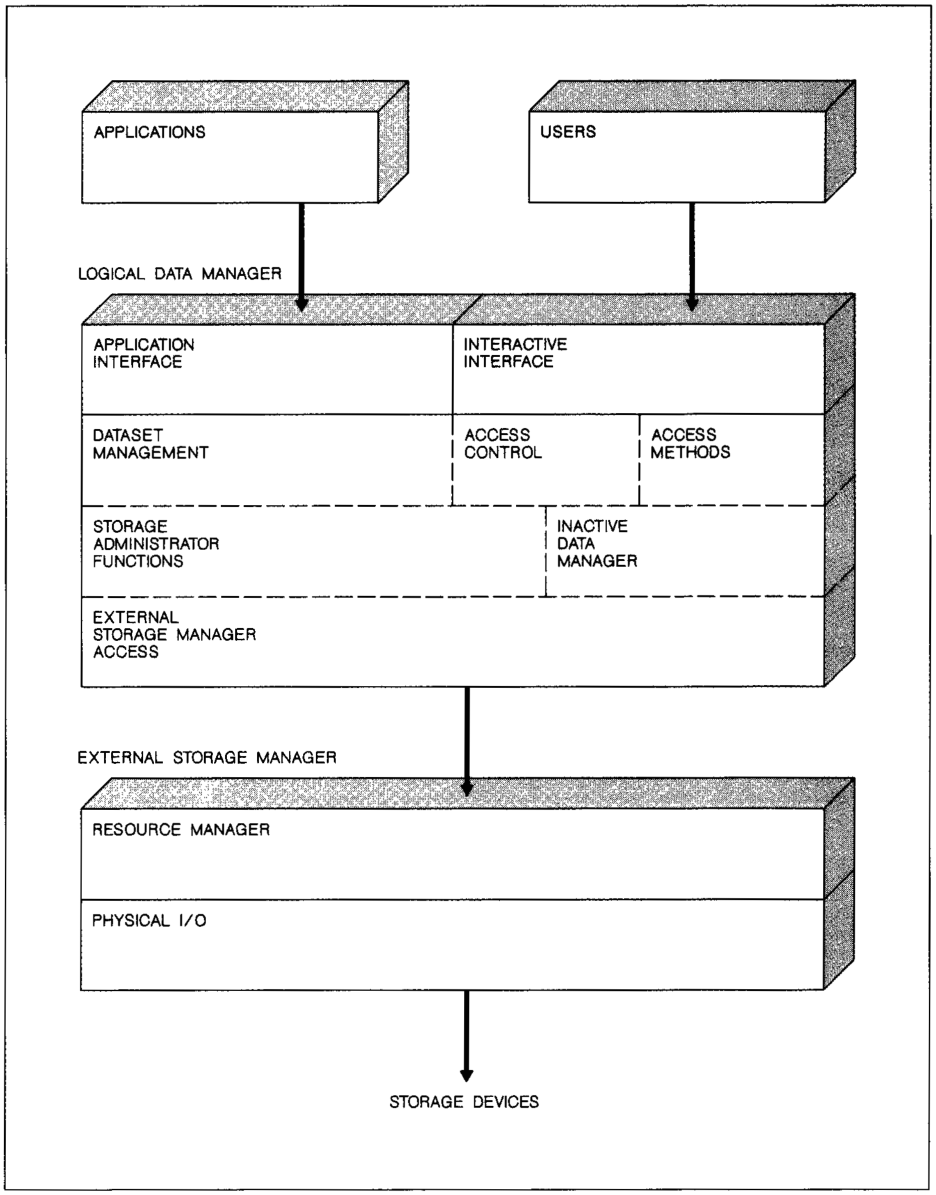
utilities that have no device dependencies continue to function as they do today; the application interface remains unchanged. Adaptors can be provided to the logical data manager to enable access to any new data format used by the LDM and to ensure compatibility at the access method level.

The LDM also provides an interactive interface, termed the interactive storage management facility (ISMF), for user and administrative access to its functions and facilities. Figure 5 shows the functional components of the logical data manager and its relationship to users, applications, and the external storage manager.

The logical data manager's focusing on the application-oriented domain and the external storage manager's focusing on the physical storage-oriented domain, constitute the components of the storage management subsystem. These subsystems communicate with each other using a common virtual view of storage.

Storage management constructs. To aid in the realization of the objectives of the separation of logical and physical views of storage and the simplification of the user interface, system-managed storage introduces new concepts and externals to describe and control storage and data. These constructs seek to minimize user awareness and specification of storage, abstract and centralize control of storage configuration and utilization, formalize the management of data disposition, simplify the specification of data requirements for allocation, and provide a focus for storage management products. Through the use of the constructs, current data and storage externals

Figure 5 Role of logical data manager



may be simplified, positive control over storage may be initiated, and device dependencies may be eliminated from applications.

The constructs provided for system-managed storage are storage group, storage class, management class, and data class.

A *storage group* represents a dynamic pool of external storage volumes. A volume may belong to one and only one storage group. However, multiple volumes may belong to the same group. Logically, a dataset is constrained to reside in one storage group, although it may span the volumes within that group. The storage associated with a storage group will be managed by the storage management subsystem.

