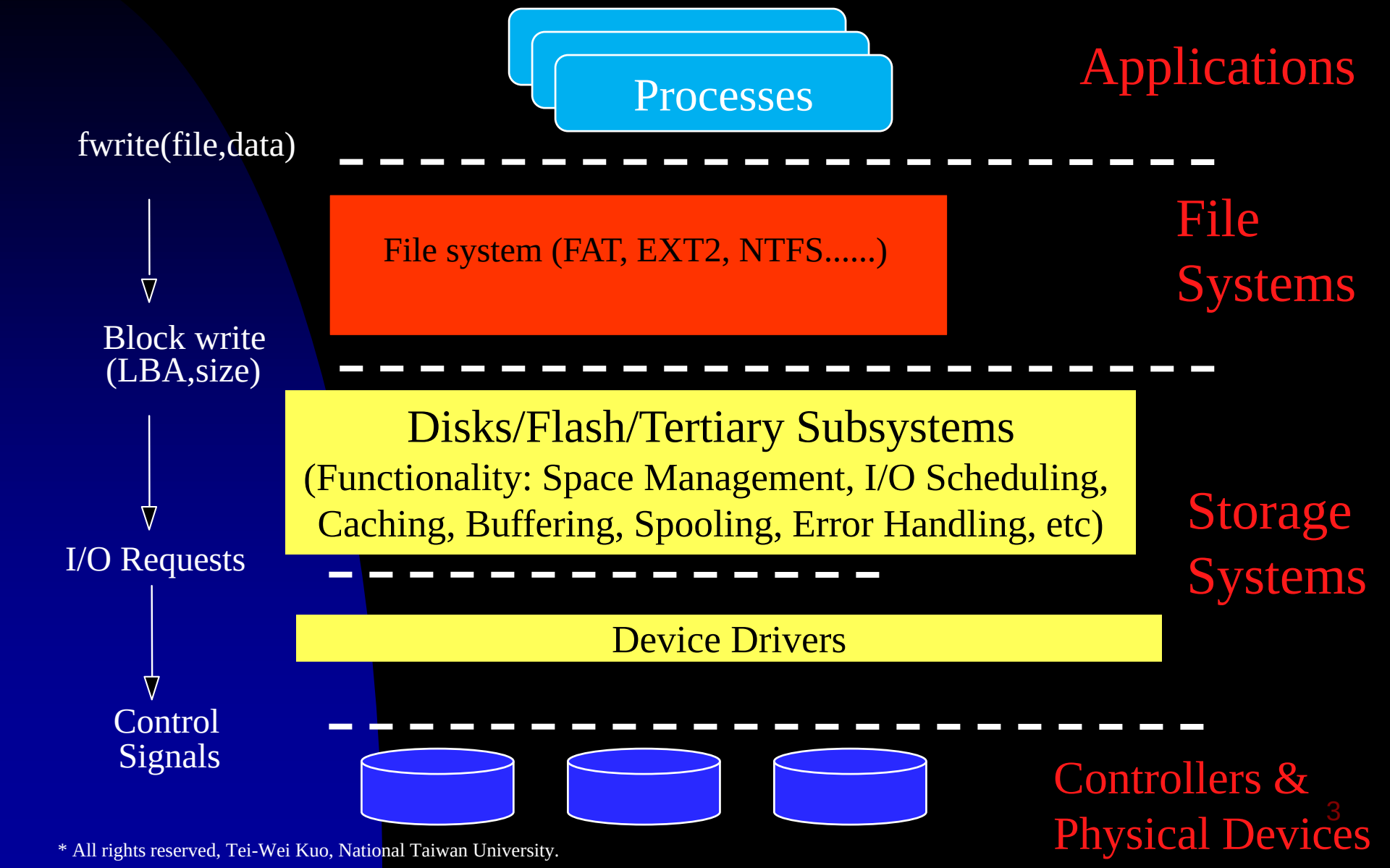
Contents

1. Introduction
2. System Structures
3. Process Concept
4. Multithreaded Programming
5. Process Scheduling
6. Synchronization
7. Deadlocks
8. Memory-Management Strategies
9. Virtual-Memory Management
10. File System pp. 455-485.
11. Mass-Storage Structures
12. I/O Systems
13. Protection, Security, Distributed Systems

Why Storage Management

- Motivations
  - Main memory is too small to accommodate all the data and programs permanently → Secondary Storage
  - A mechanism is needed for on-line storage of and access to both program and data residing on the secondary storage → File System
- Device Variety
  - Speed, Dedication, Read/Write, Char/Block Transfer, Synchronous Mode, etc.

