Object storage hierarchy management

by W. B. Harding
C. M. Clark
C. L. Gallo
H. Tang

The Object Access Method (OAM) component of MVS/Data Facility Product is responsible for the storage, retrieval, and management of objects in IBM MVS/ESA™ ImagePlus™ systems and in other applications. The OAM Storage Management Component is the subcomponent of OAM that provides storage management for objects stored within an object storage hierarchy. Storage management is a cyclic procedure which assures that data are stored in conformity to a policy defined by the data processing storage administrator. During a storage management cycle, the OAM Storage Management Component (OSMC) selects objects for processing based on requirements for backup, expiration, or service level changes. This paper describes the concepts of object storage management using a storage hierarchy that contains DASD and optical disk storage.

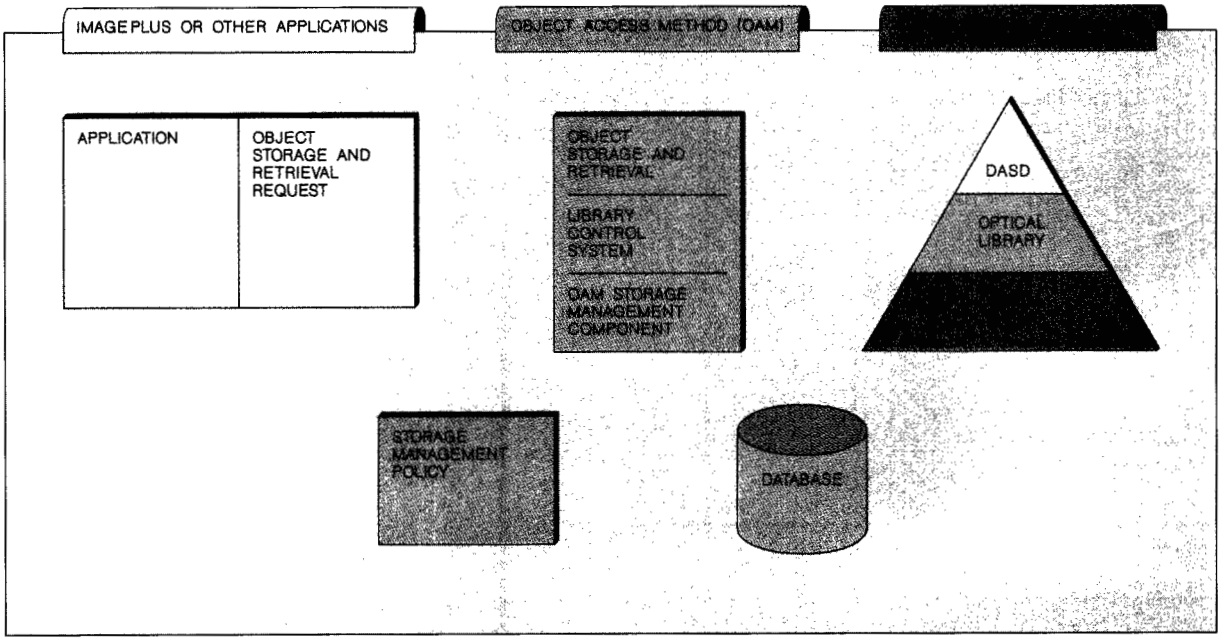
The introduction of images as a data type for computer storage and processing generated a requirement for convenient services to enable the development of applications that use images. It was soon recognized that images were only one of many kinds of data that are currently not subject to computer processing simply because of the volume and cost of storage. Individual items of these data, such as images or voice recordings, began to be discussed as "objects" to identify them as data elements that do not have conventional record structure. An object can be treated as a named bit stream of known length, whose size can vary from a few bytes to megabytes. This application development requirement led to the design of the Object Access Method to provide for storing, retrieving, and managing objects.

The Object Access Method (OAM) consists of three parts. The first of these, the Object Storage and Retrieval subcomponent, provides an application programming interface for storing, retrieving, and deleting individual objects and changing the management policy for those objects. It provides these functions for applications developed to operate in a number of environments on MVS/ESA™ systems.

The second part of OAM, the Library Control System subcomponent, writes and reads objects on optical disk storage, manages the optical disk volumes on which the objects reside, and controls the associated hardware resources.

The OAM Storage Management Component (OSMC), the third part of OAM, determines where objects should be stored, manages object movement within an object storage hierarchy, and manages object expiration and backup based on the storage management policy of the installation. The need for storage management is defined by the large volume of data anticipated, and the range of performance requirements in accessing the data, along with the expectation that the access requirements of the data will change as the data age. The emerging concepts of systems-managed storage were applied and extended to provide the basis for OSMC.

Figure 1 Overview of access to object storage

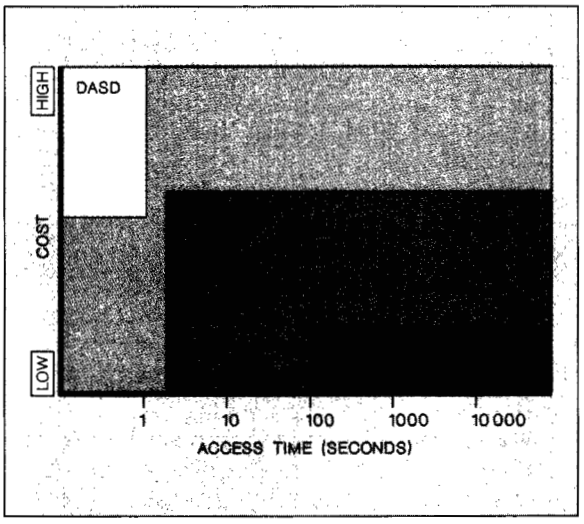


OSMC provides two critical functions in the Object Access Method. It interprets the storage management policy, and it provides an automatic management capability for the stored objects. Figure 1 illustrates the relationship between OAM, an application program, and an object storage hierarchy. The figure shows that an application program requests services from OAM to access the object storage hierarchy through an application programming interface. OAM uses database services and a storage management policy as needed to process the request.