The content of storage groups is installation determined. A task of the storage administrator is the definition and manipulation of these groups, that is, the mapping of volumes to storage groups. There are potentially many reasons for configuring storage groups one way as opposed to another. In general, the mapping should be based on environmental considerations, such as the following: device, string, and/or control unit isolation; similar device types (a current DFSMS restriction); common device characteristics (e.g., magnetic DASD versus optical); physical location and/or security; system accessibility, switchability, and/or paths; common power bus; or based on business need. Besides these physical constraints, storage groups may also be created for the purpose of maintaining data isolation. Thus a storage group configuration may be established to keep one kind of data from coresiding with another (e.g., test versus production) as disaster protection.

In the current implementation of DFSMS, the storage group contains parameters for describing the following:

- The set of unique volumes ("pool") associated with the group
- The status of each volume with respect to the group (e.g., available for new allocations, available for read/write only, not available)
- The set of systems in the complex that can access the group
- The manner in which each system can access the group
- Space allocation thresholds
- Migration, backup, and dump characteristics
- Virtual I/O (VIO) limits for datasets associated with the group

The storage group construct is not specifiable or manipulable by programmers or end users. Indeed, its existence does not directly affect the way users manage data, because it is externalized only to the storage administrator. Changes to storage group definitions may require the movement of data to conform the new parameters and the physical data location.

It is anticipated that an installation will require few storage groups to be defined. The actual number will depend on the degree of pooling separation desired or perceived as necessary.

The *storage class* construct represents the lowest level of storage visibility available to users. It is a logical storage construct, in that it denotes the desired level of storage service, based on data requirements. Every dataset that is to be system-managed must be associated with a storage class.

The level of service associated with a storage class is essentially an agreement between the system (through the storage administrator) and the user with regard to the utilization of storage for data. At a minimum, it reflects the logical performance and availability requirements of the data in conjunction with the capabilities of the storage.

The performance service level within a storage class represents the response-time requirement of datasets

Availability, as a service level attribute, implies a degree of reliability and accessibility to the data.

associated with the class. In the current implementation of DFSMS, performance objectives are defined through parameters indicating the following:

- Desired millisecond response time for data using direct access
- Direct access READ/WRITE biasing, reflecting whether the data are more frequently read (as with catalogs) or written (as with logs)
- Desired millisecond response time for data accessed sequentially
- Sequential access READ/WRITE biasing

Availability, as a service level attribute, implies a degree of reliability and accessibility to the data. It reflects the degree to which the system can provide access to data in the light of physical storage outages. In the current implementation of DFSMS, availability is specified as either STANDARD or CONTINUOUS, where the former indicates that normal procedures are adequate and the latter indicates that special procedures, such as duplexing via dual copy hardware, are required.

The relationship between a logical service level represented by a storage class and the physical device capabilities contained within storage groups is a many-to-many mapping, established via a dialog between the storage administrator and the system. The requirements of a single storage class may be met by several storage groups. The capabilities of a single storage group may meet the requirements of several storage classes. Through the establishment of the storage class–storage group relationship, the system has the power to optimize and tune I/O and place data to achieve a best fit from a system throughput perspective.

The storage class is the vehicle through which the service level requirements and capabilities of storage are materialized to the user. It replaces today's externally specified physical storage (i.e., UNIT=, VOL=, MSVGP=) with a logical specification of requirements (e.g., STORCLAS=CRITICAL). Changes to storage class definitions apply retroactively to associated datasets upon their re-allocation, in the current DFSMS implementation.

Through defaulting mechanisms to be discussed later, users are not required to have even an awareness of storage classes. They are, however, able to affect which storage class is selected for a specific dataset. As with storage groups, the number of storage classes defined in an installation is expected to be small. This is primarily due to the reasonably few distinct levels of service that can be effectively materialized. For example, given eight distinct levels of performance and two distinct levels of availability, a maximum of sixteen (8×2) unique storage classes may be defined. Not all sixteen storage classes may be interesting to a given installation.

The *management class* construct represents the criteria by which the life cycle of data is managed. Specifically, the management class contains the policies that control data migration, backup, and retention. The migration and retention policies also contribute to the management of space.

At any point in time, a dataset may be viewed as in one of the following states:

- Active—available to an application for processing
- Less active—candidate for migration
- Inactive—a backup copy or archived

Datasets that are *active* reside in storage groups, the primary volumes of the storage hierarchy. These datasets are directly accessible to applications, and are referenced through integrated catalog facility (ICF) catalogs. Less active datasets that are migrated are moved to other volumes in the storage hierarchy, though they are still considered active. While migrated datasets are still referenced through the catalog, the system must stage them up to primary volumes for application access. This staging can be accomplished automatically (through dataset referencing) or manually (by user command).

The majority of datasets should follow relatively few standard management controls throughout their life cycles.

Inactive datasets are viewed as outside of system-managed storage. Manual intervention (e.g., user command, volume mounting) is required to retrieve inactive data and reallocate the datasets to primary storage. They are not referenced via the catalog of active data, but through an auxiliary catalog or control dataset.

