TEN-SYS-1

To: PDP-1Ø System Design File

From: DLM, TRS, RST, JDB, ERF

Subj: Tentative Plan for Ten-Sys File Structure

Date: Feb. 14, 1969

- - - - - - - - - - - - - - -

Note: Believing that the time has come to select a convenient handle by which to refer to out forthcoming PDP-1Ø ultimate OLRTTS* system, I have coined the name Ten-Sys as above. I would have included BBN in the name somehow, but it is unpronouncable. Any other suggestions are welcome.

* * * * * * * *

Names vs. Master File Numbers

A file in Ten-Sys will often contain pages which are to be shared among several files. We have decided that such pages should be referenced by actual file name and page

- - - - - - - - - - - - - -

* - On Line Real Time Time Sharing

number rather than by any scheme of file numbers, however obtained. The most frequest case of this will be when an environment or part thereof is saved as a file. The monitor will translate all shared pages in such an environment to the original file name, including directory name if necessary, and write that as additional data on the output file. This will require that such files be slightly longer, but will prevent problems in generating unique numbers.

Names

File names should include optional version numbers. Reason: when a new version of a shared subsystem is put into use, dump files made from the old version should remain usable. Additional thought: the EXEC will need to be told explicitly the version number of a subsystem when a new version is put up.

File Directories

There should be one directory for all files for all users. This will be a regular file which will always be open. It will be a table which is indexed into with a hash on the user number and file name. For shared files, an FD entry will contain the user number and file name of the indirect file. When such a indirect pointer is setup, a share count in the owning FD entry will be incremented. If the owner deletes the file, and the share count is not zero, the file will become invisible to the owner, but will remain accessable to users having the share pointers. When the share count is reduced to zero as share pointers are deleted, the backup system will delete the file.

When an FD listing is made, or at any other time when it is necessary to examine all the FD entries for a particular user, all blocks of the FD which may contain entries for that user must be referenced. Hence it seems desirable to find some scheme for clustering files of each user within the FD. What to do about users with many files is the problem.

Multiple FD's for Users

It seems desirable to provide users with a means of having access to several file directories (all within the one big FD, of course). These may be directories shared by users working on a common project, or just a means of separating files into useful groups.

To do this, we propose the following scheme. File directories will be identified by a number. There will thus be at least one file directory for each user corresponding to the user number. Users may attach to other users file directories if access words so permit. Users may also define new FD's with an EXEC command which simply creates a new user with an indication that this user may not "enter". Users should be able to enter only under their own names. Once entered however, thay may attach to any permitted file directory. File directories may have passwords. There will thus be two numbers maintained in the job TS block, the real user number, and the FD number.

Short and Long Files Question

It seems desirable that a way be found to allow small files (less than 8 pages or so) to exist without requiring an entire page for the page table. The question: where to keep the pointers if not in a real page table. Possibilities:

1. In the FD. (Bad because FD must be changed if file is lengthened or shortened.)

2. Beginning of first data block of file. (Bad because indexed random references to file would be offset.)

3. After last word of data in last block of file. (Must be protected against user modification, but

this seems like small problem. Also loses if data in file fills up to end of block requiring pointers to cross page or be put in separate page anyhow.)

Files longer than 256K will be needed. Page tables (index blocks) will be chained, last word (with "long file" indicator bit) will point to next page table.

File Handles

File numbers for open files are needed, ala 940. File numbers (and file indexes if we have them) must indirect through job TS block so that environment can be saved and restored while maintaining status of open files. File indexes probably aren't needed. JSYS's can use string pointer to name just as well.

Files should be referencable sequentially with byte-IO or multiple byte-IO instructions. These will simply use windows in the monitor map as "buffers". Data needed for open file: Job number, byte pointer, page table pointer, delay spec, some in resident monitor, some in job TS block.

TS Blocks

There must be a TS block for each job as well as one for each process of a job. It will contain job global information, e.g. login time, open file data, etc.

Questions Still Under Discussion

User directory: 1. Part of FD or separate; 2. User data separate from name/number table.

Sharing and protection - protection strategy number.

Formats