To: Distribution

From: A. Bensoussan

Date: December 3, 1975

Subject: Overview of the New Storage System

INTRODUCTION

The goals that are expected to be reached by reorganizing the storage system have been discussed earlier in the MTB-110. They are summarized below for the convenience of the reader.

- Reduce the amount of lost information
- Reduce the system resources spent by the backup facility
- Provide for the handling of a large number of disks
- Provide additional features such as the ability to group a set of segments in the same volume and the ability to handle removable disk packs
- Improve the operator interface

We expect to reduce the amount of lost information by organizing each disk pack so that it contains a description of what is stored in it. In the old system, a disk pack contained a collection of anonymous pages. In the new system, a portion of each pack is reserved, at a conventional location, to keep its table of contents. Now, given a pack that stands by itself, one can determine which segments reside on it, where on the disk the pages of each segment reside, and where in the tree structure the segment resides. Losing the directory entry for a given segment no longer implies losing the content of the segment.

We also expect to reduce the amount of resources spent by the backup facility. In the old system, it was necessary to walk through the directory tree in order to find out which segments had to be dumped, and it was necessary to activate all parents of a given segment in order to dump it. Now the backup process is driven by a list of segments to be dumped. Each element of this list specifies where the segment physically resides. Backup becomes independent of the directory hierarchy.

The maximum number of disk packs that could be on line in the old system was limited by 2 factors: (a) the size of the address field for a disk address; and (b) the size of the allocation map (known as the FSDCT) for the entire disk storage of the system, a map that was required to be core resident. If the number of volumes were to increase, to say 100, the address field for a disk address would overflow and the FSDCT would become too large to always be kept in core. These two limitations are removed in the new storage system.

Also in the new storage system, the physical grouping of a set of segments is made possible by a new disk allocation strategy. All pages of a given segment are allocated in the same volume and a way of defining the set of segments to be allocated in a given volume is available to the user.

A removable disk facility is provided, with disk registration procedures to control who can mount and demount a given volume. One important property of this facility is that, after a disk has been mounted, its segments become part of the virtual memory.

The remainder of this document describes the new concepts that have been introduced and the design decisions that have been made in order to integrate these concepts in the Multics system. The following topics are discussed:

- Physical volume organization
- Logical volumes
- Segment grouping conventions
- Logical volume assignment
- Physical volume assignment
- Quota
- Removable volumes
- Backup
- Salvager

PHYSICAL VOLUME ORGANIZATION

The significant points of the new physical volume organization are the following:

1. Page allocation strategy

   All pages of the same segment are allocated in the same physical volume. This new strategy is required for the implementation of the removable volumes.

2. Volume Map

   On each physical volume an array is kept with one entry for each physical 1024-word record of the volume. This array is called the Volume Map and each of its entries is called a Volume Map Entry. Each volume map entry consists of a single bit defining whether the page it represents is free or has been allocated. The idea of keeping more information in each volume map entry, such as the unique identifier of the segment to which the page has been allocated, and the page number within the segment, was first considered but was rejected because of the large number of additional disk transfers it would have required.

3. Volume Table of Contents

   On each physical volume, an array is kept with one entry for each segment stored in the volume. This array is called the Volume Table of Contents (VTOC). The purpose of the VTOC is to provide, on the disk itself, the list of all segments stored in it, with their physical attributes. Each VTOC entry contains the unique identifier (uid) of the segment, the file map, the maximum length, the current length and the number of records used. It also contains what one might call the "operational" attributes of the segment, which are systematically updated by the supervisor each time a segment is activated or deactivated, such as the date and time used, date and time modified, and the quota information. In addition, it contains some information that is used by the salvager to locate the position of the segment in the directory structure, such as the uid's of all the parents, the primary name, the time the segment was created, etc. (see Fig. 1).

   The access class used for security purposes is kept in the branch and in the VTOC entry.