The definition of the object storage hierarchy considered the diverse capacity, performance, and cost characteristics of DASD, magnetic tape, and optical disk. Figure 2 relates the cost and access time of DASD, optical disk, and magnetic tape storage, and shows that DASD has very low access time at a cost that is higher than optical disk or magnetic tape, whereas optical disk storage provides faster access than is available with magnetic tape, at a slightly higher cost.

Many applications require the subsecond access time of DASD for short-term storage so that DASD must be included as the highest level of the hierarchy. However, the volume of object data and the expected cost

Figure 2 Relative cost of storage media



objective for object-processing applications effectively preclude retaining objects on DASD storage permanently. Recent development of optical storage has provided a new large-capacity storage medium with cost and performance characteristics that are

Figure 3 Object storage hierarchy characteristics

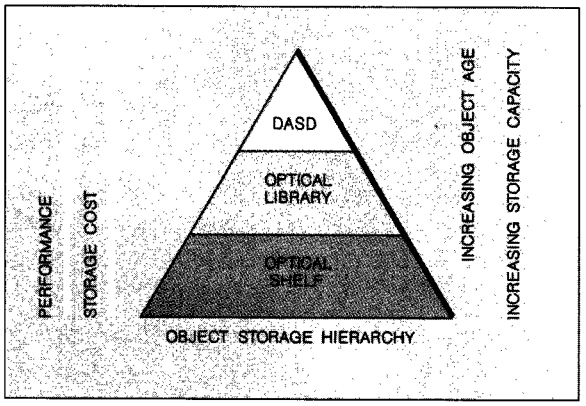
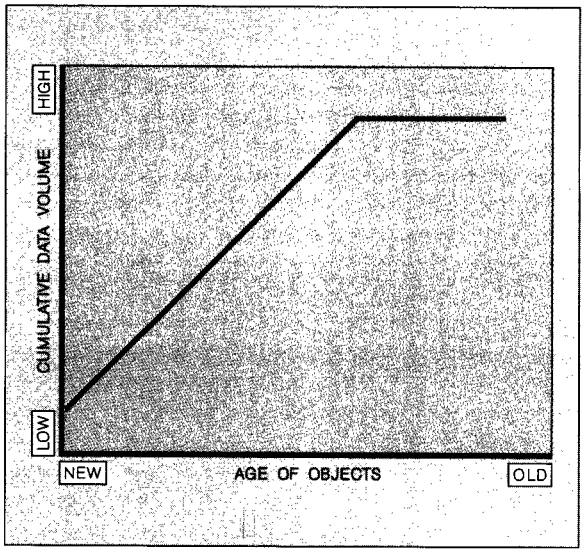


Figure 4 Volume of data



intermediate to magnetic tape and magnetic disk. Optical storage may actually cost more than magnetic tape storage, but its direct access capability makes it preferable for a wide variety of applications. The media life of optical storage is suitable for long-term storage, and the cost makes it possible to store a very large quantity of data at a fraction of the cost of magnetic DASD. For this reason, an optical disk library was selected as the next level in the hierarchy. The third level of the hierarchy is the shelf storage that is used to store the optical disks that have been removed from a library to make space available for storage of additional objects. Although the media cost of magnetic tape storage is less than that of optical storage, when the cost of moving vast amounts of data is considered, it is preferable to retain the data on the optical disks.

To provide the flexibility required for a large variety of applications involving objects, and to provide access times necessary to meet the performance requirements of those applications, OAM was designed to allow partial hierarchy configurations. A configuration may have only one level of storage, or may have two levels, as long as it supports the application requirement and is consistent with the storage management policy.

A system-managed storage hierarchy thus provides both near-term and long-term storage for large amounts of data at a reasonable cost and performance. Optical disk cartridges stored on external shelves and mounted on a stand-alone optical disk drive by an operator can provide essentially unlimited long-term storage with a response time that is manageable. A combination of DASD, library-resident optical disks, and shelf-resident optical disks as an object storage hierarchy can provide a cost-effective means for storing large volumes of object data, while providing reasonable response times for many applications. This is shown in Figure 3. The arrows show that performance and storage cost increase at the higher levels of the hierarchy, whereas object age and storage capacity increase at the lower level of the hierarchy.

Objects in an object storage hierarchy are managed by placing them at the hierarchy level appropriate to their current usage. They reside on DASD when they require short response time, on a library-resident optical disk for moderate response time, or on a shelf-resident optical disk for the longer response time acceptable for long-term storage. Many object processing applications are expected to deal with very large numbers of objects—from hundreds of thousands to millions per day. The daily volume along with the data retention requirement can easily result in an inventory of hundreds of millions of objects. Figure 4 illustrates the cumulative distribution of object data by age. The figure shows that the number of objects in the object storage hierarchy grows until the maximum object retention time is exceeded and then remains constant since the deletion of older objects offsets the storing of new ones. An automated storage management facility is a necessity for dealing

with an inventory of this size in a timely and efficient manner.

The next section on storage management concepts describes the use of a storage management policy to manage objects using logical and physical definitions. Other sections of the paper discuss the performance and design considerations for implementation of both the OAM Storage Management Component and the optical library interface.

