Operating Systems Concept

- What is an operating system?
- Operating system architecture
- Process concept
- CPU scheduling
- Memory management
- File and I/O systems

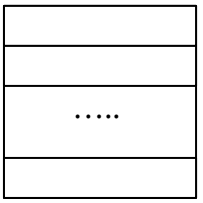
File System

- Provides a mechanism for on-line storage of and access to both data and programs
- Components
  - Files: logical unit abstracted by the OS.
  - A directory structure: special files to organize and provide information to files.
- What should be considered?
  - File attributes such as name, file operations such as seek, file types such as executable, file structures such as those for Word.

* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 337-342,346-348.

File Systems

File: a sequence of bytes



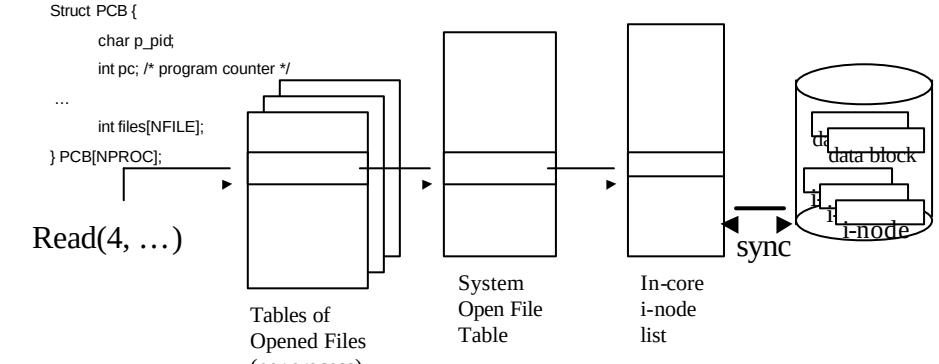
a sequence of (logical) blocks

- File Methods
  - Sequential Access
    - Basic Operations
      - READ, WRITE + file pointers
  - Direct Access
    - Basic Operations
      - READ N or Write N, where N is the relative block number.

File System – A UNIX Approach

- Process Control Block

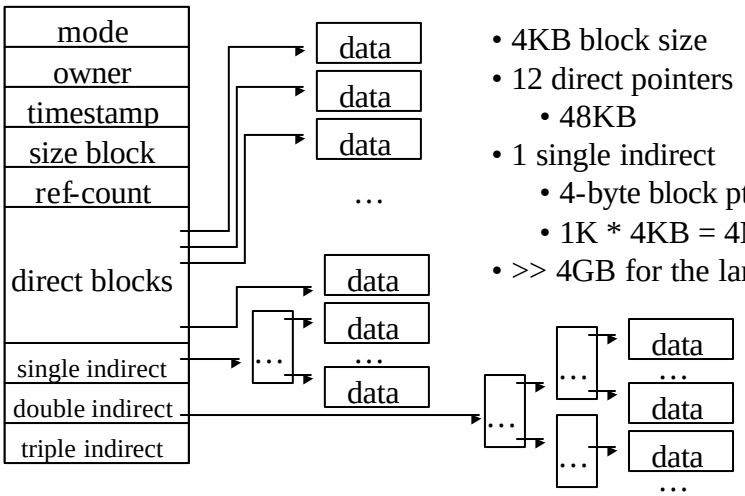
```
Struct PCB {
        char p_pid;
        int pc; /* program counter */
…
        int files[NFILE];
} PCB[NPROC];
```



open — create an entry (file current position, etc) — file-open-count++ — Load the corresponding i-node if it is absent.

* The i-node structure of a file includes info regarding the disk location of the file.

BSD UNIX i-node

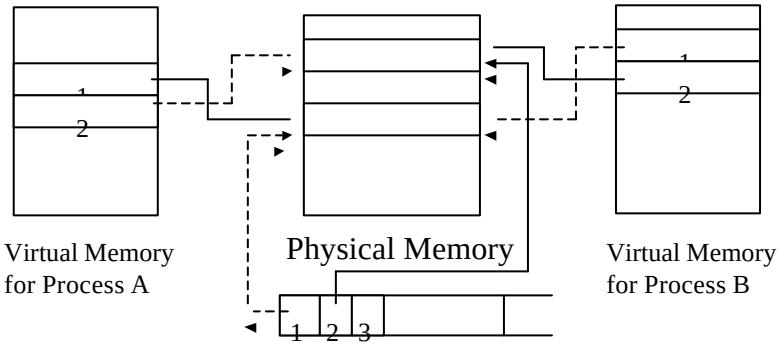


- 4KB block size
- 12 direct pointers
  - 48KB
- 1 single indirect
  - 4-byte block ptr
  - 1K * 4KB = 4MB
- >> 4GB for the largest file!

