Principles of Operating Systems

CS 446/646

6. File System

René Doursat

Department of Computer Science & Engineering
University of Nevada, Reno

Spring 2006

Principles of Operating Systems

CS 446/646

0. Course Presentation
1. Introduction to Operating Systems
2. Processes
3. Memory Management
4. CPU Scheduling
5. Input/Output
6. **File System**
7. **Case Studies**

Principles of Operating Systems

CS 446/646

6. File System

a. **Overview of the File System**

b. **User Interface: Files**

c. **User Interface: Directories**

d. **File System Implementation**

Principles of Operating Systems

CS 446/646

6. File System

a. **Overview of the File System**

b. **User Interface: Files**

c. **User Interface: Directories**

d. **File System Implementation**

6.a Overview of the File System

- The need for long-term storage
  - ✓ it must be possible to store a very **large amount** of information
    - memory is too small to hold large databases of records, for example airline reservations, bank accounts, etc.
  - ✓ the information must **survive** the termination of the processes using it
    - it must also not go away if the computer crashes
  - ✓ multiple processes must be able to **access** the information **concurrently**
    - for example, a phone directory should not be only stored inside the address space of a single process
  - → *store information on disk, and group it in units called* ***files***

6.a Overview of the File System

➢ Chart of Operating System Responsibilities

§E – The O/S is responsible for providing a uniform logical view of information storage

- ✓ the O/S defines a logical unit of storage, the **file**, and groups files in a hierarchy of **directories**
- ✓ the O/S supports primitives for manipulating files and directories (create, delete, rename, read, write, etc.)
- ✓ the O/S ensures data confidentiality and integrity
- ✓ the O/S implements files on stable (nonvolatile) storage media
- ✓ the O/S keeps a mapping of the logical files onto the physical secondary storage

6.a Overview of the File System

- The file system is the most visible aspect of an O/S
  - ✓ files are managed by the O/S
  - ✓ how files are
    - structured
    - named
    - accessed
    - used
    - protected
    - implemented

  . . . are major topics in operating system design

6.a Overview of the File System

- Users' standpoint vs. designers' standpoint
  - ✓ for the O/S users
    - the most important aspect is how files **appear** to them
    - how files are named and protected
    - what operations are allowed, etc.
  - ✓ for the O/S designers
    - must decide whether to **implement** files with linked lists, tables, etc.
    - how to map file blocks to disk sectors
    - how to keep track of free storage, etc.

Principles of Operating Systems

CS 446/646

6. File System

a. Overview of the File System

b. User Interface: Files

c. User Interface: Directories

d. File System Implementation

6.b User Interface: Files

- Files are an abstraction mechanism
  - ✓ the concept of “file” is the central element of the file system
  - ✓ a file is a complete collection of data (as text or a program) treated by a computer as a unit especially for purposes of input and output
  - ✓ files provide a convenient way to store information on the disk and read it back later
  - ✓ they shield the user from the details of where the information is stored and how the disk works

6.b User Interface: Files

File naming

- Naming is the most important aspect of abstraction
  - ✓ when a process creates a file, it gives it a name; when it terminates, the file continues to exist
  - ✓ naming rules vary from system to system
    - allowed name length can go from 8 to 255 characters
    - UNIX systems distinguish between uppercase and lowercase, MS-DOS and Windows do not
    - many systems support two-part, period-separated naming: the second part is called the **extension**
    - in UNIX, the extension is a user convention; not enforced
    - Windows is extension-aware and associates files with specific applications

6.b User Interface: Files

File naming

file type	usual extension	function
executable	exe, com, bin or none	read 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, rrf, doc	various word-processor formats
library	lib, a, so, dll, mpeg, mov, rm	libraries of routines for programmers
print or view	arc, zip, tar	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	binary file containing audio or A/V information

Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

Common file types & extensions

6.b User Interface: Files

File structure

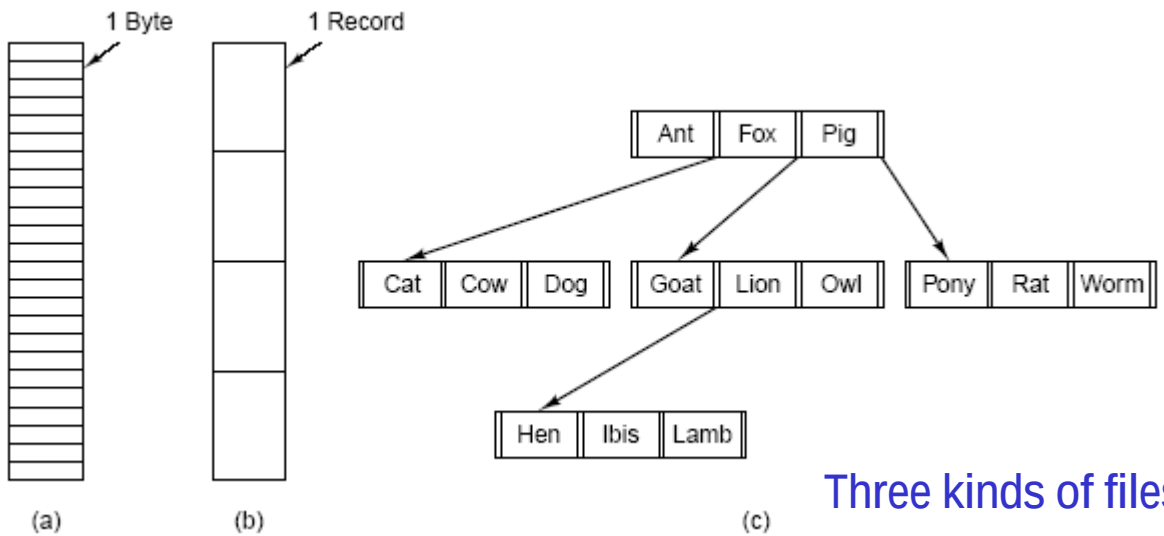
- A file can be internally structured in several ways

a) **pure byte sequence** — O/S doesn't care about the contents; all meaning imposed by user application; generic O/S (UNIX, Win)

closer to database system techniques

b) **record sequence** — fixed or variable-length records with internal structure; historical 80-column punch card systems

c) **tree** — key-accessible records; mainframes commercial data processing



Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

Three kinds of files

6.b User Interface: Files

File types

- An O/S supports different types of files

Windows

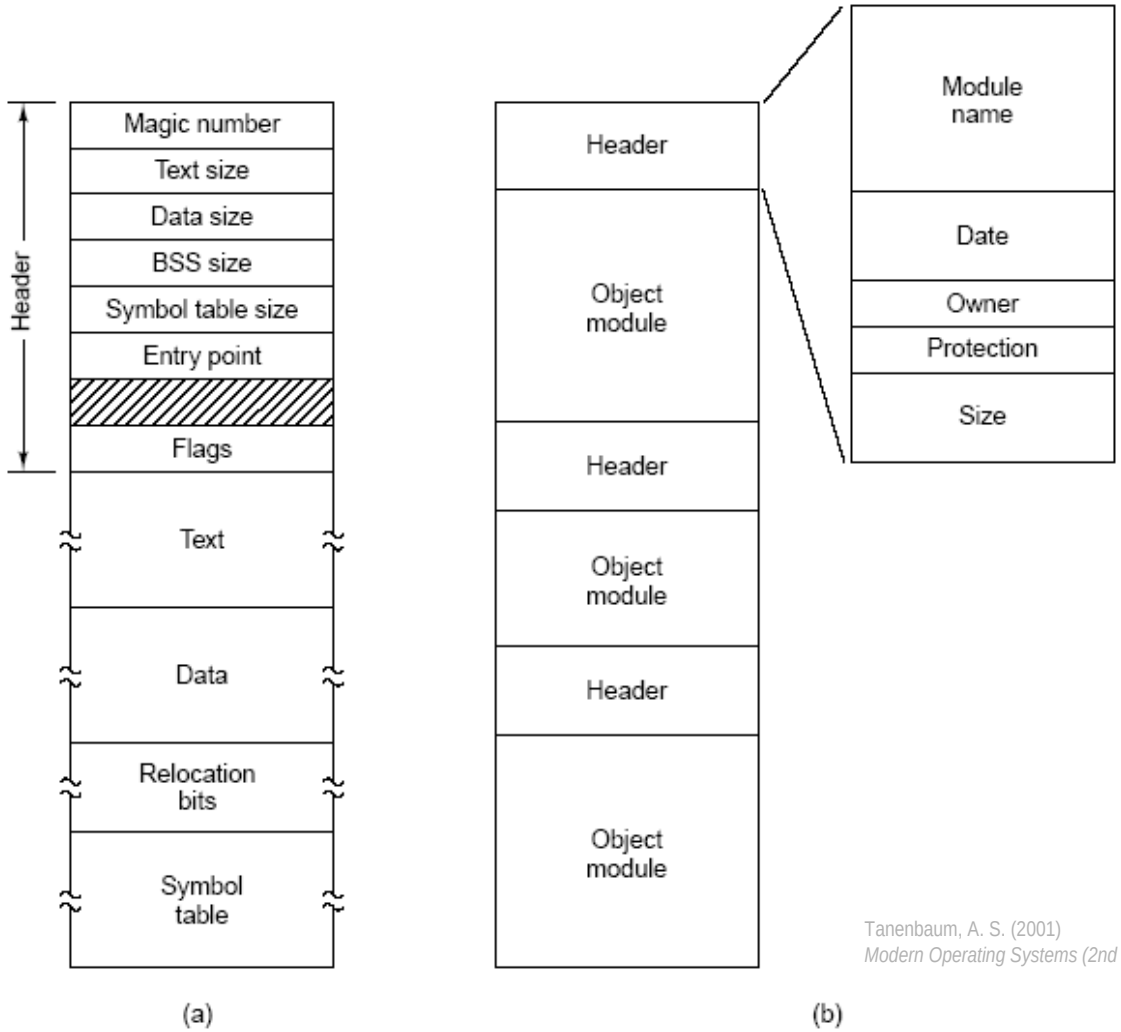
- ✓ regular files
  - the files that contain user information, ASCII or binary
- ✓ directories (directory files)
  - system files that contain information about the file system organization