A System Architecture



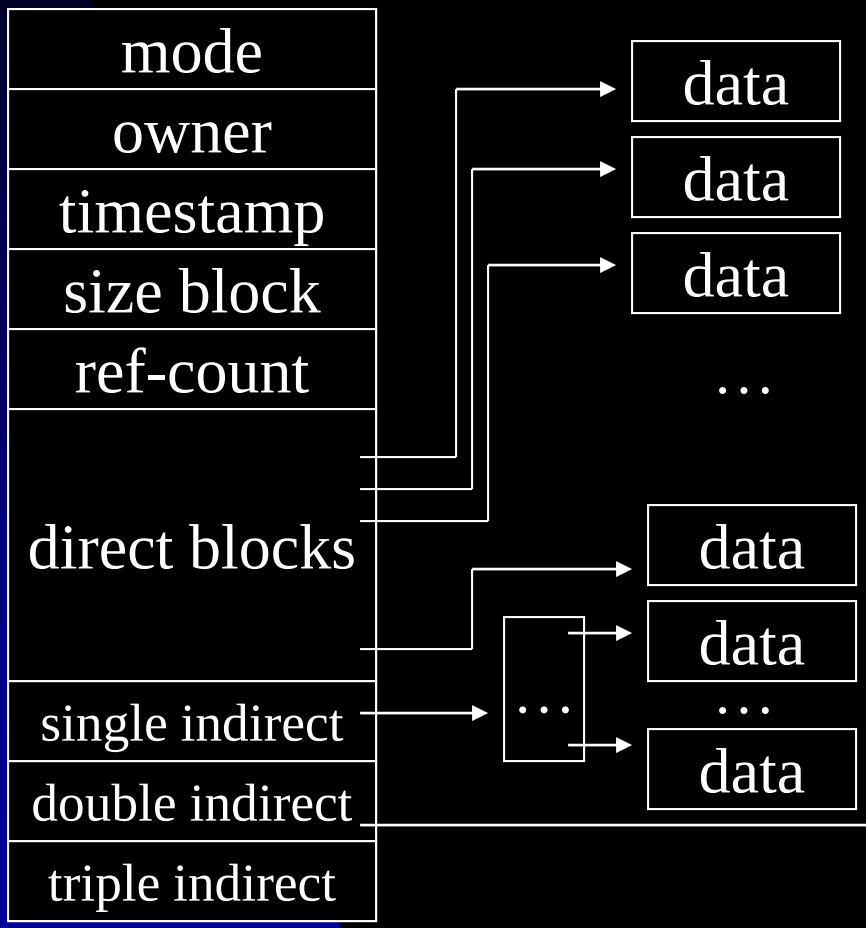
File Concept – A File System

- Files
  - Each is a named collection of related information.
  - Each is a logical unit often with its interpretation left for applications, creators, or users.
    - Text, Source, Object, Executable Files
- A Directory Structure
  - Meta data & File Organization

File Concept – File Attributes

- File attributes vary from one OS to another:
  - Name – Case-Sensitive or Not?
    - The only information must be kept in human-readable form
  - Identifier – A Unique Tag
  - Type
    - It is only for systems that support file types.
  - Location
  - Size – Current and Max Sizes
  - Protection – Access Control
  - Time, Date, and User Identification
- File attributes are usually kept in the directory structure.

File System – 4.4BSD i-node



- 4KB block size
- 12 direct pointers
  - 48KB
- 1 single indirect
  - 4-byte block ptr
  - 1K * 4KB = 4MB
- >> 4GB for the largest file!
  (offset = 32 bits, $4G=2^{32}$)