The definition and maintenance of management classes are tasks of the storage administrator. Changes to management-class definitions retroactively affect active datasets associated with that class. In the current implementation of DFSMS, management class parameters describe the following:

- Migration age options associated with storage hierarchy levels
- Expiration and retention date defaults and limits
- Backup frequency, number of versions, and copy retention
- Generation dataset options
- Partial space release control

As with the two previous constructs, it is not anticipated that an installation will define a large number of management classes. The majority of datasets should follow relatively few standard management controls throughout their life cycles, with the remainder requiring some specialized but easily containable handling.

The *data class* construct represents a template for the allocation and definition of datasets. It provides a mechanism to simplify the externals necessary for specifying dataset organization, record format and

Space requirements can be specified in bytes, thereby eliminating device dependencies, such as tracks and cylinders.

length, block size, and security, as well as other parameters used for dataset definition. In addition, it allows the dynamic creation of virtual storage access method (VSAM) datasets without the need for a separate job step.

It is anticipated that the data class will also provide some level of project management and control for the common specification of attributes for a collection of datasets with similar needs. The use of a data class precludes the necessity of specifying individual parameters on data definition. Changes to data-class definitions are not retroactively applied to existing datasets.

In the current DFSMS implementation, the following data-class parameters are included:

- Dataset type (e.g., keyed, sequential, or partitioned)
- Record length
- Space requirements
- Expiration and retention dates
- VSAM dataset allocation options (key length and offset)

The space requirements can be specified in bytes, thereby eliminating device dependencies, such as tracks and cylinders. Several of the data-class parameters may be specified by users explicitly (e.g., via JCL or TSO), overriding those in any associated data class.

We may summarize the new constructs as those intended to provide a simplifying, unifying, and centralizing approach to the management and control of data and storage. They help to raise the level of data and storage specification from the vagaries of the physical world to the requirements of the logical environment.

Each instance of a construct created by an installation has a unique name. Three of the constructs—data class, management class, and storage class—are externalizable to users as parameters on JCL, TSO, access method services (AMS), and dynamic allocation via new keywords. By providing increased capabilities for the implicit determination of the constructs, additional control and simplification over construct usage may be achieved. In particular, it will be possible for an inexperienced user to allocate and use data on storage by merely specifying the dataset name and perhaps a disposition (DISP) parameter to indicate that the dataset is new or existent. The applicable constructs can be determined by the system from information supplied by the administrator for that user.

Throughout the evolution of system-managed storage, the constructs will remain durable in their mode of definition and use. It is expected that the attributes, policies, and characteristics within the constructs will change over time as system-managed storage evolves. However, the externals and their perception by users and administrators will remain unchanged.

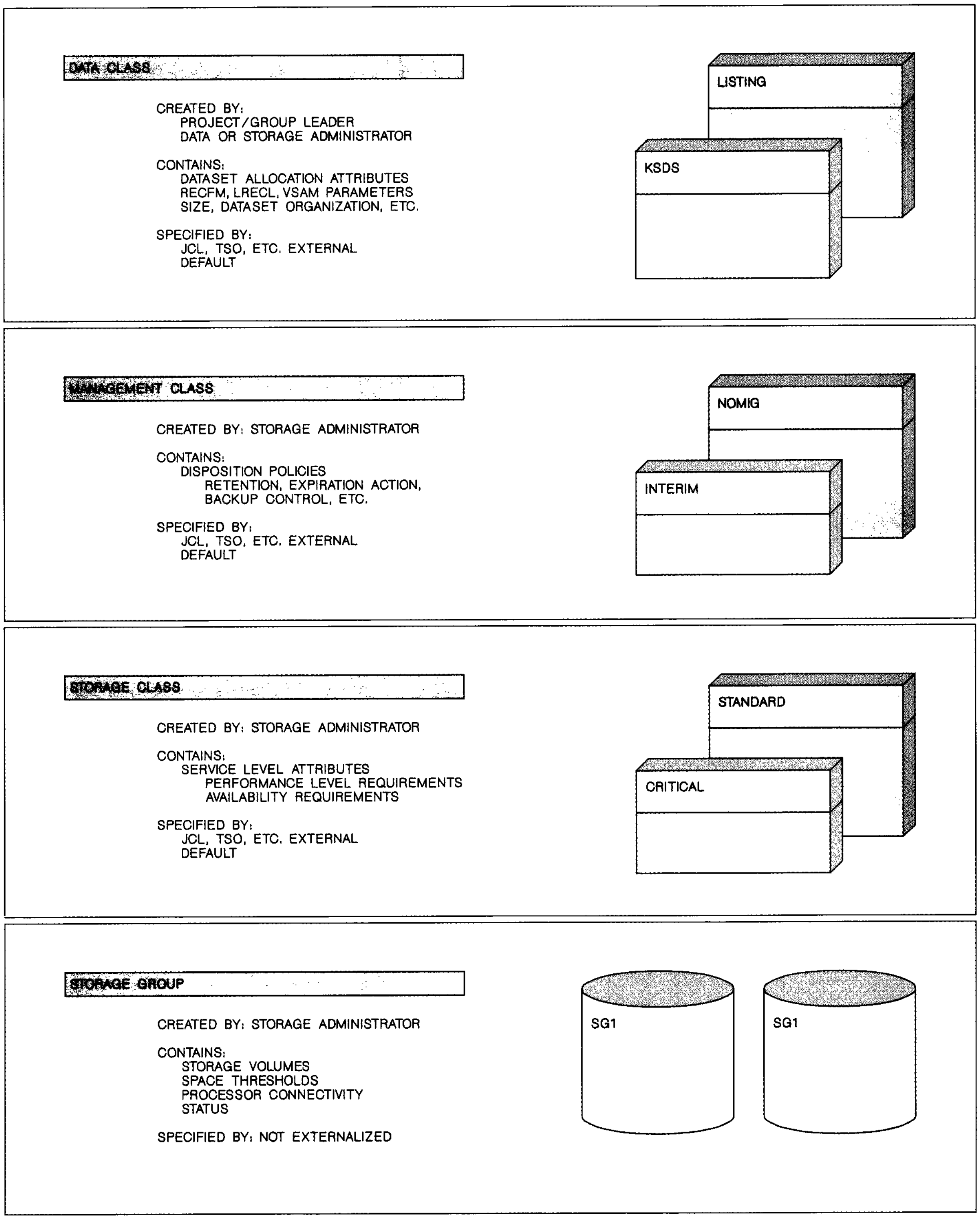
Figure 6 summarizes the constructs and their characteristics.