UNIX

- ✓ character special files
  - used to model serial (character-mode) I/O devices: terminals, network
- ✓ block special files
  - used to model parallel (block-mode) I/O devices: disks

6.b User Interface: Files

File types



Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

(a) An executable file and (b) an archive of unlinked compiled modules

6.b User Interface: Files

File attributes

- The O/S associates management information with files
  - ✓ in addition to its name and data, a file also has **file attributes**
  - ✓ the list of attributes varies considerably from system to system, but typically:
    - file's owner and protection
    - various bit flags: hidden, read/write, etc.
    - record length, key, etc. for record-structured files
    - timestamps: created, accessed, modified, etc.
    - size values
  - ✓ just as process control blocks (PCBs), the O/S maintains file control blocks (FCBs) → *see file system implementation*

6.b User Interface: Files

File attributes

Attribute	Meaning
Protection	Who can access the file and in what way
Password	Password needed to access the file
Creator	ID of the person who created the file
Owner	Current owner
Read-only flag	0 for read/write; 1 for read only
Hidden flag	0 for normal; 1 for do not display in listings
System flag	0 for normal files; 1 for system file
Archive flag	0 for has been backed up; 1 for needs to be backed up
ASCII/binary flag	0 for ASCII file; 1 for binary file
Random access flag	0 for sequential access only; 1 for random access
Temporary flag	0 for normal; 1 for delete file on process exit
Lock flags	0 for unlocked; nonzero for locked
Record length	Number of bytes in a record
Key position	Offset of the key within each record
Key length	Number of bytes in the key field
Creation time	Date and time the file was created
Time of last access	Date and time the file was last accessed
Time of last change	Date and time the file has last changed
Current size	Number of bytes in the file
Maximum size	Number of bytes the file may grow to

Some possible file attributes

Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

6.b User Interface: Files

File operations

- Most common system calls related to files
  - ✓ create/delete
    - creates a file with no data, initializes file attributes
  - ✓ open/close
    - loads file attributes and disk addresses in memory
  - ✓ read/write, append
    - transfers data from/to a buffer starting at a current position
  - ✓ seek
    - in random access files: repositions file pointer for read/write
  - ✓ get/set attributes, rename
    - some attributes are user-settable (name, protection flags)