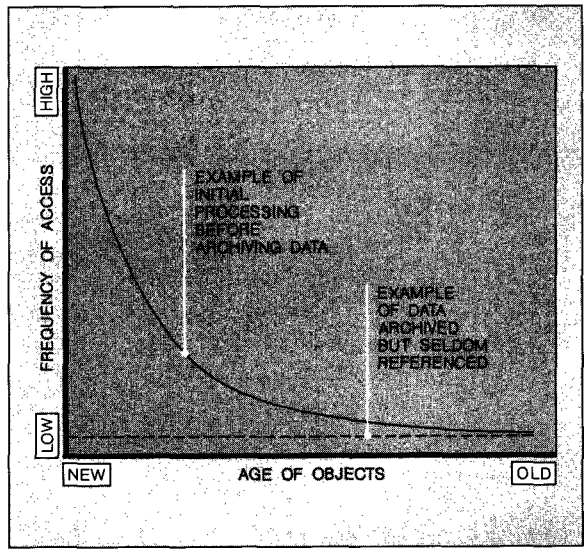
Storage management concepts

MVS/Data Facility Product (MVS/DFP™) Version 3 Release 1 introduced the Storage Management Subsystem (SMS) and the concept of managing the data set storage for an installation according to a policy defined by an administrator. The storage management policy identified the systems being managed, provided lists of characteristics for data sets and grouped DASD volumes, and allowed the storage administrator to define selection routines that associated management criteria with data sets.

For OAM, the storage management policy is a description of the volume and life cycle of object data used for a business application. For example, a business may wish to capture its incoming mail as image data and replace current paper handling procedures with automated processes using images. After initial processing, the data may never be referenced again but must be kept a specified length of time for legal purposes. The response requirements for the current mail may be very high, but those for older documents much less stringent. Another business, or segment of the same business, may wish to capture and archive paper documents that may never be referenced. In each case, there is a need to describe how documents are grouped and to manage how those groups of documents are stored throughout their expected lifetime. These examples are illustrated in Figure 5. The curve represents an application where objects are frequently referenced when they are new, but the reference frequency declines as the object ages. The dashed line represents an archive application where objects are rarely referenced, regardless of age.

The implementation of system management for object storage required additions to the system catalog and to the definition of a storage management policy. The system catalog was extended to provide for cataloging of collections of objects. The storage management policy was enhanced to include the means to define an object storage hierarchy with a relational

Figure 5 Data access frequency as objects age



object directory structure,[1] and to extend the range of performance and management parameters to provide for object placement and movement within that storage hierarchy.

The Storage Management Subsystem (SMS) provides several key functions for OSMC. It provides services to define, store, control, and access storage management policies for objects managed by OSMC. SMS also invokes the automatic class selection routines as requested by OSMC to define or change the class or storage group.

A storage management policy is stored in an SMS source control data set. A complete storage management policy for OSMC has definitions for the storage class, management class, and storage group constructs, and a set of automatic class selection routines. The storage class construct is used primarily to determine whether the performance objective of an object will tolerate storage on an optical volume, and has been extended to include a parameter that defines the maximum allowable access delay. The management class construct is used to determine backup and expiration requirements and control the timing of storage and management class changes. It has been extended to provide for the definition of transition criteria. The storage group construct provides the definition of physical storage, including the designation of the databases and optical libraries that

Table 1 Relationship of OSMC to SMS

OSMC	SMS
Performance objectives	Storage class
Expiration requirement Backup requirement Modify performance objectives	Management class
Placement to a set of volumes	Storage group
Initial assignment to Storage class Management class Storage group Modify assignment to Storage class Management class	Automatic class selection routines

can be used for selected objects. This construct has been extended to include new types of storage groups, definition of optical libraries eligible for object storage, and definition of storage management cycle start times. The automatic class selection (ACS) routines are used to assign storage class, management class, and storage group for a collection, and to validate or override storage class and management class assignments requested by an application. The ACS services have been extended to provide the environments needed for object storage management, and to provide for the collection name, object name convention used for identifying objects.

OSMC usage of SMS constructs is summarized in Table 1. OSMC uses parameter values from the named SMS construct or service to determine the action to be taken for the object being processed.

Storage classes for objects. Storage class allows the storage administrator to define a level of service for data sets and objects. Levels of service are based on the performance objectives for an application. This allows the storage administrator to define device-independent performance levels that OAM can use to determine the best fit to the available hardware.

While SMS manages only permanently mounted media for data set storage, OAM manages an object storage hierarchy which can contain permanently mounted DASD volumes and mountable optical volumes. The mountable optical volumes can reside either inside an optical library where mechanical location and mounting is available or outside an optical library where an operator must locate and mount the volume.

The storage class construct was extended for OAM to differentiate between permanently resident DASD and mountable optical volumes within a library or on a shelf. This allows the storage administrator to expand the performance range for object read or write operations to include a new factor in that performance—the delay the application can tolerate before the initial availability of the data.

For example, an application may require a short response time for new objects but can accept a longer response time for objects as they age and are less likely to be referenced. The storage class is used by OSMC to determine the performance objective for an object and to influence where the object is placed within an object storage hierarchy. Assigning different storage classes to an object over time can be a means to define a series of steps for moving the object to slower, less expensive devices in the object storage hierarchy.

Management classes for objects. Management class parameters are provided to allow a storage administrator to define retention, backup, and class transition criteria for object management.

The retention characteristic of an object is used to determine when an object reaches its expiration date. When an object expires, all reference to that object is deleted.

The backup parameter of the management class assigned to an object determines whether a backup copy of the object is required. Creation of a backup copy of an object may be completed during the first storage management cycle after the object is entered into the system, or it may be deferred until the management class assignment is changed. Because objects do not change, only one backup copy of an object is ever written.

The changing logical characteristics of objects are recorded within OAM by changing the storage class and management class assignments. The process of changing the class assignments is called *class transition*. Storage class transition can imply movement of objects between different levels of an object storage hierarchy since the different levels of the hierarchy have different performance characteristics. Management class transition implies the immediate application of new management criteria. Both transitions

can occur at the same time. The results of a class transition are determined by running the storage class and management class automatic class selection routines at the time the transition occurs. That time is determined by using the object class transition criteria specified in the management class currently assigned to the object. The automatic class selection routines are discussed in a later section of the paper.

OAM recognizes three mutually exclusive class transition criteria: 1) the time since the object was created, 2) the time since the object was last used, and 3) a periodic calendar function.

Time since creation and time since last use are both specifiable in years, months, and/or days. For example, a class transition could be scheduled for 30 days after the object was last referenced or 2 years, 3 months, and 15 days after the object was created.

