

CS3600 — SYSTEMS AND NETWORKS

NORTHEASTERN UNIVERSITY

Lecture 10: File System Interface

Prof. Alan Mislove (amislove@ccs.neu.edu)

File Concept

- Contiguous logical address space
- Types:
 - Data
 - numeric
 - character
 - binary
 - Program

File Structure

- None - sequence of words, bytes
- Simple record structure
 - Lines
 - Fixed length
 - Variable length
- Complex Structures
 - Formatted document
 - Relocatable load file
- Can simulate last two with first method by inserting appropriate control characters
- Who decides:
 - Operating system
 - Program

File Attributes

- **Name** – only information kept in human-readable form
- **Identifier** – unique tag (number) identifies file within 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

Access Methods

- **Sequential Access**

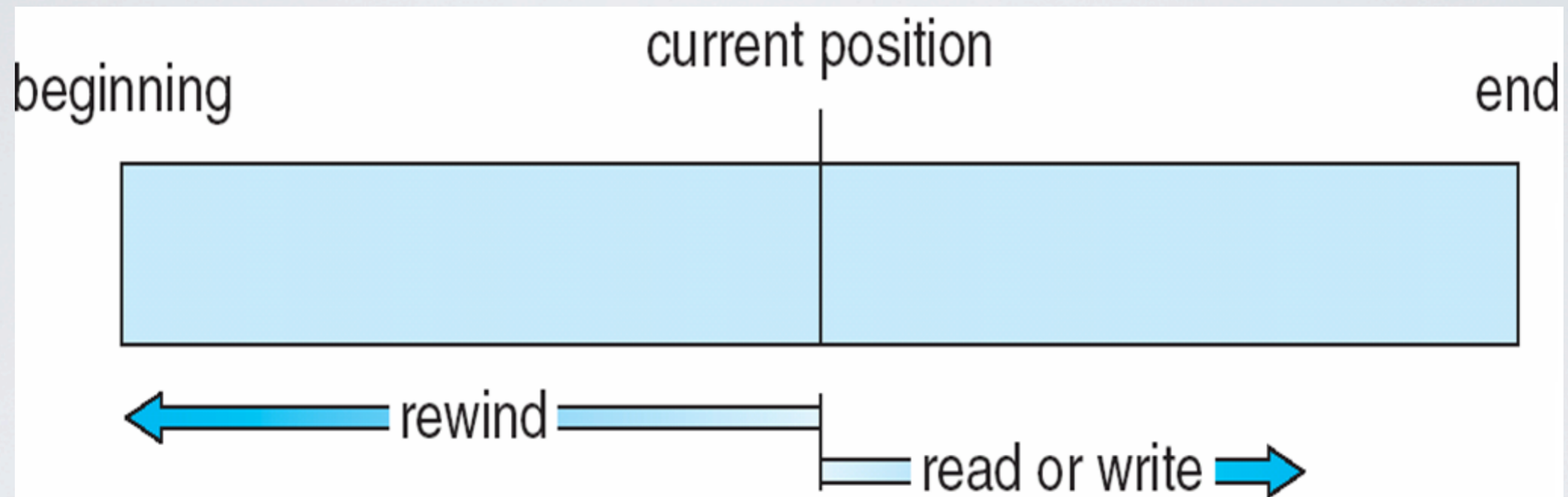
read next
write next
reset
no read after last write
(rewrite)