6.b User Interface: Files

File operations

```
#define BUF_SIZE 4096                   /* use a buffer size of 4096 bytes */
#define OUTPUT_MODE 0700                /* protection bits for output file */

int main(int argc, char *argv[])
{
    int in_fd, out_fd, rd_count, wt_count;
    char buffer[BUF_SIZE];

    if (argc != 3) exit(1);             /* syntax error if argc is not 3 */

    /* Open the input file and create the output file */
    in_fd = open(argv[1], O_RDONLY);    /* open the source file */
    if (in_fd < 0) exit(2);             /* if it cannot be opened, exit */
    out_fd = creat(argv[2], OUTPUT_MODE);   /* create the destination file */
    if (out_fd < 0) exit(3);            /* if it cannot be created, exit */

    /* Copy loop */
    while (TRUE) {
        rd_count = read(in_fd, buffer, BUF_SIZE);  /* read a block of data */
    if (rd_count <= 0) break;           /* if end of file or error, exit loop */
        wt_count = write(out_fd, buffer, rd_count);  /* write data */
        if (wt_count <= 0) exit(4);     /* wt_count <= 0 is an error */
    }

    /* Close the files */
    close(in_fd);
    close(out_fd);
    if (rd_count == 0)                  /* no error on last read */
        exit(0);
    else
        exit(5);                        /* error on last read */
}
```

Tanenbaum, A. S. (2001)
Modern Operating Systems (2nd Edition).

Principles of Operating Systems

CS 446/646

6. File System

a. Overview of the File System

b. User Interface: Files

c. User Interface: Directories

d. File System Implementation

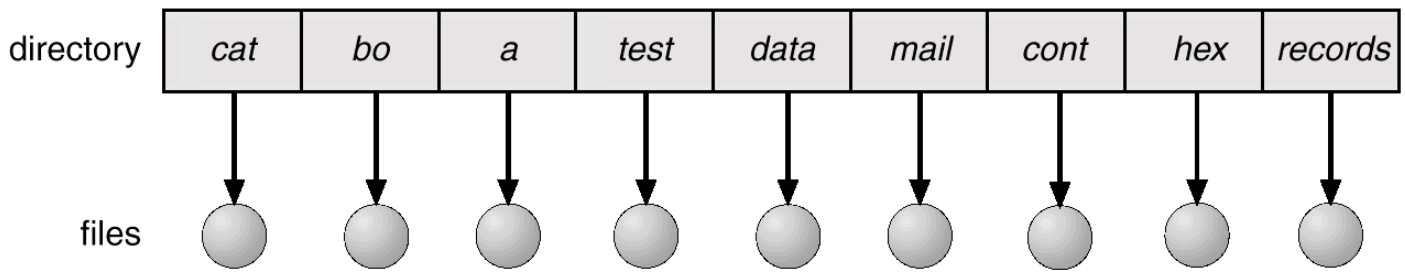
6.c User Interface: Directories

- Directories are special files that keep track of other files
  - ✓ the collection of files is systematically organized
  - ✓ first, disks are split into partitions that create logical volumes (can be thought of as “virtual disks”)
  - ✓ second, each partition contains information about the files within
  - ✓ this information is kept in entries in a **device directory** (or volume table of contents)
  - ✓ the directory is a symbol table that translates file names into their entries in the directory
    - it has a logical structure
    - it has an implementation structure (linked list, table, etc.)

6.c User Interface: Directories

➢ Single-level directory structure

- ✓ simplest form of logical organization: one global or **root** directory containing all the files
- ✓ problems
  - global namespace: unpractical in multiuser systems
  - no systematic organization, no groups or logical categories of files that belong together