   Since the segment attributes are now partly in the branch and partly in the VTOC entry, the branch contains a pointer to the VTOC entry. This pointer consists of the physical volume identifier and the VTOC entry index.

Two important advantages result from this VTOC organization. First, it becomes possible to perform some form of physical dump of a disk and use this dump later for salvaging, reloading, or retrieval purposes. Second, the number of I/O transfers for directory pages is significantly reduced since activation and deactivation are now performed without modifying the parent directory. In fact, a directory modification can only be caused by an explicit request such as create segment, delete segment, change ACL etc.

4. Physical volume registration

Each physical volume is registered, the same way each tape is registered. The registration of a physical volume causes the creation of a "registration record" in a registration file or data base; the registration record is addressable by a unique name or unique id associated with the physical volume, and is protected by some form of access control list (see Fig. 2).

LOGICAL VOLUMES

We provide a facility by which the user can specify a group of segments and cause all segments of the group to be allocated in a given volume. The method that has been chosen to define groups of segments is not relevant for the moment and will be described in the next paragraph. The only point of interest is that any segment is part of one and only one group.

A user should be able to request that all segments of a given group G1, and only they, be stored in a given physical volume PV1.

A user should also be able to request that all segments of the groups G1, G2,...Gn, and only they, be stored in the physical volume PV1.

If the number of segments in the groups G1, G2,...Gn increases to the point that the physical volume PV1 becomes full, we would like to be able to extend PV1 with another physical volume, say PV2, and use it as the continuation of PV1 to store those segments of G1, G2,...Gn that did not fit in PV1. The physical volumes PV1 and PV2 are said to be part of the same logical volume.

A logical volume consists of a set of n physical volumes, with n>=1. It is identified by a unique name and a unique id and is the secondary storage entity to which a group of segments may be assigned.

A user should be able to request that all segments of the groups G1, G2,...Gn, and only they, be stored in the logical volume LV1,

which consists of the physical volumes PV1, PV2,...PVm.

Note that two segments of the group G1 are not necessarily in the same physical volume; one may reside on PV1 while the other may reside on PV2.

Logical volumes are registered. Their registration causes the creation of a "logical volume registration record" in a logical volume file or data base; this registration record is addressable by the name or uid of the logical volume, and is protected by some form of access control list. It contains the following information (see Fig. 3):

- The logical volume identifier (name and/or uid)
- The list of physical volume identifiers (name and/or uid) defining all physical volumes that are part of the logical volume.
- The list of segment group identifiers (name and/or uid) defining all segment groups that may and must reside on the logical volume.

A physical volume retains its characteristics, as described above, regardless of the logical volume considerations. That is:

- All pages of a given segment are allocated in the same physical volume
- A physical volume always has its own volume map
- A physical volume always has its own VTOC
- A physical volume always has a physical volume registration record.

SEGMENT GROUPING CONVENTIONS

The most general way of specifying a group of segments would be by an explicit list of all segments of the group, regardless of their position in the tree structure. For practical reasons that are explained below, two conventions have been made that establish some relationships between the directory structure and the way a group of segments is defined.

Convention 1: All directories are part of the same group and therefore must reside on the same logical volume.

Convention 2: All immediately inferior non-directory segments of a given directory are part of the same group and therefore must reside on the same logical volume.

The first convention was made for two reasons: (a) It makes it possible to guarantee that all directories are always on line. It does not seem desirable to allow directories to be stored in removable volumes because, when a directory is not on line, all segments of the subtree below it cannot be accessed even though they might be on line. Also, if directories were demounted, it is not clear how their content could be completely validated when remounted. (b) It greatly simplifies the mechanisms to maintain a duplicate copy of the entire directory hierarchy, a strategy that may be used as a complete or partial substitute for incremental backup for directories.

The second convention was made primarily because of quota and accounting considerations. With this convention, quota and accounting policies, which are merely extensions of the old policies rather than different ones, can be specified. In addition, the mechanisms that implement these policies are very similar to the old ones.