Storage management facilities. In support of system-managed storage, additional facilities are provided in DFSMS. These facilities enable the user, the storage administrator, and the system to manage data and storage more effectively than in the past.

The *interactive storage management facility* is an important concept within the approach to data and storage management that provides a user-friendly interface to the strategic functions and facilities. This interactive interface has a two-fold purpose. It supports the task-oriented use of the functions and facilities by both end users and system administrators, and it enables the orderly evolution of facilities "under-the-covers" while maintaining consistent and familiar access to the functions.

The first purpose is accomplished through a full-screen, menu-driven set of dialogs to perform the

Figure 6 Constructs



tasks of data management and storage management. The tasks are represented via tree-structured access to data and function, strongly weighted to reduce key strokes by users and provide the information necessary to operate intelligently on the data presented in a straightforward, sequential manner. Online HELP facilities and tutorial information are provided to minimize the need for external documentation and error determination.

A tenet of system-managed storage is that fewer personnel and/or lower skill levels will be able to control and fully utilize the system.

The second purpose is performed by the provision of a common, consistent set of external commands over the set of products that provide the concept of system-managed storage. As the products grow and change to realize the key management functions, the commands to control those products are intended to remain constant. The user should be able to remain in touch with the dialogs, with consistent syntax and displays, and remain unaware of the facilities being invoked to accomplish a task.

The interactive storage management facility provides separate dialogs and the authorization checking to enter those dialogs for the end user and for the storage administrator. The end-user dialog focuses on dataset maintenance and manipulation, dataset status and information, and logical resource utilization. In each case, the user is presented with the information and commands that are authorized for use. For those commands with potentially disastrous consequences, verification will be requested before execution is performed.

The system administrator dialog focuses on storage management tasks, such as logical and physical storage status and information, logical and physical storage utilization and control, and construct definition and manipulation. Again, authorization to obtain this information and perform these tasks is checked prior to display and/or execution.

The intention of the interactive interface is to simplify the tasks necessary to create, modify, and manipulate the rather complex environment of storage and data management. A tenet of system-managed storage is that by reducing the complexity of the tasks required, fewer personnel and/or lower skill levels will be able to control and fully utilize the system.

The interactive storage management facility (ISMF) is that part of the storage management subsystem that supports interactive access to storage and data management functions. Its purpose is to simplify the interface for the management and control of data and storage and to provide consistent, integrated access to the products and functions that comprise DFSMS.

ISMF consists of base support and applications. The base support is a set of service facilities upon which the applications may be built. It utilizes the functions of the interactive system productivity facility (ISPF) to synthesize panel structures and provides basic screen-oriented functions for the selection and manipulation of data to be displayed on the panels. The ISMF applications built on this base focus on task-oriented dialogs for data and storage management. In the current DFSMS, this support includes dialogs to perform the following functions:

- Display information and operate on datasets
- Display lists of datasets
- Define, alter, and delete constructs
- Display lists of constructs
- Display contents of constructs
- Move data
- Modify the storage management environment
- Report on the status of data and storage

The display and reporting functions can be restricted to particular criteria (e.g., dataset size, specific construct, etc.) to reduce the amount of information returned and to allow management by exception.

Authorization facilities, via RACF[3] or its equivalent, allow the restriction of access to the dialogs, commands, information, and certain keywords displayed, based on the user requesting the function. Thus centralized control of construct definitions and usage may be enforced. Transitions among panels supported by different products will be smooth and will have common display and action formats.

An *integrated catalog facility* was developed as a consequence of a need for logical data management and system-managed storage to locate datasets independently of the physical devices on which they reside. To that end, all active datasets residing on managed storage must be cataloged. The integrated catalog facility (ICF) is the main vehicle through which datasets are cataloged, located, and accessed under system-managed storage.

Consistency in the cataloging and locating of datasets is a key concern. This implies doing away with such user-oriented facilities as JOBCATS (job catalogs), STEPCATS (job step catalogs), and catalog parameters on commands. Rather, the function of determining in which catalog a dataset is to be placed is administratively controlled and enhanced through a multilevel aliasing capability. Removing catalog awareness from the user domain will greatly simplify the task of effective catalog management.