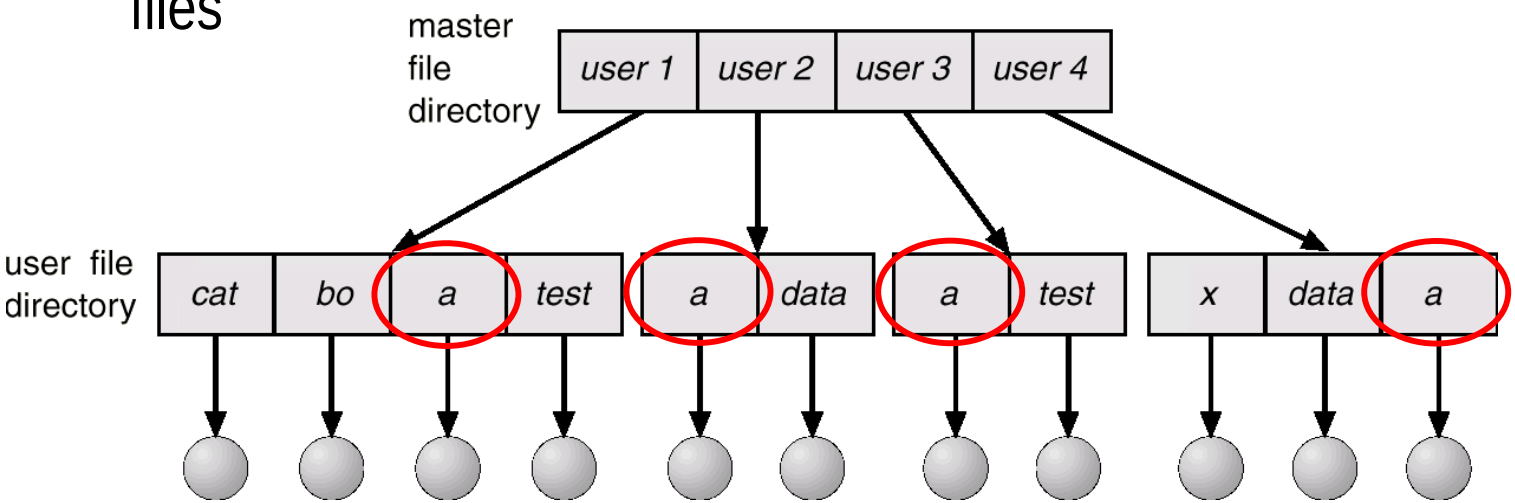
Single-level directory

Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

6.c User Interface: Directories

➢ Two-level directory structure

- ✓ in multiuser systems, the next step is to give each user their own private directory
- ✓ avoids filename confusion
- ✓ however, still no grouping: not satisfactory for users with many files