- **Direct Access**

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

n = relative block number

Sequential-access File



File Operations

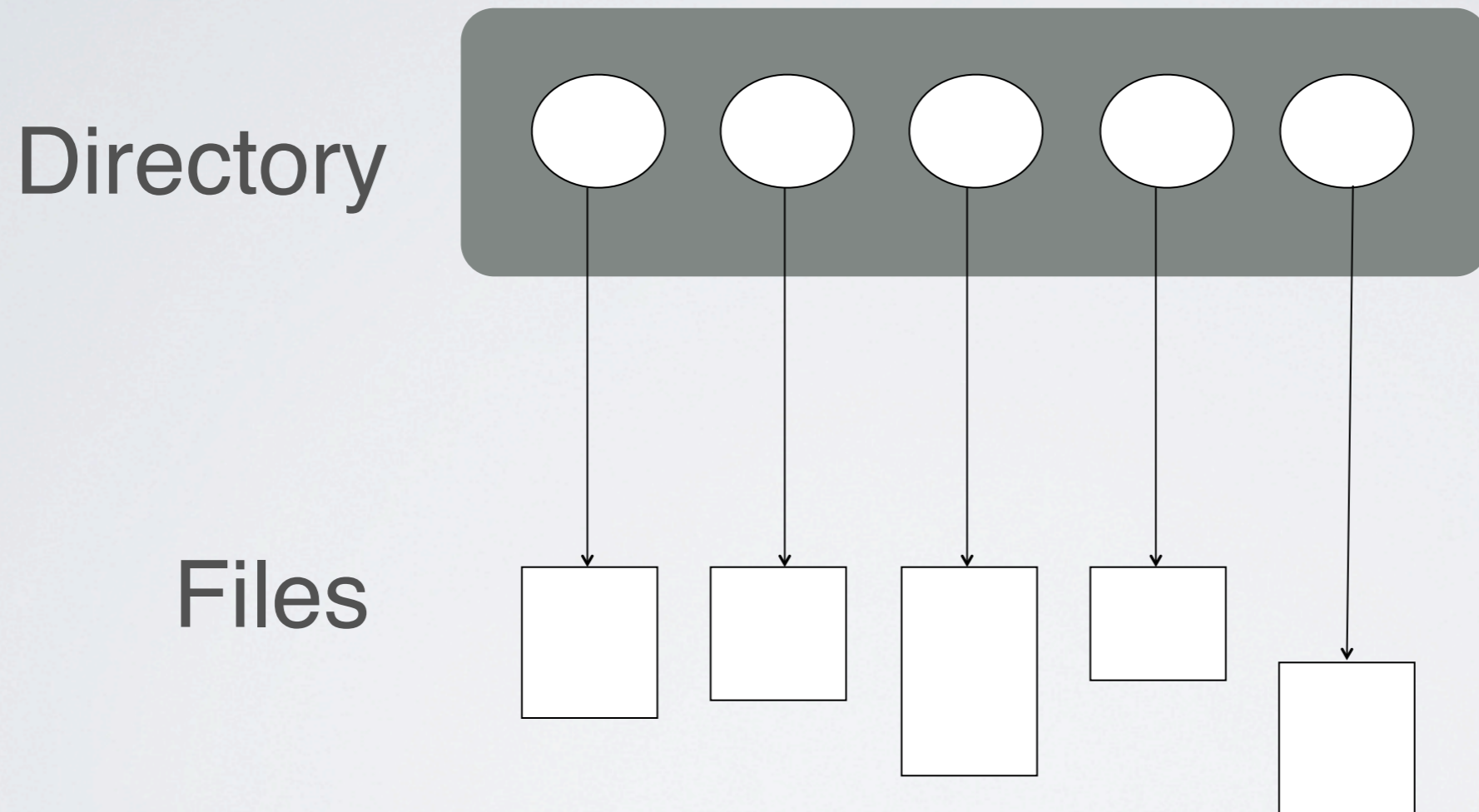
- File is an **abstract data type**
- **Create**
- **Write**
- **Read**
- **Reposition within file**
- **Delete**
- **Truncate**
- *Open(F_i)* – search the directory structure on disk for entry F_i , and move the content of entry to memory
- *Close (F_i)* – move the content of entry F_i in memory to directory structure on disk

Open Files

- Several pieces of data are needed to manage open files:
 - File pointer: pointer to 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

Directory Structure

- A collection of nodes containing information about all files



Both the directory structure and the files reside on disk
Backups of these two structures are kept on tapes

Operations Performed on Directory

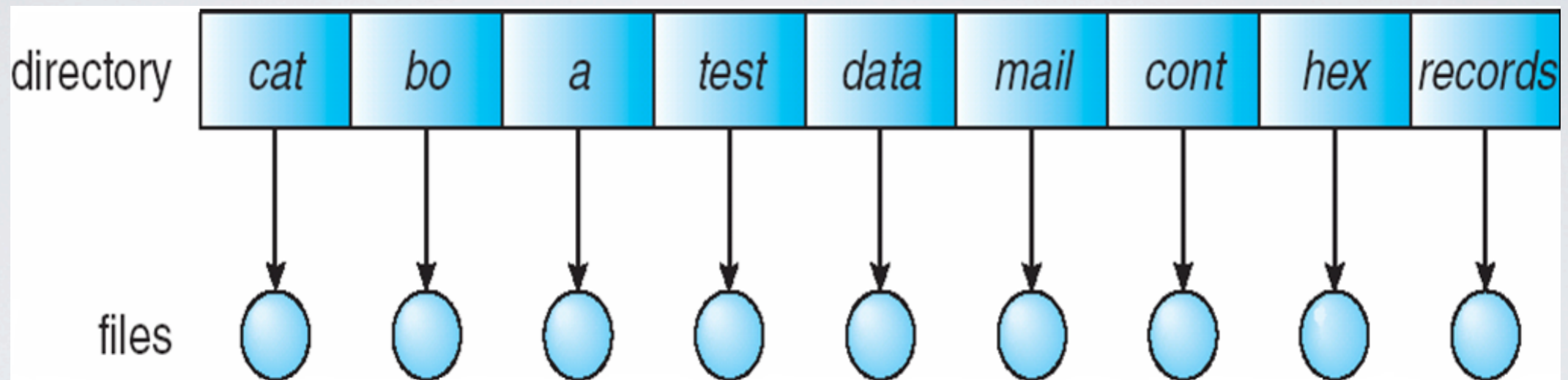
- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

Organize the Directory (Logically) to Obtain

- Efficiency – locating a file quickly
- Naming – convenient to users
 - Two users can have same name for different files
 - The same file can have several different names
- Grouping – logical grouping of files by properties, (e.g., all Java programs, all games, ...)

