

TO: A. Bensoussan  
F. Corbató  
R. Feiertag  
J. Gintell  
N. Morris  
J. Saltzer  
D. Vinograd  
V. Voydock  
S. Webber

FROM: Mike Spier

DATE: July 14, 1971

SUBJECT: A preliminary Study of the proposed off-line segment  
storage capability for the Multics file system.

RECEIVED  
JUL 19 1971  
J. H. SALTZER

*Cent 5505*

This is a preliminary study of the proposed off-line segment storage capability for the Multics file system, which provides for the orderly movement of segments between on-line (random access) and off-line (dismountable) storage media.

The off-line storage medium which is currently most applicable for Multics is magnetic tape; other media, however, such as disk packs or even punched tape or cards are equally conceivable. The implementation is therefore expected to be sufficiently flexible to allow for such storage media, if and when required.

It has been argued that the off-line storage capability should be restricted to terminal data (non-directory) branches of the file system hierarchy only. There are strong arguments in favour of a general capability to move both terminal branches and terminal sub-trees of the hierarchy to an off-line storage medium. One is the Multics files, currently being developed by André Bensoussan; these files<sup>which</sup> are implemented in the form of sub-trees are the intended vehicle for the storage of high-volume data, and as such are prime candidates for off-line storage. Also, it seems strongly advisable for both functional and economical reasons to integrate the basic mechanisms of both the off-line segment storage facility and the incremental backup facility, in which case the ability to output sub-trees becomes implicit.

For the sake of clarity, we shall henceforth use the term "off-line segment" to refer to both segments and sub-trees.

### Output and Retrieval

The directory branch of a segment which had been moved off-line remains in on-line storage, allowing the system direct access to the segment's attributes, and giving the user the impression that the segment is directly available to him (albeit after some delay in time). The user may display the branch (e.g., using commands such as 'list', 'status' etc.), modify it (e.g., 'rename', 'set\_acl' etc.) or even delete it from his directory without ever having to resort to the physical reading of the off-line storage medium.

Commands will be provided to allow a user to effect the bilateral movement of a segment to and from off-line storage. An analogous command will be provided

to allow a user to request (from his console) the retrieval of the last backed-up version of some segment; the output request to backup a given copy of a segment is, of course, issued by the system automatically. Following the implementation of the dynamic page migration facility, dynamic segment migration to off-line storage might be considered, in which case the output request would be issued by the system, automatically, for selected segments.

### Suggested Implementation

The segment, together with all or some of its branch's information (e.g., the unique identifier) are output onto the off-line storage medium, and the segment proper is removed from the on-line storage (i.e., the pages which it occupied are freed). The device address in the segment's branch now carries the unique identifier of the new storage medium (e.g., magnetic tape reel number 3712) and perhaps (as in the case of disk packs) an address within that medium. In order to accelerate the reading of magnetic tapes, a scheme may be worked out by which a tape reel is subdivided into section (by means of some tape mark character) which may be skipped over at high speed, and the segment's 'address' on that reel would correspond to the section within which it resides.

A single device (e.g., tape drive) is permanently reserved for the writing of segments which are moved off-line, and all such segments are output sequentially onto the storage medium, so that the output function (i.e., liberation of on-line storage space) will be as effective as possible and will require a minimum of manual operator's intervention. A request for retrieval will display an appropriate message to the operator who will then locate the specified storage medium, mount it on an available device and let the system search it until a segment, featuring a unique identifier which is identical to the one in the segment's branch, is located.

The integration of the off-line storage facility with the backup facility may be trivially accomplished by storing in the segment's branch two additional items which are a) an off-line storage medium address, and b) a unique identifier (which, naturally, will have to be derived from the segment's unique identifier) of the current backed-up copy of the segment. This would allow for the automatic retrieval, by means of a user command, of the historically last backed-up copy of a segment. The retrieval will cause the updating of the "last backed up copy address" in the branch, thus providing a continuous sequential historical backup capability.

An on-line reference to an off-line segment will result in an 'off-line segment' fault. After considering the pros and cons of the automatic retrieval of an off-line segment as the result of such a fault, it seems advisable to effect only the signalling of a 'reference to an off-line segment' condition to be handled by the default error handler in its customary way, leaving it up to the user to decide whether or not to issue an explicit retrieval command.

### Formats

In order to maintain compatibility between the format of a branch as stored on the off-line storage and that of a branch in the current on-line hierarchy (and which is subject to change), branch information will be written onto off-line storage in a symbolic, self-designating, format. The current input/output handlers for the off-line storage facility will always be cognizant of the current structure of an on-line directory branch, and will perform the appropriate translation.