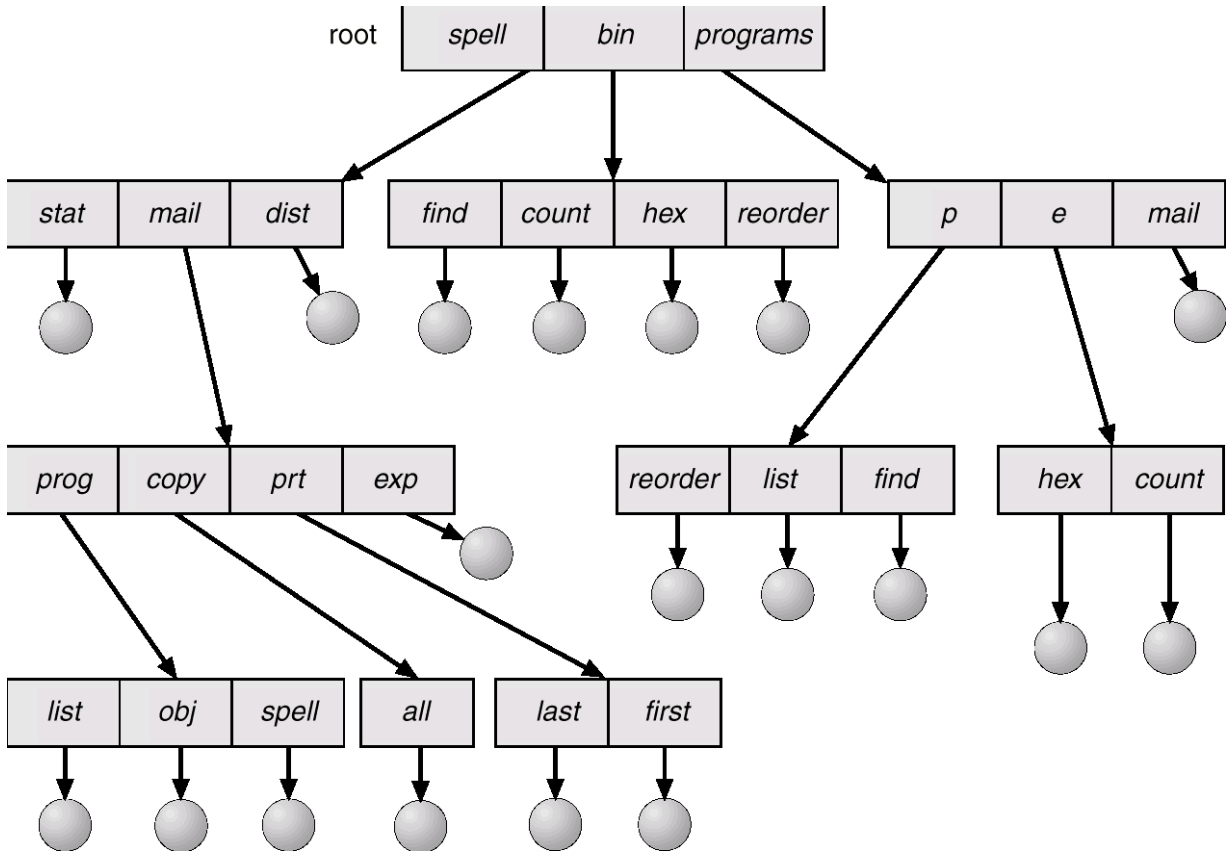


Two-level directory

Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

6.c User Interface: Directories

- Tree-structured directory structure



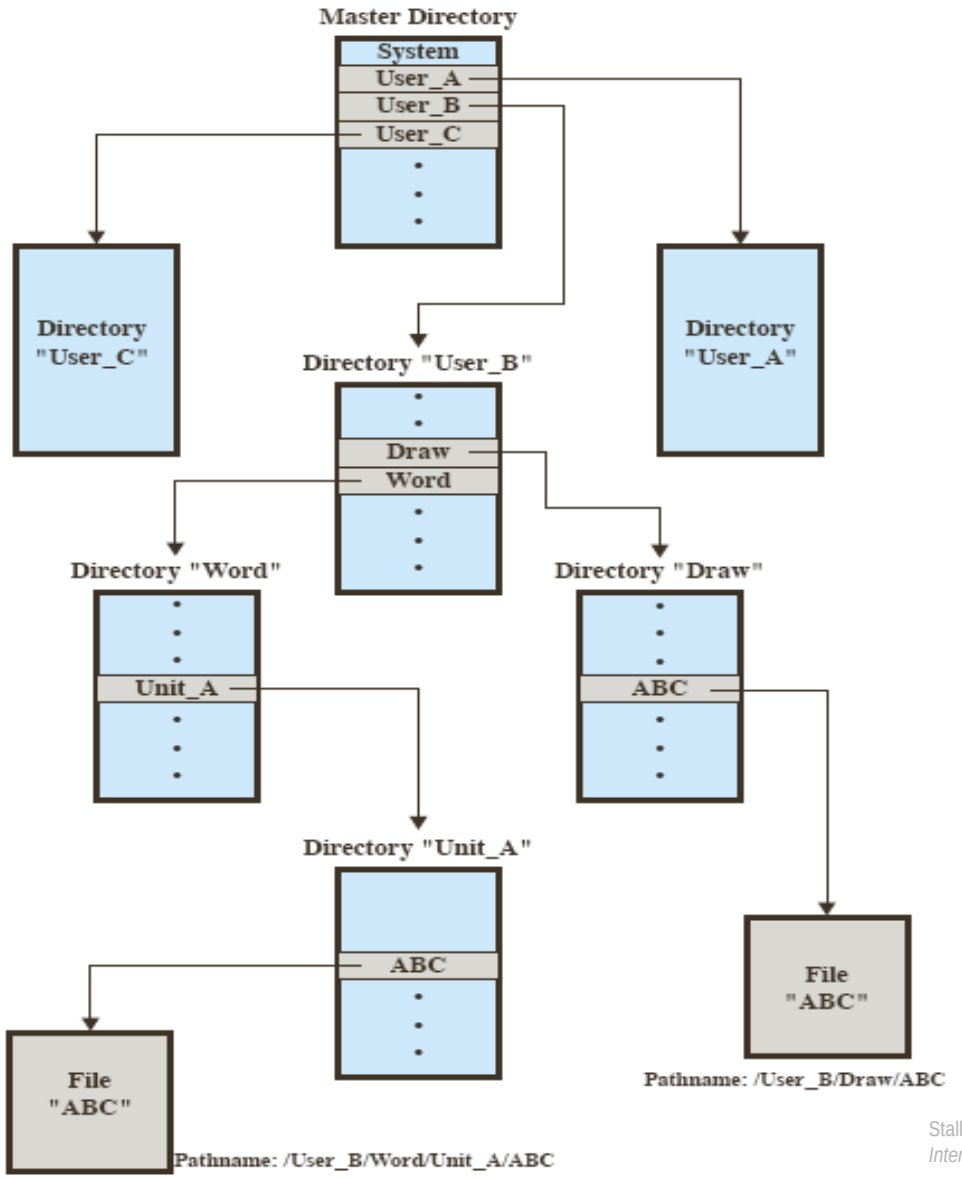
Tree-structured directory

Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

6.c User Interface: Directories

- Tree-structured directory structure
  - ✓ natural extension of the two-level scheme
  - ✓ provides a general hierarchy, in which files can be grouped in natural ways
  - ✓ good match with human cognitive organization: propensity to categorize objects in embedded sets and subsets
  - ✓ navigation through the tree relies on **pathnames**
    - absolute pathnames start from the root, example: /doursat/academic/teaching/cs446/assignment4/grades
    - relative pathnames start at from a current **working directory**, example: assignment4/grades