A periodic class transition can be requested either on the first, last, or *n*th day of the month, the quarter, or the year; or on the first, last, or *n*th day of the *m*th month of the quarter or the year. For example, if today is October 12th, an object whose management class had scheduled a class transition on the last day of the first month of the quarter would undergo a class transition on October 31st.

When the class transition criteria for an object are met, the automatic class selection routines are called to determine the new storage class and/or management class for the object. After the class transition, the object is managed according to its new storage class and/or management class. This includes the possible movement of the object to a new level in the storage hierarchy.

Storage groups for objects. For data sets, a storage group is one or more volumes that are treated as if they were a single volume. OAM defines two new types of storage groups for objects—the OBJECT storage group and the OBJECT BACKUP storage group. An OBJECT storage group defines an object storage hierarchy. An OBJECT BACKUP storage group defines where an optical backup copy can be written.

Each OBJECT storage group has one or more types of data storage device. The DASD level of an OBJECT storage group and the object directory are defined by selecting one of the 100 possible OBJECT storage group qualifiers. The qualifier identifies a database containing an object directory, object storage, and their associated indexes.

The object directory is included as an indexed database within the storage group because it provides the performance, recoverability, and relational capability required for handling the very large number of objects expected by OAM.

The OBJECT storage group can optionally contain optical storage. Its definition can include either one to eight optical libraries or all of the stand-alone optical drives (pseudolibrary) associated with the system. The storage group definition only includes those optical devices to which objects can be written. Once written, an object can be read using any optical drive capable of reading the media.

Differences in the physical geometry of the devices within an object storage hierarchy are handled by OAM by segmenting objects, as needed, for writing to DASD or to optical disk. The only device characteristics of concern to the application developer and storage administrator are the quantity and performance of available storage.

Automatic class selection routines for objects. Automatic class selection (ACS) routines determine SMS classes and storage groups for objects and collections of objects. ACS routines are run for objects in four cases as follows:

1. When the first object is stored in a new collection
2. When a storage class and/or management class is specified on a request to store a new object
3. When an explicit request is made to change the storage class and/or management class for an object
4. When a class transition event occurs for an object

When the first object is stored in a collection, the ACS routines are invoked to assign the collection to a storage group and to assign a default storage class and management class for all objects in the collection. Once assigned to a storage group, a collection and all objects in that collection remain in that storage group.

When an object is stored, its initial storage class and management class can be determined by using the default storage class and management class assigned to the collection, or by specifying either or both of them explicitly and running the storage class and management class ACS routines to approve or alter the request(s). If there are no explicit classes specified on a request to store an object, the ACS routines are not invoked for the object.

There are two ways to alter the storage class or management class assignment of an object after the object has been stored. The user can request a change for that object or a class transition can occur. When a user requests a change for an object through the OAM application programming interface, the ACS routines invoked depend on the type of change requested.

When a class transition event occurs, as defined in the current management class of the object, OSMC invokes the ACS routines to determine if the object should be assigned to a new storage class and management class. The performance level requested under the new storage class definition may require that the object be moved to a different level in the object storage hierarchy defined for its storage group. The management criteria associated with a new management class establish the criteria for the next class transition. They may also cause a backup copy to be made or cause the object to be deleted.

OAM storage management

The implementation of automatic storage management and application processing for objects required the creation of additional linkages and structures including:

- A means for locating very large numbers of objects without overwhelming the existing catalog structure
- A secondary catalog structure for data about individual objects
- An OAM interpreter for SMS storage management constructs
- A way to track and select objects for automatic storage management

The techniques used to address these object storage management requirements are described in the following sections.

Collections. The ability to locate existing objects in storage is fundamental to any data processing system. The facility that carries out the locate function for data sets in an MVS system is the catalog.[2] Typical MVS systems may have tens or even hundreds of thousands of data set entries recorded in a hierarchy of catalogs.

In contrast, OAM must be able to manage hundreds of millions of objects, which if individually cataloged would require an excessively large catalog. Because an inventory of objects already exists for each storage group in its object directory, and because the object directory contains location information for each object, an object can be located simply by looking in the correct storage group directory.

The storage group, however, is an internal storage management concept used by the storage administrator to designate distinct physical devices or data volumes, and is used by OAM to determine which databases and optical volumes are to be used for which objects. It is undesirable to require an application to have knowledge of the storage group assignment of an object. It is even more undesirable to require the end user to specify the storage group assignment of an object, since the end user should only need to be concerned with the logical characteristics of the data. This results in a requirement for an application-related, logical classification of objects that is easily mapped to a storage group. The classification is the assignment of objects to collections. A collection is a set of objects that reside in a single storage group and that have the same default storage class and management class assignments.

The system catalog was extended through the concept of collections to handle the large numbers of objects expected for image applications. A catalog entry is defined for each collection, and provides the default storage class and management class assignments for objects stored in the collection, and the identifier of the storage group where the collection is physically stored. A storage group may have any number of collections. A collection cannot span multiple storage groups.

Every object must belong to a collection because this provides the entry to the procedure to locate the object. Object names must be unique within a collection, but different collections may have objects with the same name. The objects themselves must be unique, not simply alternate pointers to the same object.

A typical application would require one or more collections, depending on the variety of objects used by the application and the management requirements of the objects. If objects used in different applications are assigned to the same collection, a change in performance or management requirements for objects used by one application would also affect objects used by the other.

The object directory. As noted in the discussion of collections, OAM introduces a supplementary catalog

structure, the object directory, to locate and track the hundreds of millions of objects expected for OAM applications. Each OBJECT storage group includes a separate object directory to reduce the size of the individual directories. This also allows physical separation of the databases where required for improved performance and to provide better recovery characteristics. An object directory in a storage group also allows automatic storage management to be readily handled at the storage group level.

Efficient selection of objects is the key to automatic object storage management.