If we define the term "son" of a directory as being an "immediately inferior non-directory segment" of this directory, convention 2 can be stated as follows: All sons of a directory are part of the same group and must reside in the same logical volume. One can also say that a directory has one and only one sons' logical volume, and that a logical volume can be the sons' logical volume for any number of directories without any consideration of their position in the tree structure.

With these conventions, the list G1, G2,...,Gn of segment groups associated with the logical volume can be expressed by a list of directories, DIR1, DIR2, ..., DIRn, in the registration record of a logical volume (see figure 4).

LOGICAL VOLUME ASSIGNMENT

According to convention 1 all directories reside in the same logical volume. This logical volume, the root logical volume, is specified at system initialization.

According to convention 2, all sons of a given directory reside in the logical volume specified as being the sons' logical volume for this directory. The sons' logical volume for a given directory is specified by the user when he creates the directory. A default value is provided by the system if the user fails to provide it. Two additional conventions have been made to govern the assignment of a logical volume to a directory for its sons:

Convention 3: A directory D is allowed to use a given logical volume L for its sons if its parent already uses it for its own sons, or if the name of the directory D appears in the registration record of logical volume L.

Convention 4: The default value for the sons' logical volume of a directory is that of its parent.

A directory whose sons' logical volume L is different from that of its parent is referred to as a "master" directory for the logical volume L.

A given logical volume may have several master directories, whose position in the tree structure may be arbitrary. All master directories for a given logical volume must have their names listed in the registration record of that logical volume (see Fig. 4).

The sons' logical volume id for a directory is kept in the branch of that directory. For convenience, it is also stored in the header of the directory.

PHYSICAL VOLUME ASSIGNMENT

Whenever a directory or non-directory segment is created, the system must select the physical volume on which the new segment will reside. It first determines the logical volume for the created segment, as explained above. From all the physical volumes that are members of this particular logical volume, the system chooses the physical volume that has the most free space. The physical volume id is then entered in the branch of the created segment. A VTOC entry is assigned on the physical volume for this segment, and initialized. The VTOC entry index is then entered in the branch.

A new situation may arise that has no equivalent in the old system. Suppose a segment residing on a given physical volume needs a new record but there is no free record on that physical volume. When this situation occurs, the entire segment, including its VTOC entry, is moved to another physical volume, which is a member of the same logical volume. If no physical volume in the logical volume has room for the segment, the "logical volume overflow" condition is signalled.

QUOTA

The quota facility of the new storage system is derived from that of the old system. A few extensions, however have been made in order to integrate the logical volume concept.

The old quota facility may be described in terms of the following statements:

1. The entity with which a quota is associated is a directory, rather than a user or a project or any other entity. A directory that has been associated with a quota is referred to as a "quota directory".

2. The quota associated with a directory defines the maximum number of disk records that can be used for all segments of the subtree that is rooted at that directory and does not include any other quota directory.

3. When a new disk record needs to be allocated to a segment, the record is to be "charged" to the nearest superior directory that happens to be a quota directory. The record is effectively allocated and charged only if the quota of that directory was not already exhausted.

4. When a disk record allocated to a segment is freed, the record is "discharged" from the nearest superior directory that happens to be a quota directory.

5. A quota directory D can "delegate" a portion x of its quota to any of its immediately inferior directories D>d. The quota of D is decremented by x while the quota of D>d is incremented by x. In addition, any record of any segment of the subtree rooted at D>d that has been charged to D is discharged from D and charged to D>d. A directory to which some quota has been delegated is a quota directory and may, in turn, delegate some of its quota.

6. A quota directory D>d may return to its parent D, from which it obtained its quota, any portion x of its unused quota. The quota of D>d is decremented by x while the quota of D is incremented by x.

   It may also return the totality of the quota it received from D, provided that it has not delegated any portion of it. In this case, the quota of D and D>d are adjusted as above but, in addition, all records that are charged to D>d are discharged from it and charged to D, and D>d becomes a non-quota directory.