Single-Level Directory

- A single directory for all users

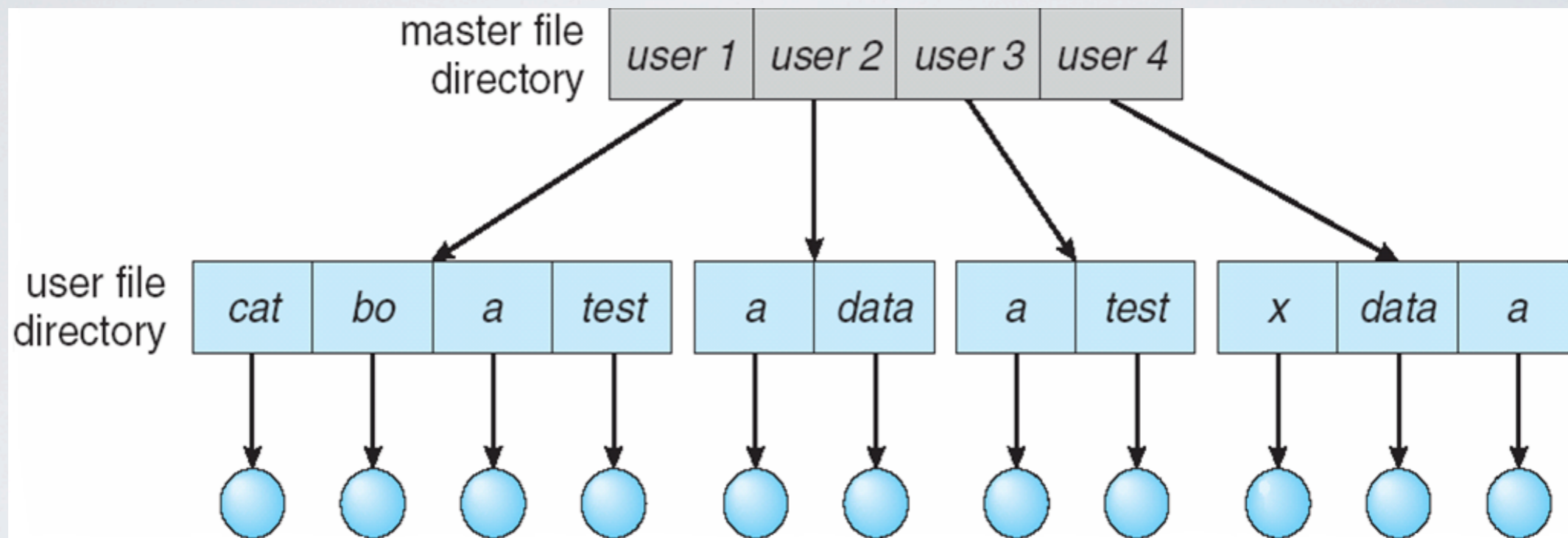


Naming problem

Grouping problem

Two-Level Directory

- Separate directory for each user



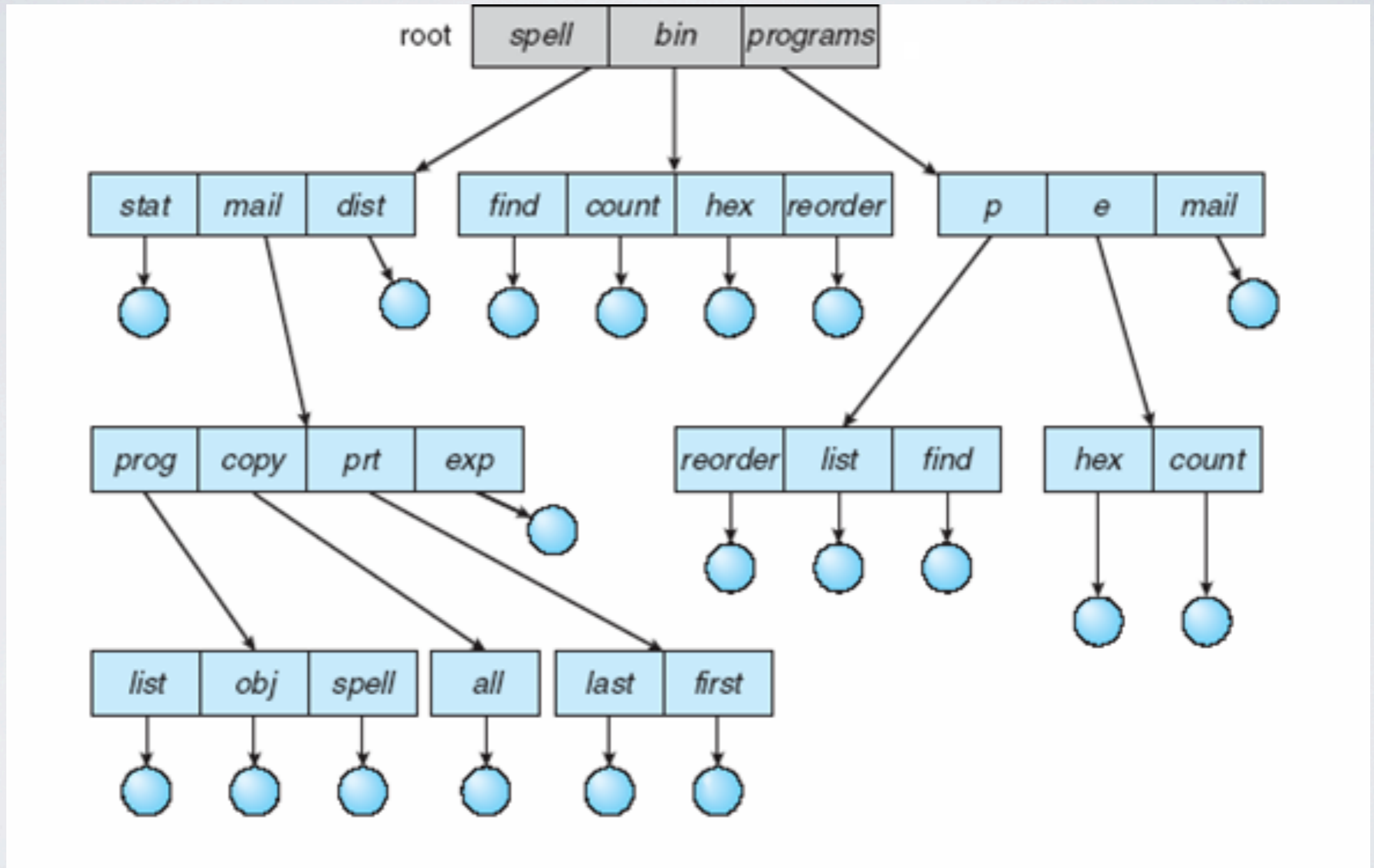
Path name

Can have the same file name for different user

Efficient searching

No grouping capability

Tree-Structured Directories



Tree-Structured Directories (Cont.)

- Efficient searching
- Grouping Capability
- Current directory (working directory)
 - **cd /spell/mail/prog**
 - **type list**

Tree-Structured Directories (Cont)

- **Absolute** or **relative** path name
- Creating a new file is done in current directory
- Delete a file