Each object directory is an indexed table in the database that is part of each OBJECT storage group. The relational characteristics and multiple index capability of the database allow objects to be selected and updated in different ways for different types of storage management with the expectation of reasonable performance. Efficient selection of objects that require processing is the key to automatic object storage management. In addition, using a database provides serialization, backup, recovery, and other necessary facilities that would otherwise have to be provided by OAM.

OAM storage management construct interpretation. Each SMS Source Control Data Set contains the description of one storage management policy for a group of MVS systems. Activating a specific Source Control Data Set provides the Active Control Data Set to be used at a particular time. The OAM Storage Management Component uses SMS facilities to access the SMS construct definitions it needs for object storage management. It also interprets the constructs and runs the appropriate ACS routines, as needed, for specific OAM object management functions.

Storing objects. The OAM Storage Management Component locates the catalog entry for the collection specified in the request and extracts the default storage class and management class assignments from the entry. For a new collection, when storing the first object in the collection OSMC creates a catalog entry. If the request includes storage class and/or management class assignments to override the defaults, OSMC will invoke the automatic class selection routines to confirm, change, or reject the request. After the storage class and management class constructs have been assigned, OSMC decides where the object should be stored based on its construct definitions and the storage group identified by its catalog entry.

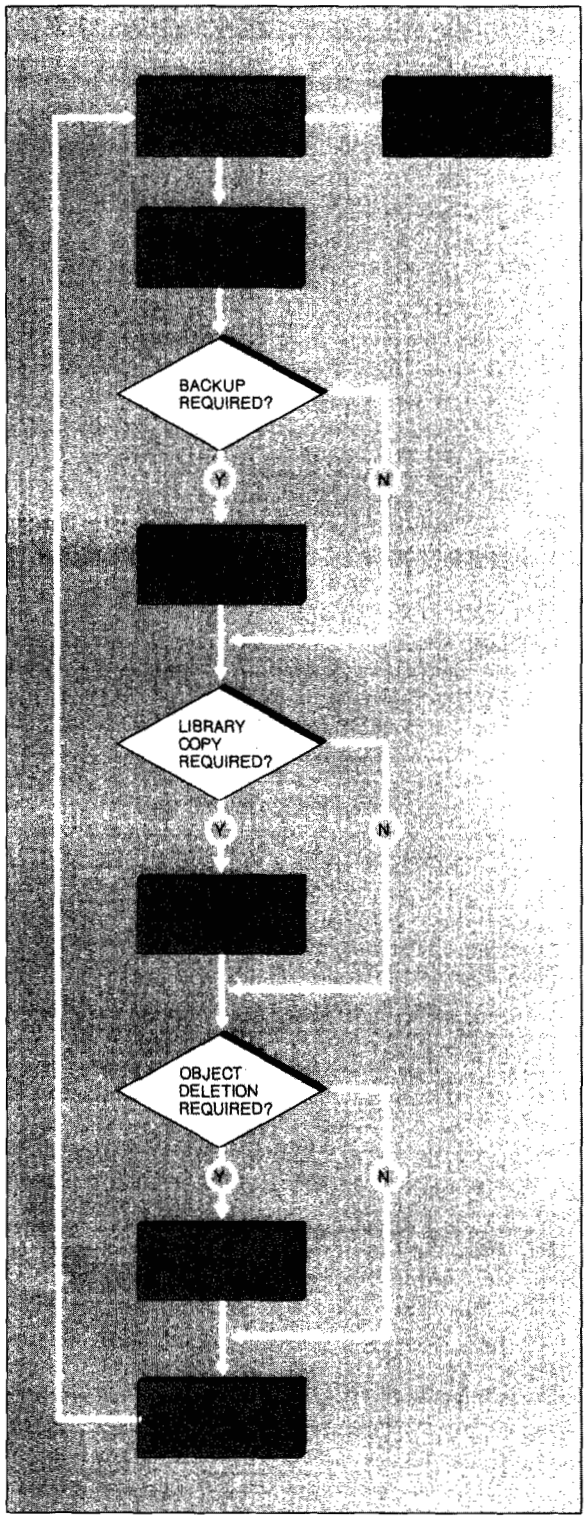
Retrieving and deleting objects. For object retrieval and object deletion requests, the OAM Storage Management Component simply locates the catalog entry, and through it the storage group, for the collection specified in the request.

Changing information about objects. For a request to change the storage class, management class, or expiration for an object, the OAM Storage Management Component locates the catalog entry for the collection specified in the request. If the request includes new storage class and/or management class assignments, the automatic class selection routines are run to confirm, change, or reject the request.

Selecting objects for automatic processing. The major function of OSMC is automatic storage management. To accomplish this, it must identify all objects requiring storage management processing for a particular storage group at the time processing occurs. A specific object may need to be moved within the object storage hierarchy (a storage class transition). A management strategy for an object could change because the object has not been referenced for a long time (a management class transition), or the object could have expired and may need to be deleted. In all cases, an individual object meets a requirement for automatic storage management defined in the storage management policy of the installation.

OSMC uses the definition of the management class assigned to an object to decide when the object needs to be managed. The expected date for the next automatic management for each object is set each time the object is processed. During automatic storage management this date, the pending action date, is set based on either the expiration or class transition criteria specified in the current management class—whichever occurs first. These criteria, in turn, may be based on when the object was stored (its creation

Figure 6 Processing flow for object services



date), the last time the object was referenced, a specific calendar date, or some periodic calendar function.

Objects are always scheduled for automatic storage management on the day they are stored, changed, or retrieved by changing their pending action date to the current date. When storing an object, object backup is deferred until the next storage management cycle, rather than being performed while the store request is being processed. This allows lengthy storage management activities, such as backup, to occur at a time when response time is less critical. At the same time, a new pending action date can be set using the current characteristics of the object. For example, a class transition may be postponed because the object was referenced if the transition is based on the number of days since the last reference to the object.