7. In order to delegate quota of a directory or to return quota to a directory, the user who issues the request must have modify permission on the directory. For "security" reasons, one cannot return quota from directory D>d to directory D if the "access class" of D is inferior to the access class of D>d.

The new quota facility can be described in terms of the above statements, with the following additions:

8. The quota for directory pages and the quota for non-directory pages are managed independently. Each directory is associated with two different sets of quota information, one for the directory pages, the other for non-directory pages. The quota information attached to each directory could basically be represented by the following declaration:

```
dcl 1 quota_info (0:1)
    2 quota_directory_switch
      quota
    2 number_of_records_charged_to_this_quota
    2 quota_delegated_to_inferior_directories
```

   It is possible for a directory to be a quota directory with respect to non-directory pages and a non-quota directory for directory pages and vice versa.

   Once it has been established if the quota of interest is for directory pages or for non-directory pages, the management of the quota is governed by the same rules as in the old system.

9. A master directory cannot receive from or return to its parent any quota for non-directory pages. A master directory is always a quota directory for non-directory pages. It obtains its quota for non-directory pages from the value specified in the registration record of the logical volume for which it is a master directory. This value can be specified only by the owner(s) of the logical volume.

REMOVABLE VOLUMES

Some physical volumes are required to always be mounted while the system is in operation. A given number of disk drives is reserved for these volumes. The other disk drives are eligible to be multiplexed among all other physical volumes, through the mount and demount facility.

The following conventions have been made regarding removable volumes:

1. When a physical volume is on line its content is part of the Multics virtual memory.

2. All physical volumes of a given logical volume must be on line or off line at the same time.

3. All directories reside in the same logical volume, called the root logical volume, and are always on line.

When a user issues a mount or demount request he specifies the logical volume for which he is requesting the operation. The system validates the request by examination of the registration record of the specified logical volume. Then it determines, from the logical volume registration record, which physical volumes are members of the logical volume. A logical volume can be mounted only when there are enough disk drives available to accommodate all physical volumes included in the logical volume. A logical volume can be demounted only after all of its physical volumes have been updated with the current information residing in core, in the paging device and in the FSDCT. The removable volume facility is described in detail in MTB-229.

BACKUP

The backup process is driven by the list of segments that have been modified since the last time they were dumped. Each element of this list conceptually consists of the physical volume id and the VTOC index of the segment. The backup process(es) acquires one element of the list, activates the corresponding segment (if it is not already active) without having to activate its parents,

dumps the segment, and records the current time in its VTOC entry. It keeps doing this until the list is empty.

Several backup processes may run concurrently without any interference, provided that each element be removed from the list in an indivisible operation when a process acquires it.

Actually, this list is implemented by several lists, one for each physical volume. The threads are located in the VTOC entries. While a segment is active, its AST entry is periodically examined. If the segment has been modified since the previous examination, its VTOC entry is entered in the backup list (if it is not already in). The periodic examination of the AST entries is a function that is already required in order to periodically update the VTOC entry of a segment during the time it remains active; it does not introduce any additional mechanism or additional overhead. A detailed description of the backup facility is given in MTB-233.

SALVAGER

In the new storage system the salvager is broken up into three parts: The physical volume salvager, which guarantees the consistency of physical volumes without any knowledge of directories; the directory tree salvager, which guarantees the consistency of the directory hierarchy, without any knowledge of the physical volume structure; and the tree-vtoc salvager, which guarantees that each branch is associated with a VTOC entry and each VTOC entry is associated with a branch.

1. The physical volume salvager.

   The role of the physical volume salvager is to verify that the volume map and the array of VTOC entries are consistent and to report and correct any inconsistency it encounters. A volume that has been processed by this salvager has the following properties: all reused address conflicts have been resolved, all file maps contain only valid (or null) disk addresses, the bit map (showing what records are allocated) agrees with all file maps, for all VTOC entries the current segment length, the maximum segment length and the number of records used are consistent with the file map and, finally, all free VTOC entries are threaded together in a free list.

