EXTENDED MODEL ANSWER

Operating Systems 2003 Paper 1 Question 11 (SMH)

Hierarchical name space

Unix supports a hierarchical name by using a tree of directories with a designated root. Each directory contains a list of names and the numbers of their associated inodes; these inodes in turn refer to either further directories, or to files (leaves of the tree).

Storage allocation

The Unix file system (the version covered in lectures) manages storage by keeping a *free list* of blocks which have not been allocated. When a new block is needed, the head of this list is chosen; when blocks become free (e.g. after a deletion), they are returned to the list. The head pointer is kept in the superblock, which lives at the start of the partition.

Filesystem metadata

A diagram here should show the hierarchical use of indirect blocks, etc.

Unix holds filesystem metadata in the superblock, and file (and directory) meta data in inodes. The superblock holds information such as logical block size, size of file system, mount status, number of inodes, start of free list, etc.

The per-file [and directory] inodes hold information about the owner of the file, the time it was created (and modified, and accessed), the size of the file in bytes, the access permissions, and the location of the file data blocks on disk.

The addresses of the first n (e.g. 10) disk blocks are held directly in the inode, followed by the location of a single indirect block, followed by the location of a double indirect block, followed by the location of a triple indirect block.

Indirect blocks represent a trade-off between the speed of block location and the size of an mode. Since many files are small, having too many "direct" pointers in the mode would waste space. Using (the three kinds of) indirect blocks allows very large files to be supported, while not penalising small ones.

Pipes

Unix pipes are implemented by using a shared memory buffer and a producer-consumer model; the writer can insert data (byte by byte) only up to a certain threshold – attempting to write more will cause them to be blocked. The read can read any existing data, but will block if there is nothing to be read. Pipes just look like files (in the sense that they're accessed by file descriptors); however the original ones had no filesystem-visible name, and hence could only be used for IPC between processes with a common parent.

Using links for reliability

As outlined, the scheme does not provide the resilience the system administrator would like; firstly, since hard links all refer to the same inode, a mistaken 'edit' will propagate to all snapshots; secondly, since hards links must refer to files on the same filesystem, there is not really any additional protection to hardware faults. It does mitigate against unintentional deletions since a file must be deleted from every snapshot in order to be truly deleted (although this is arguably a disadvantage too). Finally, the cost will not be quite so 'free' since of course directories must be copied rather than hardlinked to.