OAM Storage Management Component structure

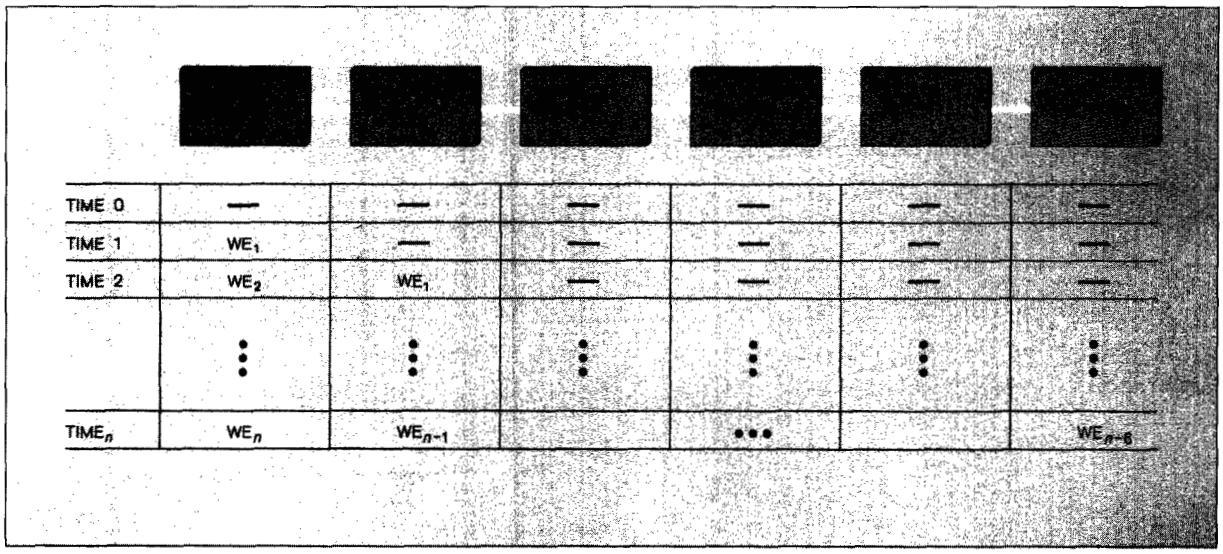
Automatic storage management can be considered as two basic functions: selecting objects that need processing and deciding what processing is required, and performing the services required to process each object. Objects are selected when their pending action date indicates they need processing. The type of processing required may be deleting the object, creating a backup copy of the object, changing its location in the object storage hierarchy, or setting a new pending action date. For example, an object that resides on DASD may be moved to the optical library, be backed up, and have its pending action date reset during a single storage management cycle.

Basic processing flow. The processing flow for the most common object service requirements (shown in Figure 6) has the following steps:

1. Select an object to be processed.
2. Read the object from DASD (if write required).
3. Write the backup copy of the object (if required).
4. Write the optical library copy of the object (if required).
5. Verify that the volume expiration date is later than the object expiration date.
6. Delete the DASD copy of the object (if required).
7. Update the object directory table.
8. Repeat steps 1 to 8 until all objects have been processed.

Initial performance concerns. Several performance concerns became apparent early in the design of the OAM Storage Management Component.

Figure 7 Synchronous



- Each of the steps in the process described above was suspect as a possible bottleneck in the process.
- Performance of object selection at step 1 and of object directory updates at step 7 was essentially unknown, and there would be contention between the two steps since they would operate on the same data in a close time sequence.
- Reading of objects from DASD at step 2 and deletion of DASD-resident objects at step 6 were both estimated to process between 5 to 10 objects per second, and there would be contention between the two steps.
- Writing data to the optical devices was estimated to have a maximum effective data rate of about 160 KBS per active path, pointing to the need for parallel operation of optical devices.

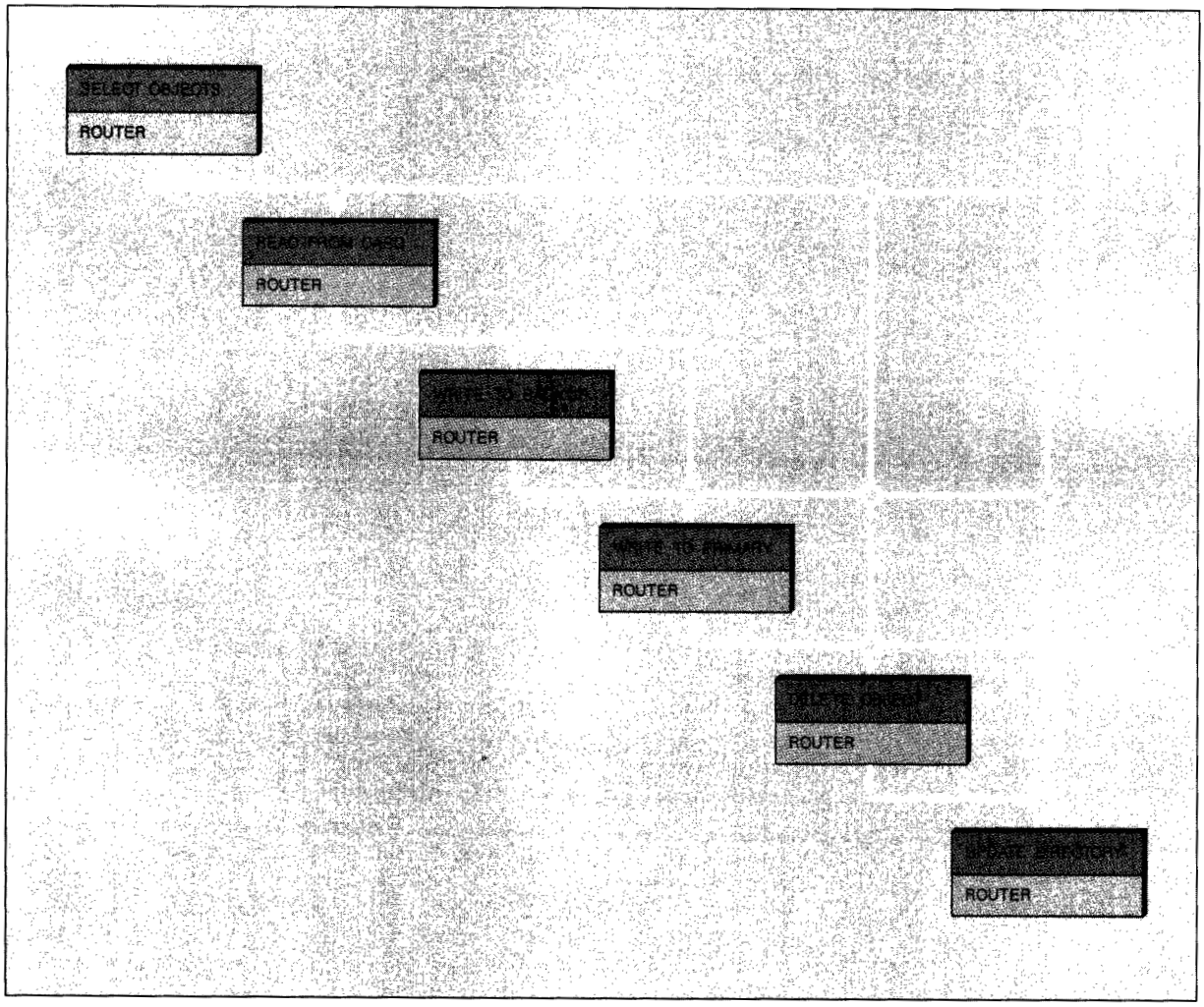
Early performance estimates concluded that if a serial process was implemented, the elapsed time for processing an object would be too slow, so multitasking alternatives were investigated. The alternative of attaching a task for processing of each object was soon rejected because of the perceived overhead of task creation, and the design effort was directed toward parallel, or pipeline, designs.