2. The directory tree salvager.

   The directory tree salvager does what the salvager used to do in the old storage system except that it does not concern itself any longer with disk addresses, file maps, reused addresses, and all items that the physical volume salvager is now responsible for.

3. The tree-vtoc salvager.

   This salvager verifies that every branch of any directory is connected to a VTOC entry and also that every VTOC entry is connected to a branch. It may be invoked during the physical volume salvaging or during the directory tree salvaging.

The salvager is described in more detail in MTB-220 and MTB-221.

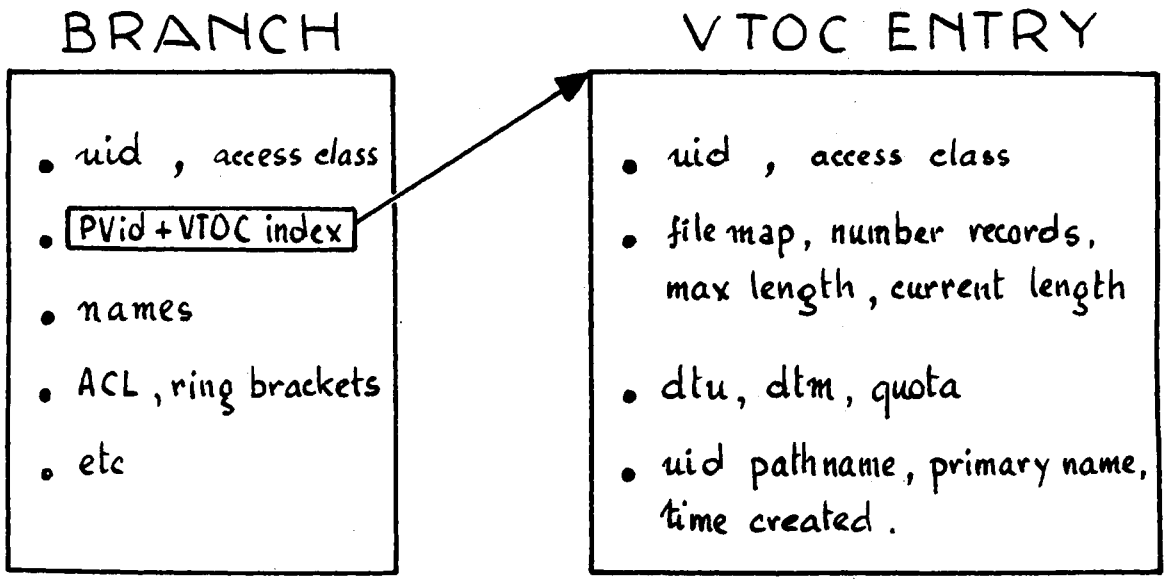

Figure 1. Segment attributes in the branch and in the VTOC entry.