The implications of forced cataloging of all datasets raises issues concerning duplicate dataset names and the system-build process and for processing datasets in generation data groups (GDGs). For reasons of repeatability, catalog determination will be based on dataset names. Thus dataset naming conventions and name uniqueness is required, at least within the scope of a job entry subsystem (JES) complex. The problems associated with system build, insofar as catalog and dataset names are concerned, are simplified through the addition of a "SYS%" facility that enables aliasing of SYS1 datasets.

Datasets associated with a GDG have special processing needs, because they can be referenced by relative generation number and can be "rolled-off" when a GDG exceeds its generation limit. The catalog maintains special information about generation datasets, when they are created and when they "roll off," to ensure that references by relative generation number are correct and that rolled-off datasets are kept until they either expire or migrate.

The catalog is the source for information needed by other DFSMS facilities on a dataset basis. This includes the constructs associated with the dataset as well as construct-derived dataset information. This focus on the catalog centralizes the access to data (and data about data), thus simplifying paths to and functions on datasets by providing a single source of dataset information. It also implies the requirement for high performance through the catalog and places increased need for the efficient and effective recovery of catalogs and the information they contain. ICF ensures that performance and recoverability are adequate and commensurate with the vital functions associated with cataloging. In the MVS/ Enterprise Systems Architecture (MVS/ESA™)[4] environment, catalogs may be placed in data spaces for increased performance.

As discussed previously, the data class, storage class, and management class constructs may be explicitly specified by users for new dataset allocations. However, to simplify the specification of constructs for new datasets, provide an interim migration path from existing externals to the new constructs, and enable administrative control, DFSMS offers facilities for implicit construct determination. These include the ability to default constructs based on user identification, to translate old externals to the new constructs without requiring JCL changes, and to override user-specified constructs.

User-based defaulting is a facility for the implicit determination of constructs under the control of RACF, which is an IBM product for identifying users and their rights within the system. It also has the ability to select constructs based on the identity of the dataset owner. The owner of a dataset is based on installation-defined naming conventions. It may be an identified user or, for datasets owned by a group of users, it may be the identified group. RACF also supports the placement of a specific dataset owner in the user profile to be used instead. In either case, RACF maintains a profile of the dataset owning user (or group), detailing the system-rights of that owner. As a part of that profile, and at the installation's option, RACF will maintain user-specifiable default values for the storage and data management constructs.

This user-based defaulting of constructs simplifies the construct specification requirements of users who always follow a standard, owner-determining dataset naming convention. It also enables users who frequently follow such a standard to specify constructs only for exceptional situations.

Defaulting constructs based on dataset owner (as opposed to the active user) was selected for repeatability reasons. That is, a user who is defining a dataset on behalf of another user should obtain the receiver's defaults as opposed to his own. This provides consistency in defaulting for subsequent data definitions, independently of who (or what, in the case of a utility program) is performing the definition.

Automatic class selection is the automatic selection of constructs to be performed by administrator-defined *filters*. A filter is a collection of specific and/or generic criteria that, when matched, results in the determination of a construct. The criteria include partially or fully qualified dataset names, job/task and application information, existing define data (DD) statement parameters (unit, volume, dataset attributes, etc.), and others. The current DFSMS implementation has over thirty variables that can be examined. The specification of the criteria is through a CLIST-like language that provides full construct selection control.

When a new data definition is encountered, automatic class selection is invoked. This facility can examine the RACF defaults, existing parameters, and other information for filter criteria satisfaction. The consequence of matching a filter is to create the construct to be associated with that data definition. Separate class selection routines are used for each of the constructs. Thus, based on minimal information specified externally, the applicable data class, management class, storage class, and storage group may be determined.

The following is the sequence for applying automatic class selection for construct determination:

1. Apply the data class selection filters
2. Apply the storage class selection filters; if no storage class is determined, the dataset will not be system-managed and no further class selection will occur
3. Apply the management class selection filters
4. Apply the storage group selection filters; at least one storage group must be selected

Each class selection routine has an associated installation exit that is invoked for ultimate class determination.

Automatic class selection should not be viewed as a long-term solution to the conversion of externals. Rather, as time goes by, the use of old externals should atrophy, and the new constructs used or defaulted take their place. As a conversion vehicle, automatic class selection represents a toleration or bridging technique to the new externals. In the long term, the filters will enable the defaulting of the constructs so that external specifications can be minimized. The dataset name may be all that is necessary externally to define a new dataset. This is especially helpful for the majority of users for whom the constructs can be simply determined and who would view their explicit specification as an unnecessary complication.

Storage resource administration. A key aspect of system-managed storage is increased administrative and system control over data and storage resources. In support of that objective, facilities for the management of those resources and the formalization of the role of the storage administrator as the controller of those resources have been introduced.

Policies and standards, once created through memos and guidelines, are now embodied in the constructs of system-managed storage.

Storage administrator role. As was discussed previously, part of the problem and complexity of today's storage management is the decentralized control over the resources. Users, through existing externals, control storage for the purpose of managing data. This decentralization often precludes those responsible for managing the system as a whole from effectively tuning for performance and space utilization. Additionally, MIPS are often wasted or unavailable, because of individual over-emphasis on backup and availability of data by dumping storage.