### Synchronization

The mechanism which moves segments from on-line to off-line storage is logically similar to the paging mechanism which moves pages in and out of core memory. A (probably substantial) delay in time is involved between the issuing of a request to move a segment off-line, and the actual accomplishment of that request. Assuming that such requests will be handled by a dedicated daemon process, the delay may be in the order of minutes or even hours. It is therefore imperative to associate a 'segment out of order' flag with each directory branch (analogous to the 'page out of order' flag) to indicate that a given segment is logically off-line, even though it has not yet been moved physically. A synchronizing algorithm should be worked out so that a retrieval request for an out of order segment would cause the deletion of the queued-up device write request and the resetting of the out of order flag, to avoid wasteful data movement between physical devices.

This raises the interesting question as to whether or not the on-line storage space occupied by an out of order segment should be credited to the user's storage quota. It has been observed with the printer daemon that during system peak load periods (or sometimes for reasons of device malfunction) requests may tend to queue up considerably. It may be argued, on one hand, that it would be improper to penalize a user who wishes to immediately relieve his congested storage space in order to proceed with some useful task; on the other hand, however, the possibility exists that a certain type of sophisticated user might abuse such an implementation in order to temporarily expand his insufficient quota. Clearly, this problem must be resolved by the application of an appropriate policy decision, the algorithm of which is however not self-evident, due to the unavailability of concrete parameters. It is therefore strongly recommended that the implementation be flexible enough to allow for experimentation and appropriate tuning.

Analogously, a 'read request issued' flag must be put into the branch in order to guard against redundant retrieval requests.


Tape Reel Management

The management (i.e., preservation and re-usage) of tape reels used for the storage of off-line segments presents a certain difficulty.

Whenever an off-line segment is restored to on-line storage (or deleted by use of the 'delete' command), its physical copy on tape cannot be deleted. Segments are output onto tape sequentially, on a first-come first-served basis, so that a single reel of tape contains a random mixture of segments, some of which may be retrieved within a short while, while others may remain off-line for a considerable period of time, or perhaps indefinitely. Given the impossibility to modify the tape copy of a segment in order to indicate its deletion, it is unclear how to determine whether or not a given reel contains any currently valid segment. Furthermore, given the fact that tape reels cannot be stored indefinitely but have to be recirculated periodically (primarily for economical reasons), some mechanism has to be devised to control their re-usage.

No  
 System should have option leaving it on-line indefinitely.  
 Read issue is price. can work cheap storage.

In the following, three progressively complex solutions are suggested:

 a) For the initial implementation, the solution which comes most readily to mind is that of setting an arbitrary time limit (say, one year) on off-line segments, after the expiration of which they are automatically destroyed, and the tape reel is released for re-usage. An attempt to retrieve such a destroyed segment will result in the signalling of an appropriate error condition, determined by the date of storage as found in the segment's branch (or perhaps such branches may be detected and deleted by the incremental backup system as it scans the entire hierarchy).

b) In the long run, the previous solution is unacceptable because Multics must have the ability to guarantee the storage of segments for as long as the user desires (and pays for). Also, it is quite conceivable that tape reels containing often-used segments may be ready for re-usage long before the above time limit has elapsed (i.e., all, or most, of the segments on a reel have been retrieved or deleted and the reel is ready for re-usage) so that it becomes economically wasteful to needlessly keep that reel for the entire storage period.

The proposed solution is to keep in on-line storage a tape reel table, which has entries for all tape reels used for off-line segment storage. Every entry contains a list of the pathnames of all currently valid segments stored on that reel. A retrieval (or deletion by means of the 'delete' command) of an off-line segment causes the appropriate table entry to be updated. Whenever the number of valid segments on a reel falls below a given threshold, a message is sent to the operator to mound that reel, and a special system procedure retrieves those valid segments, then immediately turns around and outputs them onto the tape reel currently used for off-line segment storage. In other words, those few segments which stop us from re-using an older tape reel are simply copied onto the current reel, releasing the older reel for re-usage. An additional advantage of this scheme is that users may freely copy or move off-line segments (i.e., their branches) around, without impairing Multics' ability to guarantee the indefinite storage of off-line segments.

c) The only apparent drawback of this scheme is the considerable on-line storage space necessary to maintain the tape reel table. The third suggested

solution is to make that table into an off-line segment itself, and maintain on-line only a small update table reflecting the changes which are currently taking place. Periodically, the tape reel table is retrieved, updated, the appropriate steps are taken to release tape reels which no longer contain valid segments, whereupon the tape reel<sup>table</sup> is moved off-line again. This operation may (for example) be run nightly by some dedicated daemon process.

Accounting

Accounting management for off-line segments seems to present no special problems, due to the fact that all pertinent information may be looked up in the segment's branch, which remains on-line.

↑  
but if it is a directory?

concerns  
very old files  
through system to  
look for segments in  
current hierarchy

Access Control

who can cover push down + retrieval?  
only the owner?  
Any one w/ access?