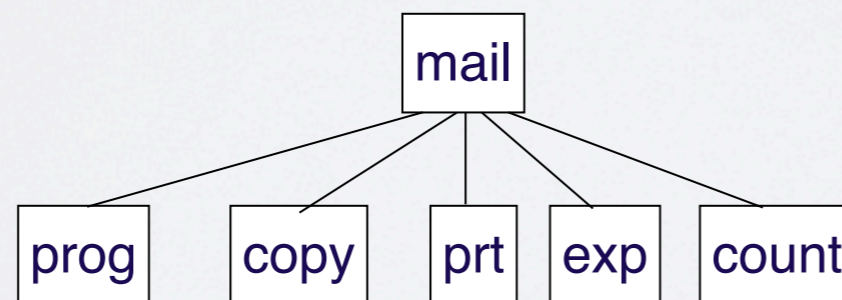
rm <file-name>

- Creating a new subdirectory is done in current directory

mkdir <dir-name>

Example: if in current directory **/mail**

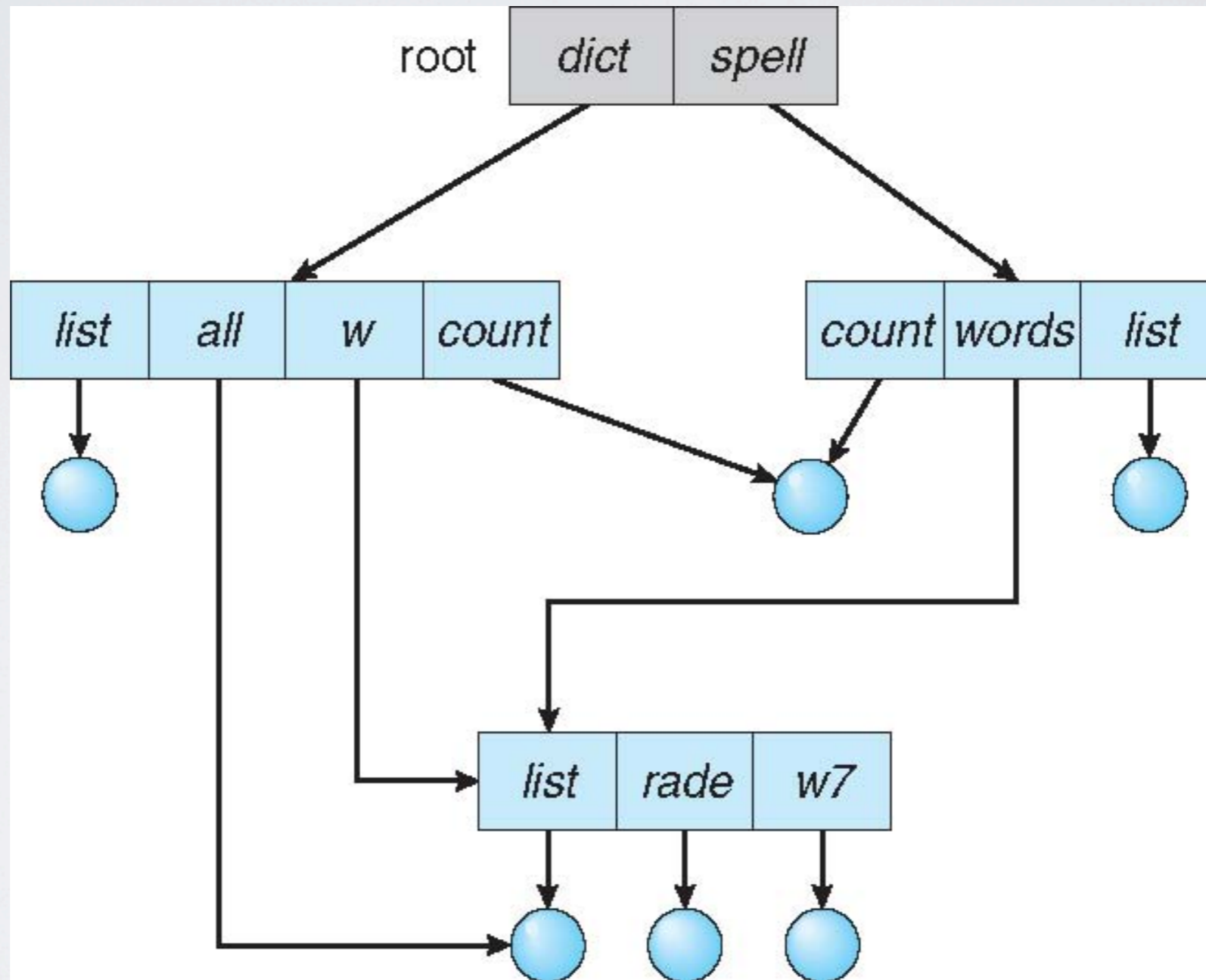
mkdir count



Deleting “mail” \Rightarrow deleting the entire subtree rooted by “mail”