    - the current and parent directory are referred to as . and ..

6.c User Interface: Directories



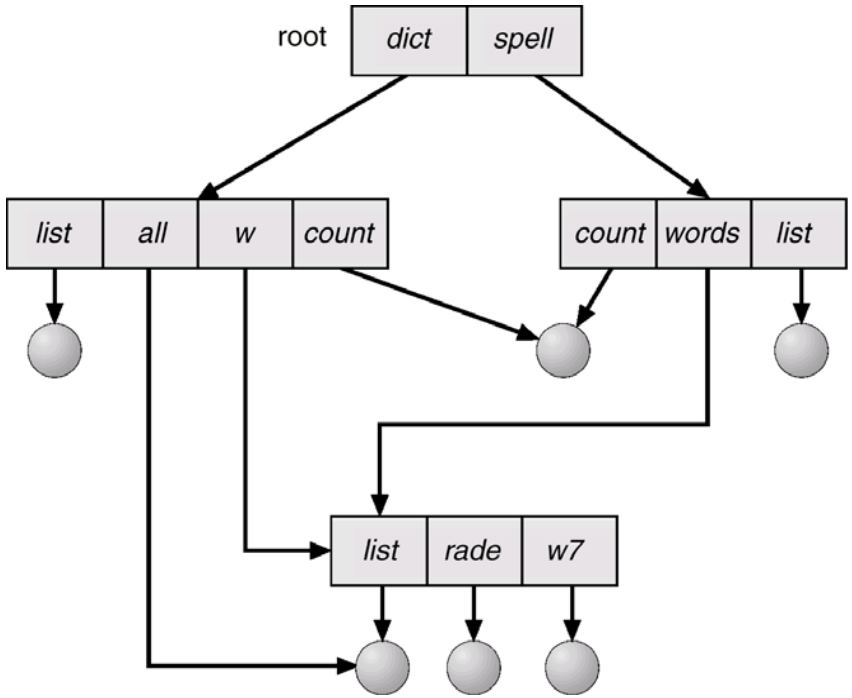
Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

6.c User Interface: Directories

- Common system calls related to directory operations
  - ✓ create/delete
    - creates or deletes an *empty* directory (except for . and ..)
  - ✓ opendir/closedir
    - loads directory attributes in memory
  - ✓ readdir
    - reads the entries in a directory (more abstract than **read**)
  - ✓ rename
    - renames a directory like a file
  - ✓ link/unlink
    - shares files by making them appear in more than one dir

6.c User Interface: Directories

- Acyclic-graph (shared file) directory structure
  - ✓ allows for different users to work on the same files while keeping their own view of the files (implemented with links)



Acyclic-graph directory

Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*