Pipeline designs. The use of pipeline designs is well known, and pipelines are frequently used in the design of high-speed processors. Several variations exist in the control of pipeline operations.

Synchronous. Figure 7 illustrates the basic concept of a synchronous pipeline. Initially, the pipeline is empty and no processing occurs. At each successive interval, a work element is introduced into the first process step (e.g., reading an object), and each work element in the pipeline is advanced to the next step. The advantage of this design is that a considerable amount of overlap is allowed between the process steps. The primary disadvantage is that the interval must be fixed at a value that is sufficiently large to allow completion of worst case processing for all steps. If the worst-case step completion time is much larger than the average step completion time, there is an unnecessary processing delay. This disadvantage resulted in the rejection of a synchronous pipeline design for OSMC.

Variable interval synchronous. This variation of pipeline design is the same as the previously described synchronous design except that the interval is variable. An interval is complete when all of the steps in the pipeline have completed processing. A simple counting of completion of processing by all the subtasks is sufficient to determine the completion of the interval. The advantage of this design is that any unnecessary delays in the interval to provide for worst case completion have been eliminated. Each interval may be of a different length, depending only on the processing time required by each subtask. The primary disadvantage of this design is that each work

Figure 8 Asynchronous



element must move through each process step, even if the step is not required for that work element.

Queued asynchronous. For an alternative pipeline design, the synchronization between subtasks is eliminated. In this design, each subtask has its own queue of work elements for objects to be processed. Each task executes at a rate determined entirely by the processing performed, and the availability of system resources to the task. Further, the sequence of steps can be varied so that an object work element is only routed through the specific steps required. This has the advantage of allowing each of the tasks to process only the objects that require its specific service, and to process asynchronously relative to the other tasks. The disadvantage is that the routing of work elements is no longer a simple stepping down the pipeline. An intelligent router is needed to determine the next processing step for a work element, and to place the work element on the correct work queue.

The OAM Storage Management Component is designed as a queued asynchronous pipeline, like the one shown in Figure 8, which provides the following characteristics:

- Processing steps or services operate asynchronously with respect to each other.

- Each step or service task operates from its own work queue.
- An intelligent router passes completed work from one service task to the appropriate work queue for the next service task required.
- The router may be called from each service task, effectively providing distributed intelligence.

Optical library interfaces

The Object Access Method can include an optical library unit and free-standing optical disk drives as parts of its object storage hierarchies and as object backup devices.[3]

Capacity and performance concerns. The device characteristics of the optical disk drives allow large variations in capacity and performance depending on the method chosen for writing the data. The data rate is relatively slow and the device uses a fixed block recording method in which data are always written in 1K sectors. Multiple optical volume table of contents (OVTOC) entries can be included in a sector. To make efficient use of optical storage for small objects of 4K in length, each OVTOC sector must contain the maximum number of object entries, otherwise the OVTOC will be filled before the data recording area, and the maximum recording capacity of the optical disk will not be used.

With one OVTOC entry recorded in a sector, each object actually requires 5K of optical space, 4K for the object and 1K for its OVTOC entry. Thus, 20 percent of the space is used for OVTOC entries. Because the OVTOC sectors are separated from the data sectors on the disk, the effects on performance are more dramatic. The recording of an OVTOC sector requires a seek, rotational delay, write, seek, and rotational delay sequence. The time required to write a OVTOC entry is roughly one-half second. If the OVTOC processing was not optimized, the time to write a 4K object and its OVTOC entry would be 16 ms for the data and 500 ms for the OVTOC, a total of about 516 ms. About 97 percent of the time would be consumed writing the OVTOC, and the resultant data rate would be about 7.7 KBS.

Chaining write requests. Because of the above concerns, the OAM Storage Management Component uses chained write requests to improve the data rate and space utilization for objects written during the storage management cycle. Multiple objects are chained in a write request, to provide a total length of data that can be written in about ten seconds, so that the time to write the OVTOC sectors amounts to less than 10 percent of the total write time. The number of objects in a chain is usually a multiple of the number of OVTOC entries in an OVTOC sector, so that no OVTOC space is wasted. This provides for complete utilization of the data-recording area of the optical disk.