Acyclic-Graph Directories

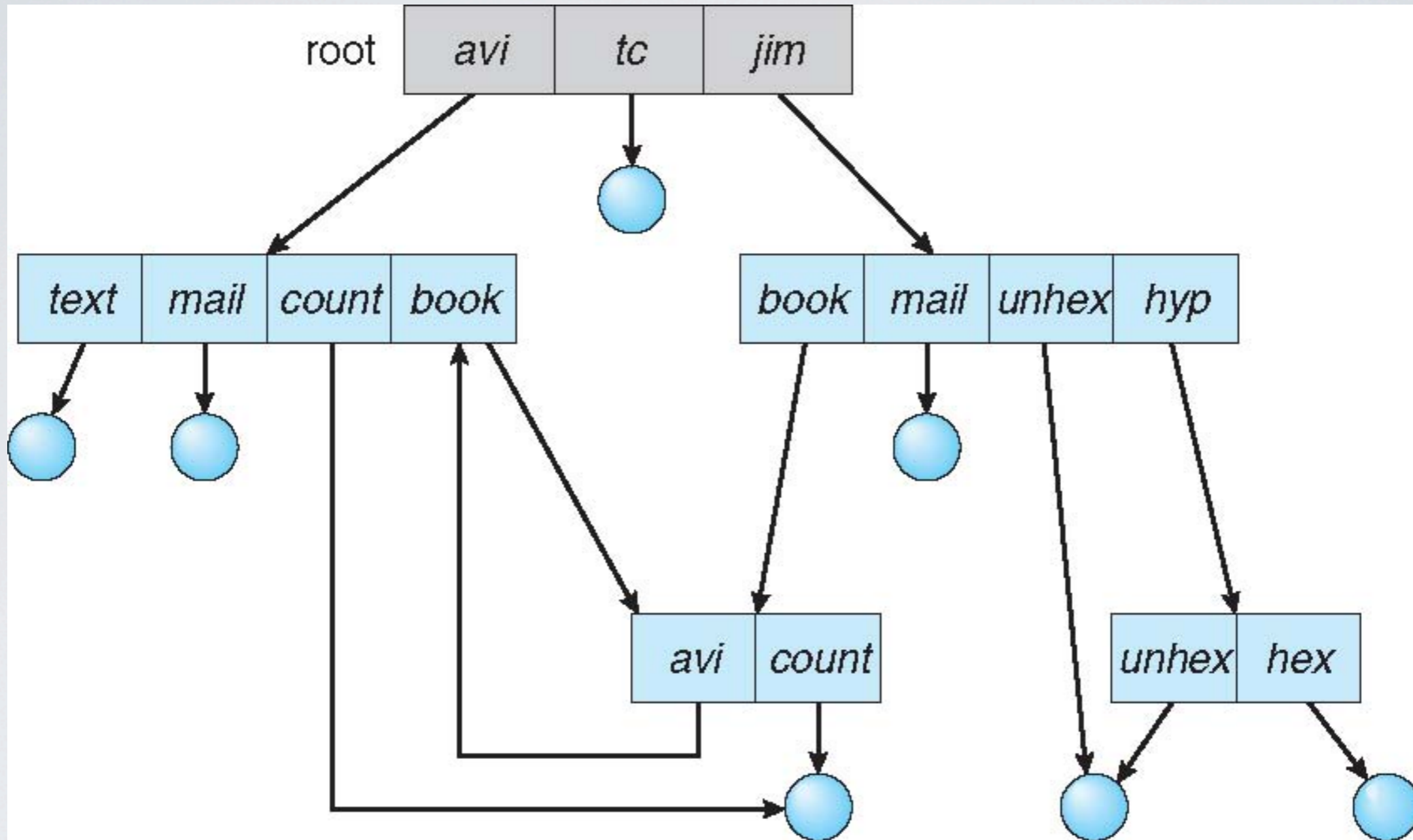
- Have shared subdirectories and files



Acyclic-Graph Directories (Cont.)

- Two different types of links
 - Hard links: Links directly to file data
 - Soft links: Links to file name
- Soft link is new directory entry type
 - **Link** – another name (pointer) to an existing file
 - **Resolve the link** – follow pointer to locate the file
- Deletes */spell/words/list* \Rightarrow */dict/w* is a dangling pointer
 - Backpointers, so we can track all pointers; variable size record problem
 - Backpointers using a daisy chain organization
 - Entry-hold-count solution