DFSMS seeks to centralize the control of users and storage, and thereby eliminate these problems, through the introduction of the formalized role of *storage administrator*. Under DFSMS, the administrator has the responsibility of informing the system of the policies and guidelines by which storage and users should be managed. At the administrator's disposal are the information and facilities through which tuning requirements, resource utilization authorization, and data disposition processes may be defined to and managed by the system. A user who is identified to the system via RACF or equivalent as a storage administrator, has authorization to the ISMF dialogs and commands through which control of the storage resource may be realized.

The functions that may be performed by an authorized storage administrator include the following:

- Definition, modification, and deletion of constructs
- Assignment of constructs via automatic class selection
- Authorization of users to specific constructs
- Definition and maintenance of the active storage configuration
- Tracking storage utilization at the logical (user) and physical (capacity-planning) levels
- Movement and placement of data to optimize utilization

The separation of logical and physical views of storage enables the administrator to inform the system of modifications to the physical resources via storage groups and storage class–storage group relationships without affecting the application's view of the data. Assisted by the system, the administrator will also be able to do the following:

- Define and control access to storage service levels via storage classes
- Define and control policies to control dataset life cycles and space via management classes
- Dynamically view the status of the storage resource via ISMF

Policy establishment. The storage administrator's primary role is in interacting with the system to define, maintain, authorize, and control the content and use of constructs and facilities. There is another task to be performed of equal importance: the effective establishment of policies and standards to be implemented. These policies and standards, once created through memos and guidelines to users, are now embodied in the constructs of system-managed storage.

Before introducing policies into the system through which data and storage will be controlled, the administrator must determine which policies need to be created. This can be accomplished through the establishment of service-level agreements with the user groups, to bridge from user requirements to system delivery. A service-level agreement constitutes a contract between the users and the system by way of the administrator for the services expected by the users and the provision of those services by the storage subsystem.

The evolution of system-managed storage implies a revolution in the way external storage and data are managed.

While there is no universal formula for a service-level agreement, it should contain the following items:

- Free space and space-usage requirements
- Availability requirements, including backup, recovery, and disaster backup/recovery
- Data retention and archive requirements
- Performance requirements
- Security requirements
- Special considerations for particular datasets
- Contingencies
- Reporting of service levels
- Education and communication services
- Responsibilities of all parties

A similar agreement should be negotiated between storage administration and operations that includes the following:

- Reporting of DASD failures
- Reporting of job failures
- Schedules for data and storage management jobs
- Responsibilities of all parties

Once these service-level agreements have been negotiated, the storage administrator can define the policies to the system. As needs change, the service-level agreements can be modified with concurrence of all parties, and changes can then be incorporated into the system.

Evolving toward system-managed storage

The evolution of system-managed storage implies a revolution in the way external storage and data are managed. It entails new facilities and functions for system-managed space, performance, availability, and device install, new components in supporting products, new constructs for using the concepts, and the formalized and centralized role of storage administration. It is technically feasible to design, develop, and target this solution for a single product release. However, that approach has several negative aspects. It would take a long time to produce such a single product release, well beyond the time frame when some customers would have encountered the problems. Such an approach might also result in idle code in dependent products and cause a major conversion/migration bottleneck in installation and use. It could be disruptive to current approaches to storage management, especially with respect to the use of other products that are unaware of the encapsulated solution and need to be used in conjunction with it. The approach might also entail a massive re-education effort, for both installation management personnel and end users.

System-managed storage includes the concept of evolving the solution over time, multiple releases, and utilizing existing program products as the basis for this evolution.

In view of these considerations, system-managed storage includes the concept of evolving the solution over time, multiple releases, and utilizing existing program products as the basis for this evolution. This direction utilizes an outside-in or top-down approach. That is, growth and change move from the user interface down to the functional internals. This approach begins with the simplification of user externals, while maintaining current technology and facilities "under the covers," and continues with functional enhancements and modifications to existing functions without impacting the externals. It ultimately replaces the underlying technology with a new approach to the management of storage. Thus, user interfaces are formalized early, re-education can begin while functions are still familiar, current practices may evolve at the installation's own pace, and the key products can grow toward the common solution, while still supporting their current data and function.

Key products. IBM produces several program products used for data and storage management that include the following:

- MVS Data Facility Product (MVS/DFP)[4]
- Data Facility Hierarchical Storage Manager (DFHSM)[5]
- Data Facility Data Set Services (DFDSS)[6]
- Resource Access Control Facility (RACF)[3]