Considering the 4K objects from the previous example, grouping the maximum number of OVTOC entries in a single sector reduces the space used for OVTOC entries from 20 percent to approximately 2 percent. Writing a chain of 360 objects would require about 9010 ms for data and 500 ms for the OVTOC. This produces a total of 9.51 seconds, of which only about 5.2 percent is for the OVTOC, and a data rate of about 151 KBS.

Volume expiration. When an object is entered into the system, it has an expected lifetime that is used to determine an expiration date for the data. As objects are written for storage to an optical disk volume, the expiration date of the volume is determined from the expiration date of the objects to assure that the volume does not expire until all the objects have expired. If there is a requirement to retain an object for a longer period, an explicit request is made to assign the new expiration date. The expiration date of the object is compared to the expiration date of the optical volume. If the expiration date of the object is later, the volume expiration date is changed to the expiration date of the object.

As time passes, objects and volumes eventually reach their expiration dates. When an object reaches its expiration date, it is deleted, and further reference to the object is no longer possible. When a volume reaches its expiration date, all objects must have expired or there has been an error in previous processing. A check is made to determine whether objects still reside on the volume. Normally, the result will be that no objects are found and the volume expires. However, when an object is found, the expiration date for the volume is adjusted to the latest object expiration date of all the objects still residing on the volume.

Summary

The Object Access Method was developed to provide convenient services for the development of application programs that process and store objects in a storage hierarchy. An object storage hierarchy utilizing DASD and an optical library provides a cost-

effective way to provide a large-capacity storage system that can satisfy a wide range of performance requirements. The very large storage requirement and the need to satisfy performance requirements that change as objects age within the system led to the development of the OAM Storage Management Component.

OSMC was designed to provide system management for objects. It uses existing system-managed storage concepts and extends those concepts to include objects. The storage group definition expands to include object hierarchies, the storage class to include mountable optical disk cartridges, and the management class to allow planned performance and management transitions. The large number of objects to be managed by OAM led to the design of a two-level catalog structure that uses the MVS catalog to map object collections to a storage group, and a relational object directory to locate objects within a storage group.

OSMC provides automatic storage management for objects. Objects are selected on the date they require processing by means of a pending-action date set during previous processing. When an object is selected, object services are performed according to the storage management policy of the installation.

In anticipation of the large volume of data to be processed during a storage management cycle, OSMC was designed as a queued asynchronous pipeline to reduce bottlenecks in the process, to adapt automatically to a varied workload, and to maximize the performance of the storage management process.

Acknowledgments

The authors wish to acknowledge the management guidance of Sharon Johnson. We would also like to thank W. O. Vomaska, L. K. Dibbern, and R. J. Lyle for their architectural insights, T. G. Thompson and T. W. Beglin, who led the development of OSR and LCS, respectively, and all of the others who contributed to the OAM development project.

Cited reference and notes

1. The object directory and other database structures discussed in this paper were implemented using IBM DATABASE 2™ (DB2™).
2. *MVS/DFP Version 3 Release 2: Managing Catalogs*, SC26-4555, IBM Corporation; available through IBM branch offices.
3. The optical library system and the stand-alone optical disk drive discussed in this paper are the IBM 9246 Optical Library Unit and the IBM 9247 Optical Disk Drive, respectively.

General references

J. P. Gelb, "System-Managed Storage," *IBM Systems Journal* **28,** No. 1, 77–103 (1989).

MVS/DFP Version 3 Release 2: Object Access Method Application Programmer's Reference, SC35-0119, IBM Corporation; available through IBM branch offices.

MVS/DFP Version 3 Release 2: Object Access Method Planning, Installation, and Storage Administration Guide, SC35-0120, IBM Corporation; available through IBM branch offices.

MVS/DFP Version 3 Release 2: Storage Administration Reference, SC26-4566, IBM Corporation; available through IBM branch offices.

MVS/ESA ImagePlus: General Information Manual, GC38-2024, IBM Corporation; available through IBM branch offices.

MVS/ESA Storage Management Library: Managing Data Sets and Objects, SC26-4657, IBM Corporation; available through IBM branch offices.

MVS/ESA Storage Management Library: Storage Management Subsystem Migration Planning Guide, SC26-4659, IBM Corporation; available through IBM branch offices.

Warren B. Harding *IBM General Products Division, 9000 South Rita Road, Tucson, Arizona 85744.* Mr. Harding joined IBM as a programmer in Endicott in 1956 and has been working in the Tucson Programming Center since 1982. He has been assigned to the development of the OAM Storage Management Component since 1987 and has received an IBM First Level Invention Achievement Award and an IBM Outstanding Technical Achievement Award in recognition of his contributions to the design and development of OAM.

Connie M. Clark *IBM General Products Division, 9000 South Rita Road, Tucson, Arizona 85744.* Ms. Clark joined IBM as a microprogrammer in Endicott in 1974 and has been working in the Tucson Programming Center since 1979. She was assigned to the development of the OAM Storage Management Component in 1987 and received an IBM First Level Invention Achievement Award in recognition of her contributions to the design and development of OAM.

Cindy L. Gallo *IBM General Products Division, 9000 South Rita Road, Tucson, Arizona 85744.* Mrs. Gallo joined IBM in Poughkeepsie, New York, in 1981 with a B.S. in computer science from California State College. She was a programmer in MVS Development and for the 3090 Processor Controller before transferring to Tucson in 1984. She joined the OAM Storage Management Component development in 1987 and is currently manager of the Object Access Method Test department.

Horace Tang *IBM General Products Division, 9000 South Rita Road, Tucson, Arizona 85744.* Mr. Tang joined IBM in 1986 and is a senior associate programmer working on the design and development of the OAM Storage Management Component. He received his B.B.A. in management information systems from the University of Hawaii in 1983, and his M.S. in management information systems from the University of Arizona in 1985.

Reprint Order No. G321-5407.