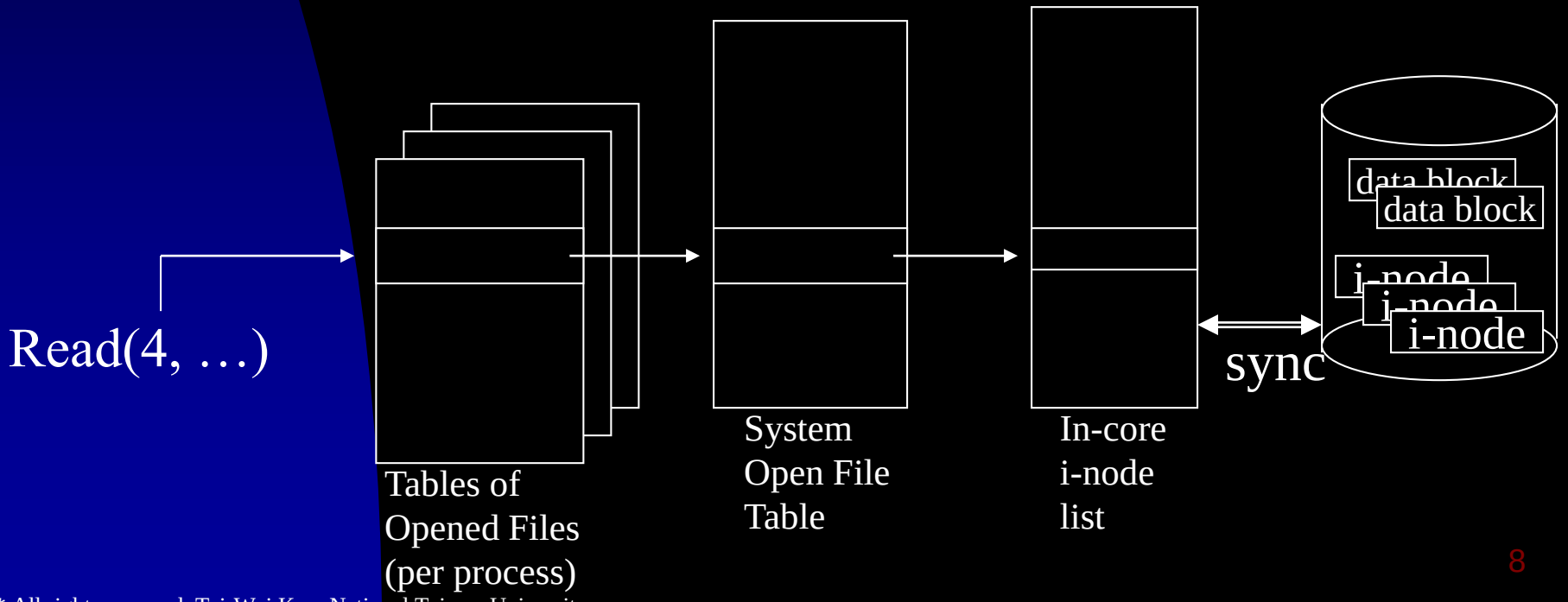
* "Operating system concept", Silberschatz and Galvin, Addison Wesley, pp. 411.

File Concept – File Operations

- Basic Directory Operations:
  - File Creation – Space Allocation & Directory-Structure-Entry Creation
  - File Writing – Write Pointer
  - File Reading – Read Pointer
  - File Reposition – Seek-Like Operations
    - File-Position Pointer
  - File Deletion – Space Reclaiming & Directory-Structure-Entry Deletion
  - File Truncating – File-Length Reseting

File Concept – File Operations

- Open, Close, Read and Write among Multiple Processes
  - File Descriptors and Tables
  - File Position Pointer, File-Open Count
  - Disk Location and Access Rights



File Concept – File Operations

- Extensions
  - File Renaming, Appending, Copying, etc.
- Other Operations
  - Attribute Retrieval and Setting
  - File Locking
    - Shared or Exclusive Locks
    - Mandatory (Windows) or Advisory (Unix) Locks
  - Search of a file
    - A File-System Traversal

File Concept – File Types

- Key Issue
  - The Recognition of File Types by OS?
- Common Technique
  - Types as Parts of File Names
    - .doc, .txt, .rtf, .mpeg, .mp3, .avi, .pdf, .ps, .tex, .exe, .com, .bin, .c, .cc, .java, .asm, .a, .bat, .sh, .o, .obj, .lib, .dll, .zip, .tar, .arc, etc.
    - Links between Files – Makefile
  - A Magic Number at the Beginning of a File
    - Enforcement or Hints? → Application Duty

File Concept – File Structure

- Why File Structure?
  - File Structure versus Program Expectation
  - Extensions by Operating Systems
    - Example: Directories and Special System Calls
- Disadvantages
  - Operating System Size
  - Portability
    - A minimum number of file structures?!
      - Executable Files and Files of a Sequence of 8-Bit Bytes by Unix; A Resource Fork and a Data Fork per File by Mac

File Concept – File Structure

- Implementation
  - Packing of Logical Records into Physical Blocks by OS or Application Programs
  - Internal Fragmentation