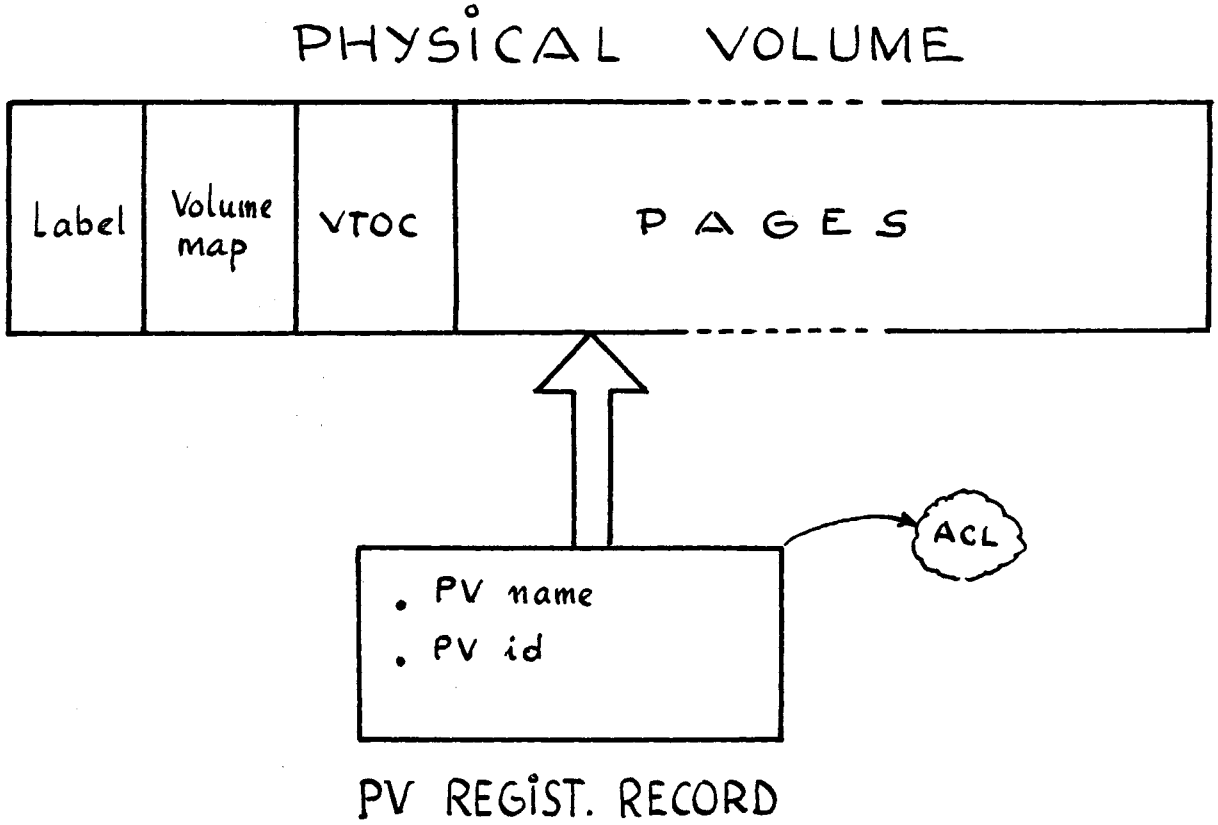

Figure 2. Physical volume and its registration record.

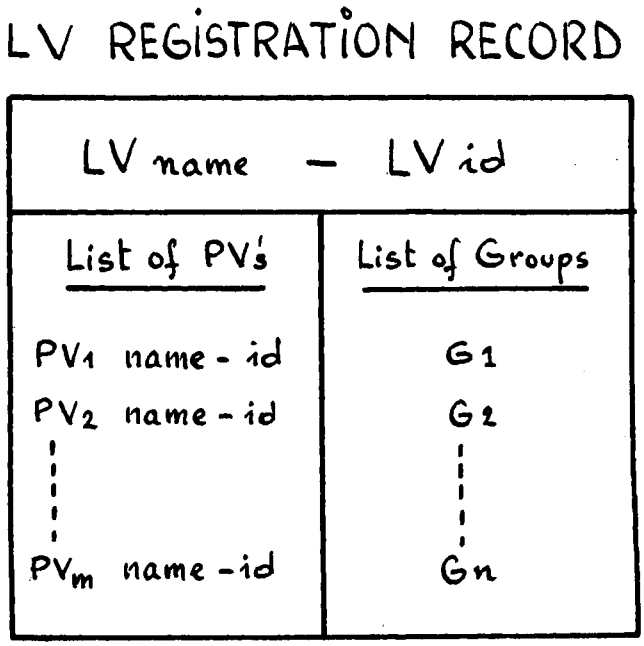

Figure 3. Logical volume registration record.

