Lecture 15

The OS file system

A. File concept

- A file is a contiguous logical address space, managed as an identifiable entity.
- Types:
  - Data
    - numeric
    - character
    - binary
  - Code

- The file system implements all the operations required to manage files.

B. File structure

- None - sequence of words, bytes.
- Simple record structure:
  - Lines of
    - fixed length, or
    - variable length.
- Complex Structures:
  - Formatted document;
  - Re-locatable load file.
- Can simulate the last two with first method by inserting appropriate control characters.
- Controlled by:
  - the operating system;
  - the program.

File attributes

- **Name –** the only information kept in human-readable form;
- **Identifier –** unique tag (number) identifies file within the file system;
- **Type –** needed for systems that support different types;
- **Location –** pointer to file location on device;
- **Size –** current file size;
- **Protection –** controls who can do reading, writing, executing;
- **Time, date, and user identification –** data for protection, security, and usage monitoring.

- Information about files are kept in the *directory structure*, which is maintained on the disk.

C. File operations

- File is an **abstract data type.**
- Operations:
  - **Create**
  - **Write**
  - **Read**
  - **Reposition within file**
  - **Delete**
  - **Truncate**

  - *Open(F_i)* – search the directory structure on disk for entry F_i, and move the content of the entry to the main memory.
  - *Close (F_i)* – move the content of entry F_i in memory to directory structure on disk.

Open files

- Open files management requires:
  - *file pointer*: pointer to the last read/write location, per process that has the file open;
  - *file-open count*: counter of number of times a file is open – to allow removal of data from open-file table when last processes closes it;
  - *disk location of the file*: cache of data access information;
  - *access rights*: per-process access mode information.

Open file locking

- Provided by some operating systems and file systems.
- Mediates access to a file.
- Mandatory or advisory:
  - **Mandatory** – access is denied depending on locks held and requested;
  - **Advisory** – processes can find status of locks and decide what to do.

File types

file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm, mp3, avi	binary file containing audio or A/V information

D. Access methods

- **Sequential Access**

  read next
  write next
  reset
  no read after last write
  (rewrite)

- **Direct Access**

  read *n*
  write *n*
  position to *n*
  read next
  write next
  rewrite *n*

where n = relative block number.

E. File system services

- Create/remove file, address file, open/close file, read/write an open file, fetch/modify metadata of a file.

- Shared/exclusive access to a file.

- Let's consider a process P. Q is the queue of processes awaiting access to that file. The algorithm for *exclusive file access* is the following:

1. If Q is non-empty, then add P to the tail of Q and return.
2. If no process has currently exclusive access to the file, P gets access and return;
3. If P requests read-only access and Q is empty and the processes with current access are readers, P gets access and return.
4. Add P to the tail of Q and return.

F. Metadata

- Metadata is data about the file: name, size, last modification date, owner, protection codes,… managed by the OS and in some cases by applications as well.
- The file type is supported by some OS.
- File types are used in a number of ways, for example they can be used to control certain aspects of reading/writing, such as end-of-line conventions for text files.
- Files that are compressed are automatically decompressed when read.
- Another aspect is assigning an application to the file.

G. The FS design

- The FS process can be part of the kernel or not, running as a distinct process in the user space. In the latter case, there are some issues:
  - How does it handle system calls?
  - How does it access process's memory space?
  - How can it obtain process information that is stored in process tables?
  - How does it access the device drivers?

- One possible answer: a new protocol of message passing between processes.

Management data structures

- The FS uses two important data structures for file management.
- The *open file table* has entries for all open files. Each entry stores file attributes (size, device, if it is shared or not,…).
- The *mount table*. Making a file system accessible is called mounting (un-mounting is the process of removing a file system from the accessible set). The details of mounting a file system vary considerable from one system to another.
- In some cases a device is referenced explicitly and the fs metadata is read at that time. For other systems, the new fs is added into an overall uniform naming scheme. In these systems, the file names are determined by the mount operation rather then by the physical operation.