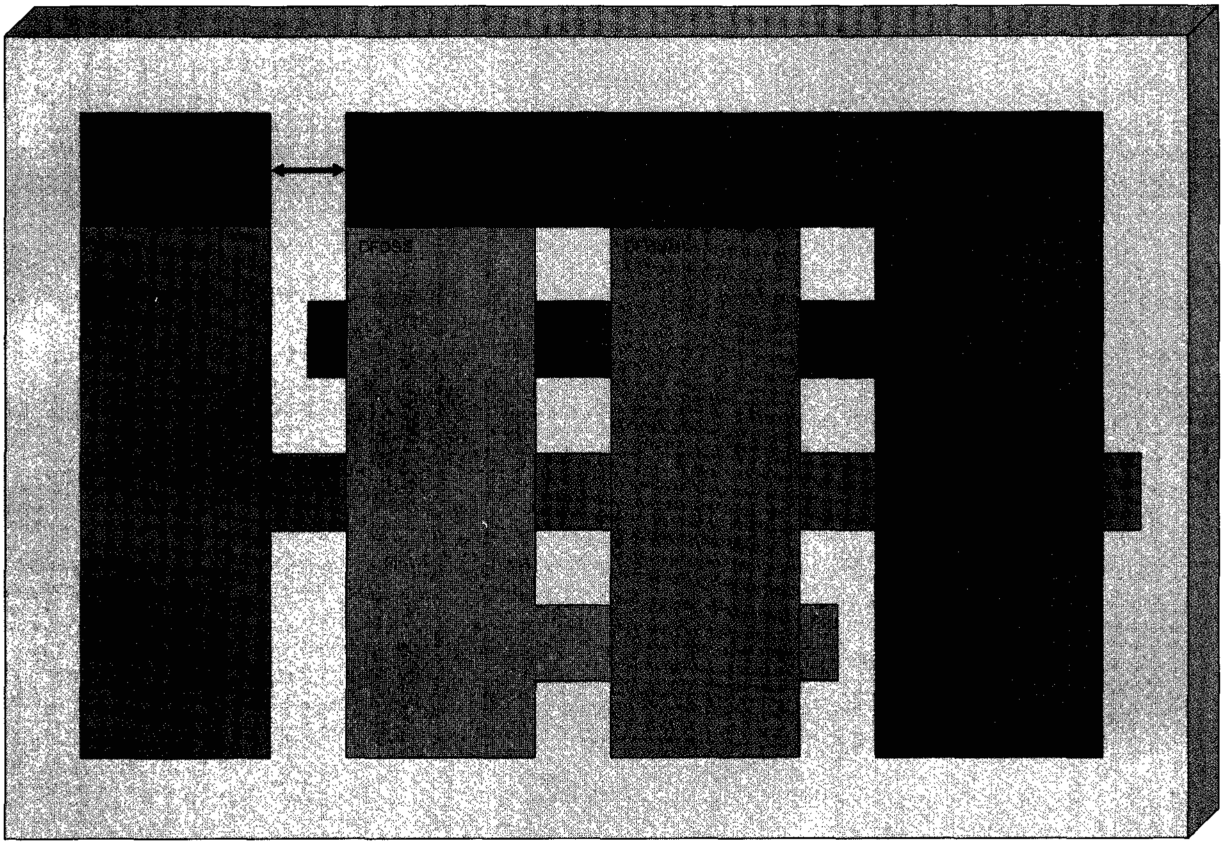
These products, together with the MVS System Product, system and storage hardware, and the MVS Storage Management Library,[7] form the base for the evolution of system-managed storage.

In the early evolution stages, support for system-managed storage existed in the extended architecture (MVS/XA™) environment. However, due to the processing power, environment, and functions that the solution requires to effect system-managed storage, the current stage exploits Enterprise Systems Architecture/370™ for DFSMS support. Future stages will continue functional evolution on the current base. In addition, other major IBM program products such as the Information Management System (IMS) and Database 2 (DB2) will continue to utilize the concepts and facilities of DFSMS in support of their function. Each of the key program products just listed has the following major role in system-managed storage and as illustrated in Figure 7.

- MVS/DFP is the manager of active data and storage.
- DFHSM is the manager of inactive data and storage.
- DFDSS is the high-speed data mover for, and provides facilities for conversion to, system-managed storage.
- RACF is the control facility for storage resource authorization, user identification, and data security.

The interaction among functions of these products comprise DFSMS. The synergy among DFSMS, hardware, and the policies, practices, and procedures

Figure 7 DFSMS key product relationships



initiated by the storage administrator make up the system-managed storage solution.

Migration/coexistence. Because the approach is one of evolution, it is critical that migration of data and externals be relatively easy and cumulative across the stages. In addition, each stage must provide for the coexistence of data and storage not under system management, or that was defined to (and desired to be maintained by) previous stages.

As was mentioned earlier, it is anticipated that over time existing externals (JCL, TSO, etc.) will atrophy and will be replaced by the new constructs (data class, storage class, management class, storage group). Through the defaulting mechanisms, they may disappear altogether. A GUIDE survey indicated that more than 50 percent of JCL and CLIST statements are normally changed within a four-year time span. With sufficient justification and motivation, this could occur in two years. The combination of explicit construct specification for new JCL, administratively controlled automatic class selection for existing JCL and construct defaulting via RACF, should be sufficient to provide an orderly, installation-policed migration and coexistence of externals. Throughout the evolution, it will be possible to run jobs requiring a mixture of the new externals for data under DFSMS control, existing externals filtered for use with DFSMS-controlled data, and existing externals referencing non-DFSMS controlled data.