Access Methods

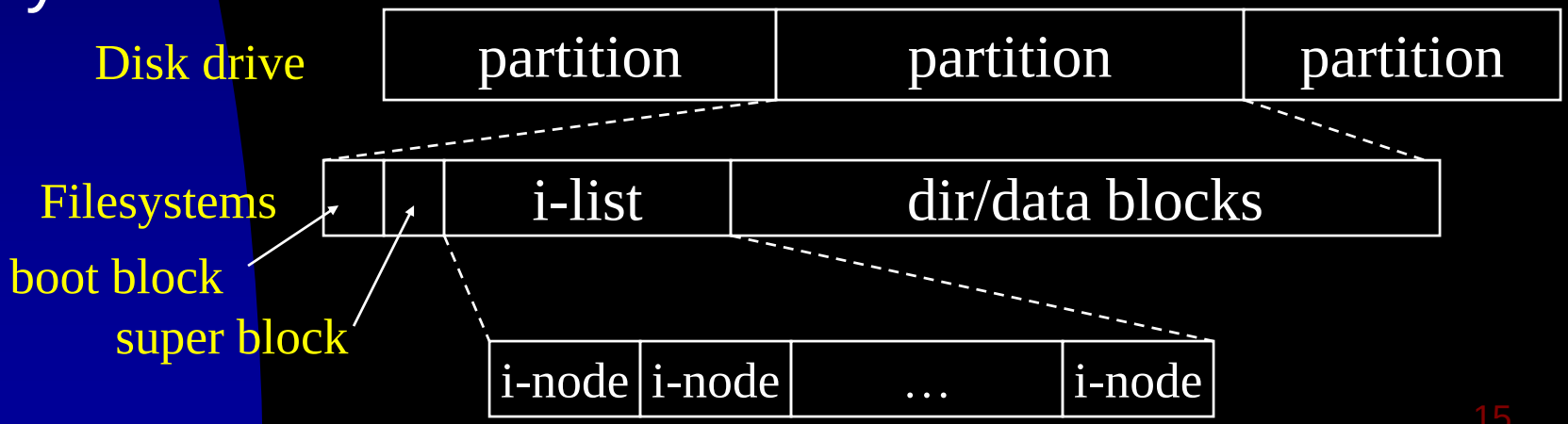
- Sequential Access
  - Read-Next and Write-Next Operations
  - Reset or N-Record Skipping (Forward or Backward) – Rewinding
- Direct Access (/Relative Access)
  - A file is considered as a numbered sequence of blocks or records.
  - Read-N, Write-N, and Position-N Operations
    - Relative Block/Record Number
  - Fixed-Size Records?! Deletion Issues?!
  - Easy Simulation of Sequential Access

Access Methods

- Index-Based Access
  - Index Searching and then Record Access
  - Index Files versus Searching Methods
    - Disk-Resident Index Files?
    - Primary and Secondary Indices
      - Logical Index – K_P in <K, K_P > is used to access records through the primary file organization