General Graph Directory



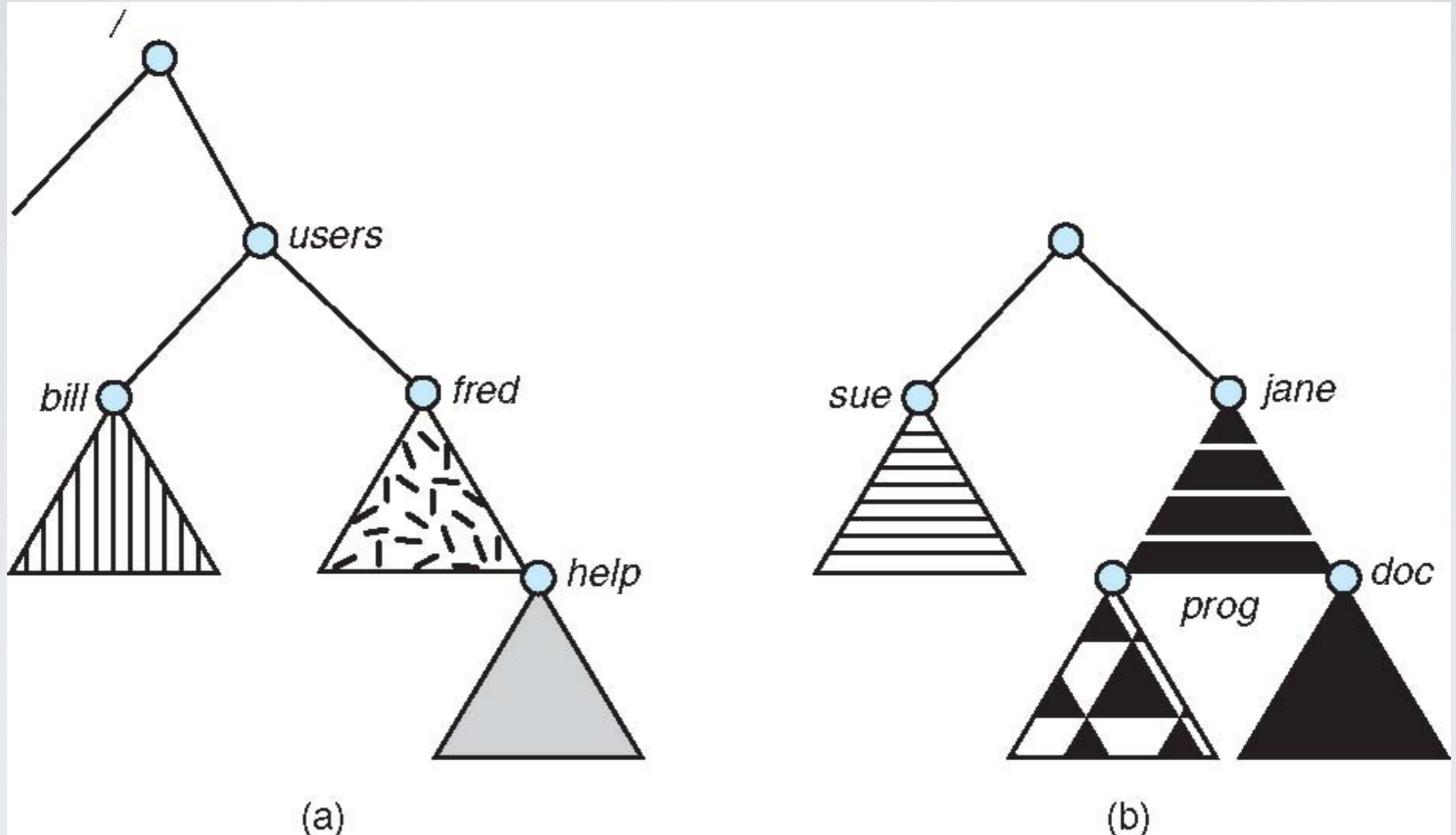
General Graph Directory (Cont.)

- How do we guarantee no cycles?
 - Allow only links to file not subdirectories
 - Garbage collection
 - Every time a new link is added use a cycle detection algorithm to determine whether it is OK