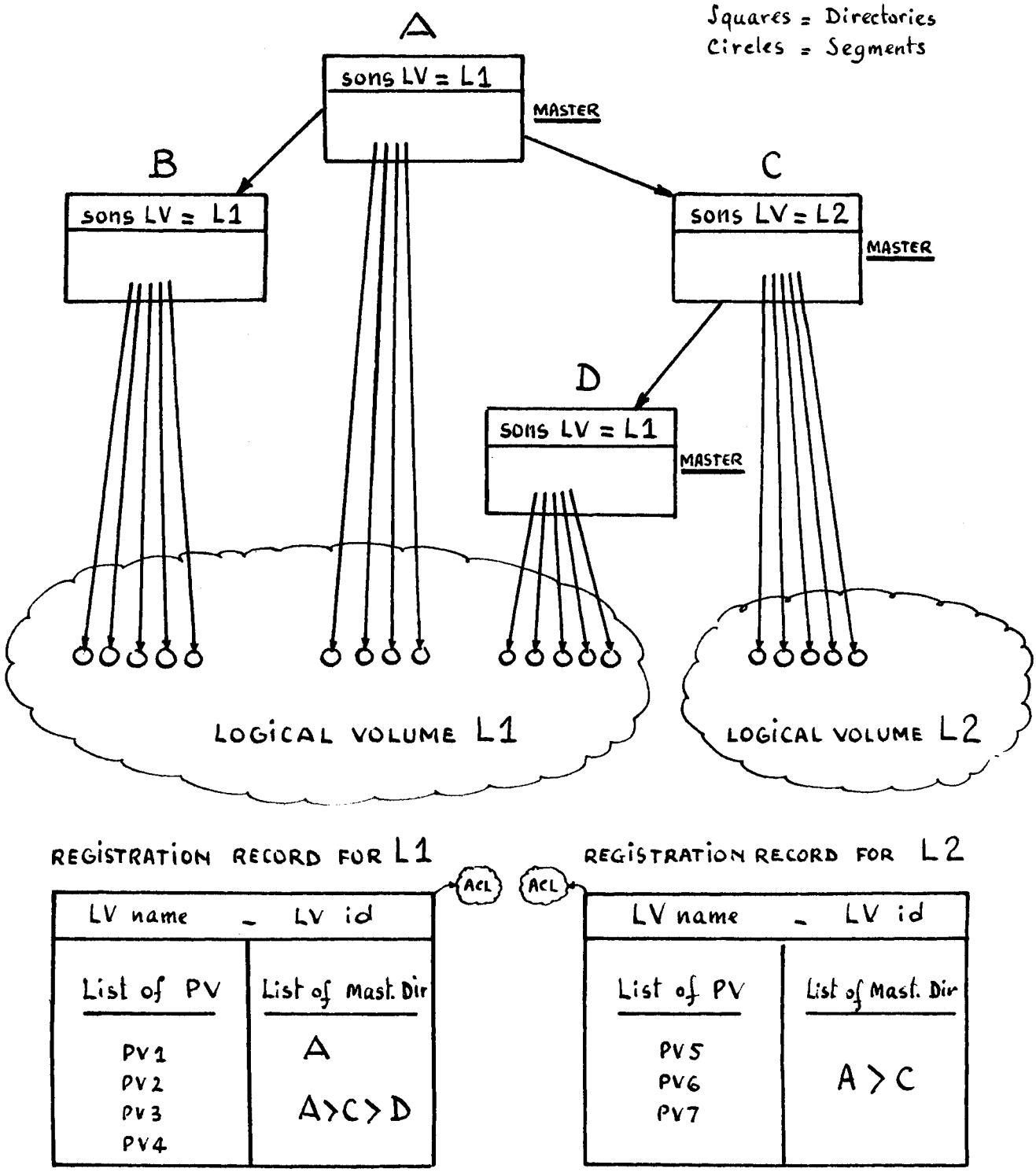

Figure 4. Segment grouping, logical volumes and master directories.