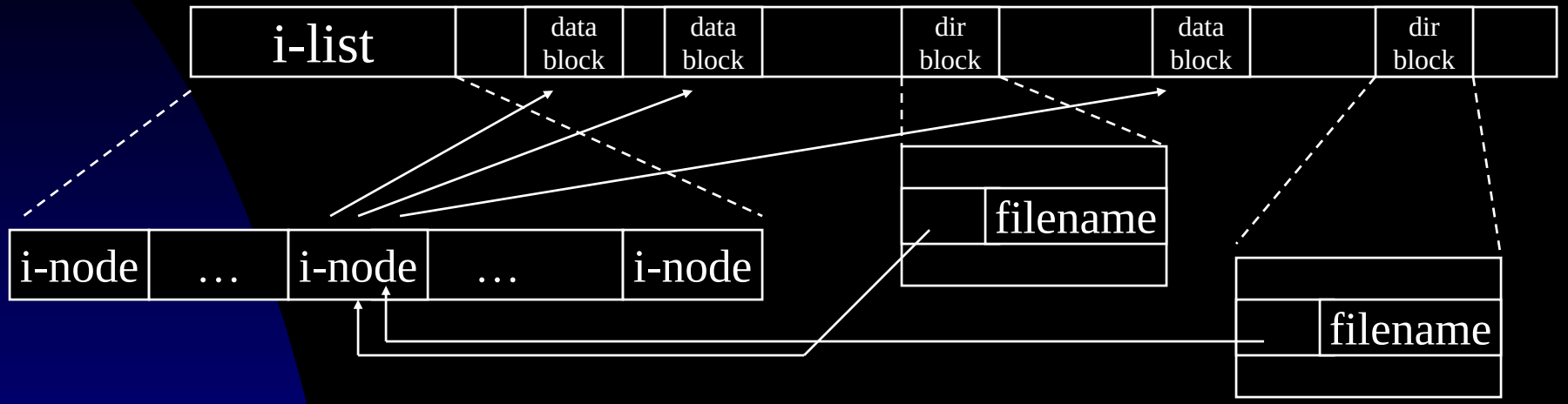
Directory Structure

- A hierarchical arrangement of directories and files – starting at root /
  - File: An abstract data type
  - Volume: A chunk of storage that holds a file system
- Several Types, e.g.,SVR4: Unix System V Filesystems (S5), Unified File System (UFS)
- System V:



* A Device Directory – A Volume Table of Contents

File Systems



- i-node:
  - Version 7: 64B, 4.3+BSD:128B, S5:64B, UFS:128B
  - File type, access permission, file size, data blocks, etc.
- Basic File Operations
  - Create, Write, Read, Reposition, Delete, Truncate.

Directory/Storage Structure

- Example File System Type of the Solaris
  - ufs, zfs – general-purpose file systems
  - tmpfs – a temporary file system created in DRAM
  - objfs – a “virtual” file system as an interface to the kernel for debugging
  - ctfs – a virtual file system that maintain “contract” information to manage which processes start when the system boots and must continue to run during operation

Directory Structure

- Directory Overview
  - Directory – A Symbol Table that Translate File Names into Their Directory Entries.
  - Operations on a Directory
    - Searching for a File
    - Create a File
    - Delete a File
    - List a Directory
    - Rename a File
    - Traverse the File System

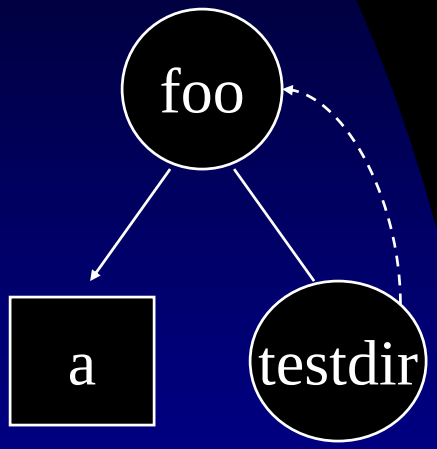
Directories

- Single-Level Directory
  - All files are in the same directory.
    - Problems occur when the number of files increases or when the system has more than one user.
- Two-Level Directory
  - The Master File Directory (MFD) → Multiple User File Directories (UFD's) → Files
    - File Sharing → Path Name and File Duplication of Shared Files (such as Command Files)
      - Volume:[sst.jdeck]login.com;1 & Search Path