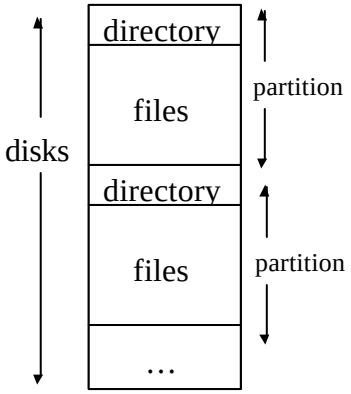
* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 380.

Memory-Mapped File

- Allow a part of the virtual address space to be logically associated with a section of a file.



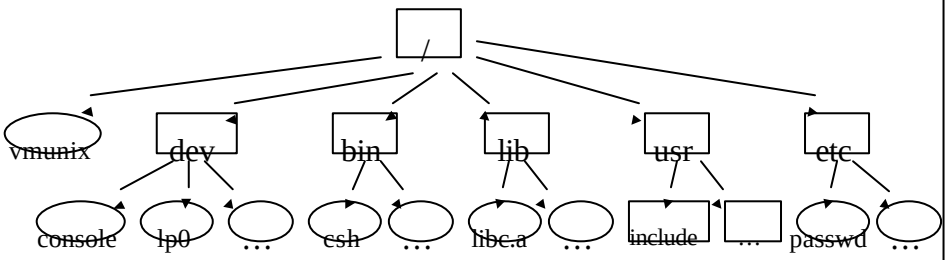
File System – Directory Structure



- Partition (/Volume):
  - a low level structure in which files and directories reside.
- Directory:
  - Records info for "all" files on a partition.

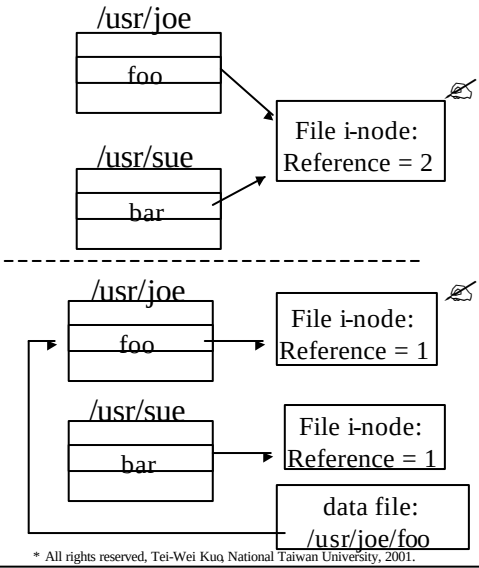
* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 349,354-358.

File System – Tree-Based Directory



- Path name
  - Specify a file by listing "node" names from the root to the corresponding node in the structure
    - /users/userA/info-sys/test

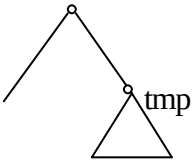
Sharing of Files



- Hard Link
  - Each directory entry creates a hard link of a filename to the i-node that describes the file's contents.
- Symbolic Link (Soft Link)
  - It is implemented as a file that contains a pathname.
  - Example: Shortcut on Windows

File Systems - Mounting

- (name of the device, mount point)



Use an appropriate device driver to read the device directory and verify the format => mount!

- Mount point: the location within the file structure at which to attach the file system.

UNIX: manual mounting

Protection

- How to prevent improper access?
  - Access control!
    - Read, Write, Execute, Append, Delete, List, etc
- Approaches:
  - Complete isolation
  - No protection at all
  - Controlled access by limiting the “types” of file access based on some factors:

* “Operating system concept”, Silberschatz and Galvin, Addison Wesley, pp. 360-362.

Protection

- Access user list for each file/directory
  - { < user-ID, allowed access types> … }
  - Tedious in maintenance and variable directory sizes -> condense such info (UNIX)
    - Read/write/execute over owner/grp/others
    - Issues
      - Control in group memberships
      - Membership per user?
- A password for each file/directory
  - A large number of passwords
  - Different passwords for different levels of protection?

Operating Systems Concept

- What is an operating system?
- Operating system architecture
- Process concept
- CPU scheduling
- Memory management
- File and I/O systems

I/O Subsystems

- Why?
  - Provide the simplest interface to the rest of the system
  - Optimize I/O for the maximum concurrency
- Variations:
  - Block vs Character I/O
  - Sequential/Random Access
  - Synchronous/Asynchronous Transfer
  - Dedicated/Share
  - Read-Only/Read-Write

* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 398, 408-410, 414-415.

I/O Subsystems

- Conflicting trends
  - Increasing standardization of software/ hardware interfaces
  - Increasing broad variety of I/O devices
    - Device drivers which provide a uniform device-access interface to the I/O subsystem.
- How the I/O system improves the efficiency of the computer?
  - Schedule I/O operations, e.g., those on disks.
  - Use techniques such as buffering, caching, or spooling.

Principles in Doing I/O

- Reduce the number of context switches.
- Reduce the number of data copyings.
- Reduce the frequency of interrupts
  - Large transfer, smart controller, etc.
- Increase concurrency
  - DMA controller
- Move processing primitives into hardware.
- Balance CPU, memory subsystems, bus, and I/O memory.

* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 424-425.