The migration of data to DFSMS may be accomplished in several ways. Normal old-master/new-master-type processing will be sufficient, in many cases, to create data under DFSMS. That is, the replacement of an existing dataset with a new version can result in the new dataset being system-managed. Similarly, the normal recall cycle for migrated datasets may be used to bring data under system control when it is reallocated by the system, for example, by DFHSM. DFDSS provides a utility path for the conversion of unmanaged volumes to system-managed volumes. It will also be possible, through DFDSS and DFHSM, to move or copy data that are on a DFSMS-controlled volume to a volume that is not under DFSMS control.

Functional staging. System-managed storage will be realized in several functional stages. Each stage may represent several releases of one or more of the key products. The first stage of system-managed storage focused on simplifying the interface for data and storage management. It also provided interim relief for some of the current storage management problems. This first stage resulted in the ISMF dataset and volume applications, catalog recovery enhancements, and virtual storage constraint relief. It also included disaster backup support, tape migrate/recall, DFDSS automatic dump capability, and ISMF support for DFHSM; logical dump, space release, partitioned dataset compression, and ISMF for DFDSS; and naming convention support, tape dataset support, erase-on-scratch support, and third-party authorization checking in RACF.

The second stage of system-managed storage began with the introduction of DFSMS, which initiated the separation of the logical view of storage and data (based on application requirements) from the physical view, centralized control of the storage resource, and formalized the role of the storage administrator. It also introduced the durable externals for control over space, performance, availability, and device install management.

Future stages of system-managed storage plan to look to changing the structure of data storage on external storage (common data format). This will enable continued functional enhancements in hardware and software to provide increasing efficiency and control for data and storage. The continued emergence of an external storage manager and a logical data manager is anticipated, together with an extended, system-managed storage hierarchy. This will tend to give the appearance of an infinite store for data. Future stages will also enhance the management of data and storage in distributed and multi-system environments, and for remote storage of inactive data.

Concluding remarks

System-managed storage is a solution to the problems of data and storage management. It provides benefits for both the users and the enterprise as a whole. The application programmer can now focus on the data and the applications using that data without the need to be concerned about how the data are stored. The requirements data has on storage can be communicated through logical, data-oriented policies specified by the storage administrator. These simplified externals also increase personal productivity by reducing the impact of storage idiosyncrasies. Service-level agreements between administrators and users and the constructs that embody those agreements are living documents to aid in communicating these requirements to the system.

With the system acting in concert with the storage administrator, the complexity of managing storage can be reduced, resulting in the realization of increased productivity for support personnel. The centralization of control over storage further reduces the need for uncontainable personnel increases and simplifies the tasks associated with storage management. Through increased utilization of storage, installations will experience less wasted space, thereby allowing more effective use of storage and free space for the new data required to support business growth.

System-managed storage has arrived. It has evolved from the user-managed status of the past to the administrator-controlled environment of today's DFSMS. The evolution will continue toward the system-controlled, system-managed environment of the future.

Acknowledgments

I would like to thank Lynn Yates, Lansing Dibbern, Norm Chase, Bill Tetzlaff, Jimmy Strickland, and John Tyrrell for their review and helpful comments for this paper. Many contributions from many people led to this realization of system-managed storage. Space does not permit recognizing them all. I would like to pay special tribute to Clarence Carr and Keith Freeman, who worked so creatively on system-managed storage and are no longer with us to see the results.

Cited references

1. *The LSRAD Report (Large Systems Requirements for Application Development)*, SHARE Inc., Chicago, IL (December, 1979).
2. *Requirements for Futures of Storage Management*, GUIDE Position Paper GPP-106, GUIDE International Corporation, Chicago, IL (May, 1983).

3. *RACF Release 8.1 General Information Manual*, GC28-0722, IBM Corporation; available through IBM branch offices.
4. *MVS/ESA DFP Version 3 Release 1.0 General Information Manual*, GC26-4507, IBM Corporation; available through IBM branch offices.
5. *DFHSM Version 2 Release 4.0 General Information Manual*, GH35-0092, IBM Corporation; available through IBM branch offices.
6. *DFDSS Version 2 Release 4.0 General Information Manual*, GC26-4123, IBM Corporation; available through IBM branch offices.
7. *MVS Storage Management Library*, SBOF-1241, IBM Corporation; available through IBM branch offices.

Jack P. Gelb *IBM General Products Division, 555 Bailey Avenue, San Jose, California 95141.* Dr. Gelb is a senior programmer in System Support Products, working primarily on strategy, design, and architecture for storage management. He joined IBM in 1970 in Poughkeepsie, New York, as a member of the future systems group, focusing on systems language design. Subsequently, he was the manager of application development tools in Mohansic, New York; interdivisional software liaison based in Palo Alto, California; and part of an advanced technology compiler design team in San Jose, California. In 1979 Dr. Gelb began working on the strategy and design for future storage management facilities and shortly thereafter became involved with the GUIDE and SHARE user groups in the area of MVS (and, later, VM) storage management. These activities led, in part, to the realization of DFSMS, for which he received an Outstanding Technical Achievement Award. His interests remain in the area of storage management (present and future), and extend to artificial intelligence, language design, and interpersonal communications. He holds an M.S. degree in mathematics and a Ph.D. in computer science from Rensselaer Polytechnic Institute, Troy, New York, and is a member of the ACM.

Reprint Order No. G321-5349.