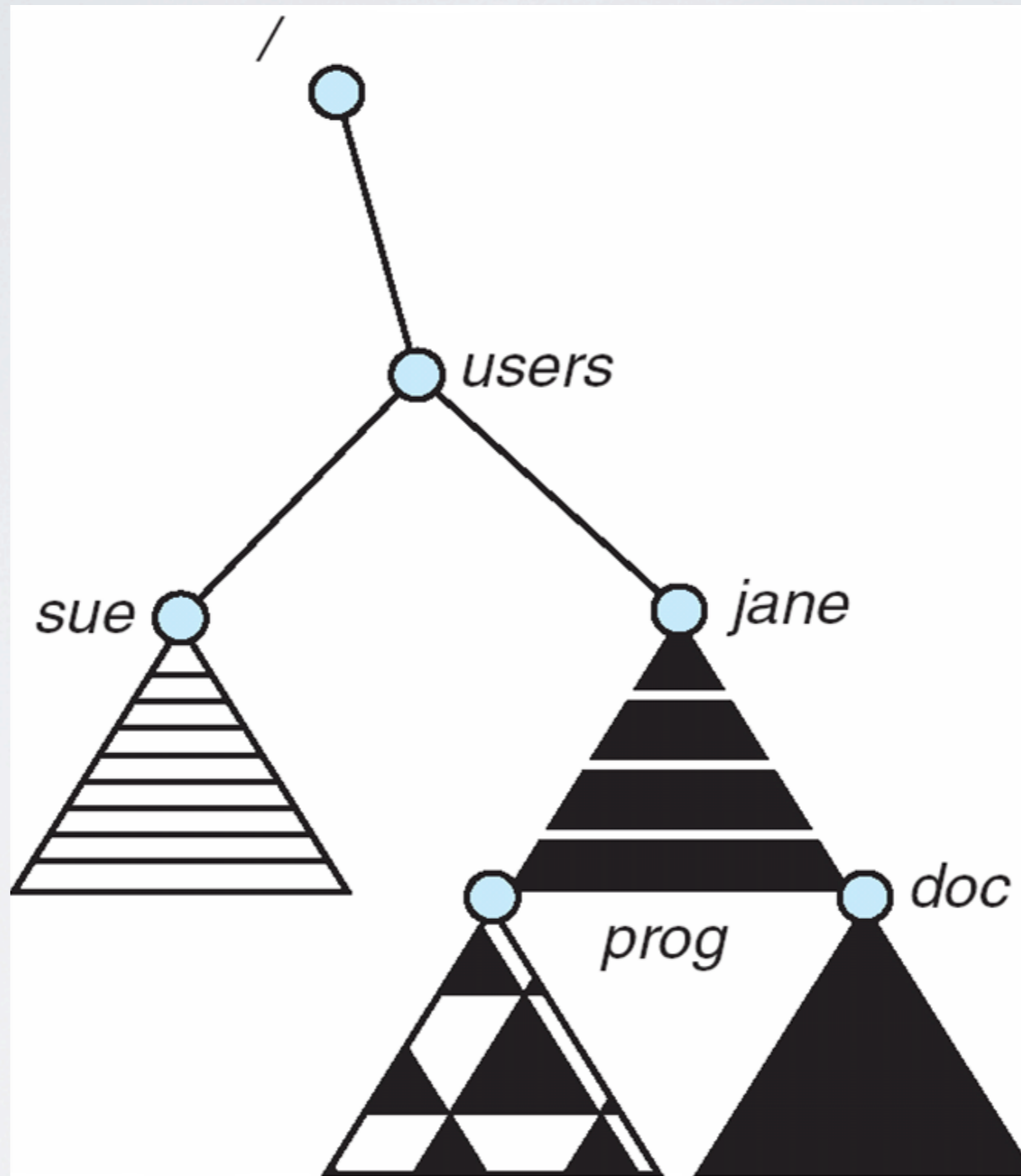
File System Mounting

- A file system must be **mounted** before it can be accessed
- A unmounted file system (i.e., Fig. 11-11(b)) is mounted at a **mount point**

(a) Existing (b) Unmounted Partition



Mount Point



File Sharing

- Sharing of files on multi-user systems is desirable
- Sharing may be done through a **protection** scheme
- On distributed systems, files may be shared across a network
- Network File System (NFS) is a common distributed file-sharing method

File Sharing – Multiple Users

- **User IDs** identify users, allowing permissions and protections to be per-user
- **Group IDs** allow users to be in groups, permitting group access rights

File Sharing – Remote File Systems

- Uses networking to allow file system access between systems
 - Manually via programs like FTP
 - Automatically, seamlessly using **distributed file systems**
 - Semi automatically via the **world wide web**
- **Client-server** model allows clients to mount remote file systems from servers
 - Server can serve multiple clients
 - Client and user-on-client identification is insecure or complicated
 - **NFS** is standard UNIX client-server file sharing protocol
 - **CIFS** is standard Windows protocol
 - Standard operating system file calls are translated into remote calls
- Distributed Information Systems (**distributed naming services**) such as LDAP, DNS, NIS, Active Directory implement unified access to information needed for remote computing

Protection

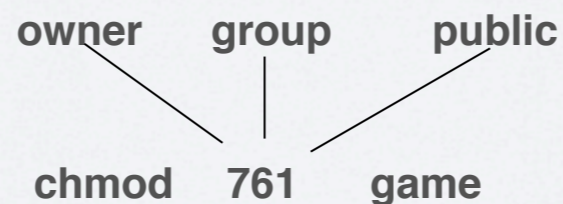
- File owner/creator should be able to control:
 - what can be done
 - by whom
- Types of access
 - **Read**
 - **Write**
 - **Execute**
 - **Append**
 - **Delete**
 - **List**

Access Lists and Groups

- Mode of access: read, write, execute
- Three classes of users

a) owner access	7	⇒	RWX 1 1 1
b) group access	6	⇒	RWX 1 1 0
c) public access	1	⇒	RWX 0 0 1

- Ask manager to create a group (unique name), say G, and add some users to the group.
- For a particular file (say *game*) or subdirectory, define an appropriate access.



Attach a group to a file
chgrp G game

A Sample UNIX Directory Listing

```
-rw-rw-r--  1 pbg  staff  31200  Sep 3 08:30  intro.ps
drwx-----  5 pbg  staff   512    Jul 8 09:33  private/
drwxrwxr-x  2 pbg  staff   512    Jul 8 09:35  doc/
drwxrwx---  2 pbg  student 512    Aug 3 14:13  student-proj/
-rw-r--r--  1 pbg  staff  9423   Feb 24 2003  program.c
-rwxr-xr-x  1 pbg  staff 20471  Feb 24 2003  program
drwx--x--x  4 pbg  faculty 512    Jul 31 10:31  lib/
drwx-----  3 pbg  staff 1024   Aug 29 06:52  mail/
drwxrwxrwx  3 pbg  staff   512    Jul 8 09:35  test/
```