Tree-Structured Directories

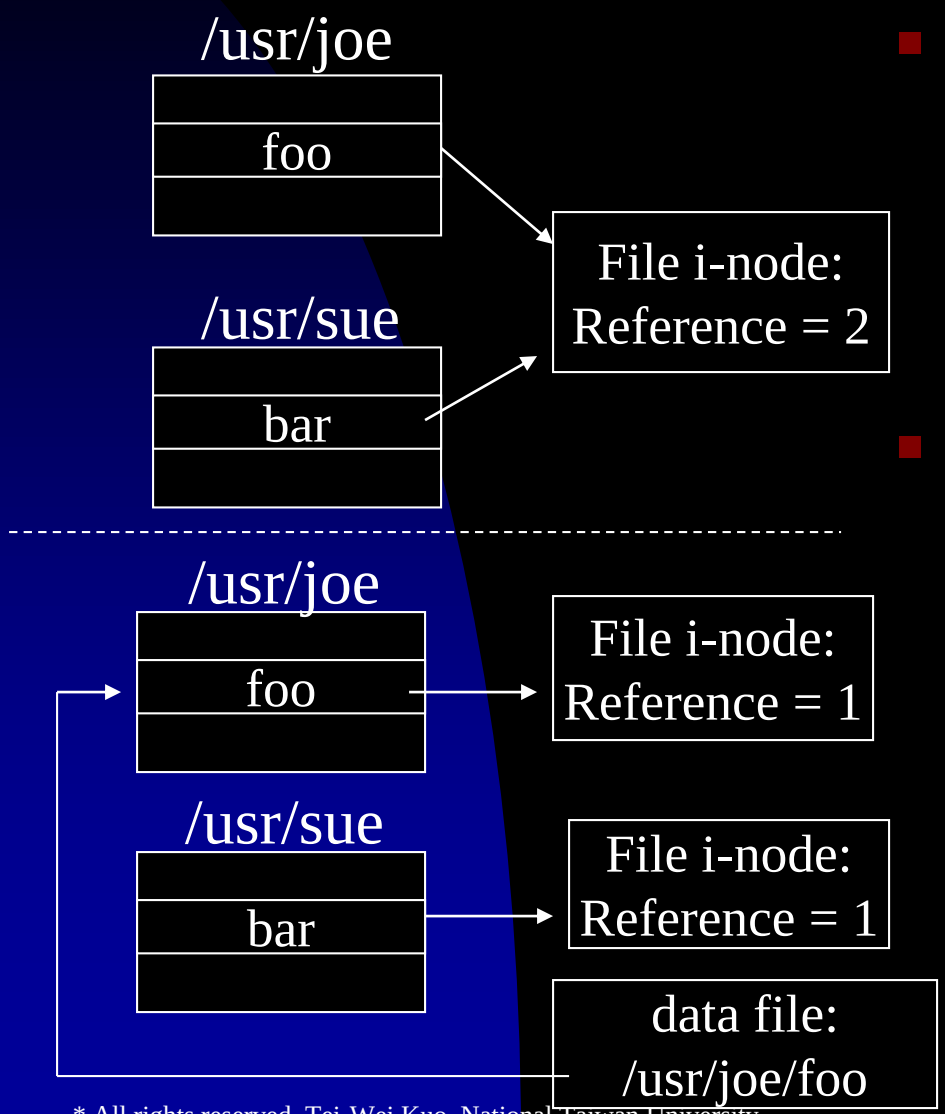
- The Root Directory → Subdirectories and/or files (Example: MS-DOS)
- Current and Home Directories
  - A child process usually inherits the current directory of its parent.
- Absolute and Relative Path Names
  - Examples: /root/spell/mail and spell/mail
  - File Searching: Executable Files or Others?
- Policies
  - Directory Deletion: Only Empty Directories? rm –r xxx

Acyclic-Graph Directories



- Motivation – Allow the Sharing of Files, Compared to Tree-Structured Directories
- File-Sharing Implementations
  - Links – A Pointer to another File or Subdirectory
    - Hard and Soft Links
  - Information Duplication
    - Consistency Issue
  - Potential Problems
    - Multiple Path Names → Traversal and Deletion Problems

Acyclic-Graph Directories



- 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.
  - Filesize = pathname length
  - Example: Shortcut on Windows

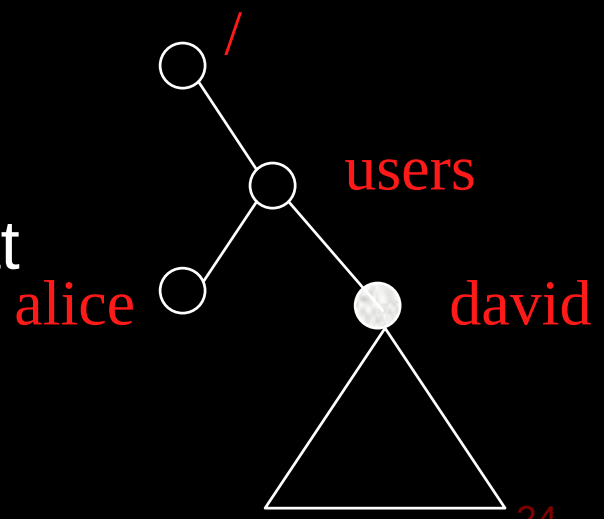
* Problem – infinite loop in tracing a path name with symbolic links – 4.3BSD, no 8 passings of soft links
* Dangling pointers

General Graph Directory

- Potential Problems:
  - Problems in Correctness and Performance in Searching Any Components.
    - Limitation on the Number of Accessed Directories?
  - Problems in File Deletion
    - Self-Referencing or a Cycle?
      - Garbage Collection: Traversing and Marking; Deletion → Extremely Time-Consuming
      - Bypassing Links during Directory Traversal?

File-System Mounting

- A Device and its Mounting Point
  - The Attached Location within the File Structure for a File System
- Features/Constraints:
  - Multiple Mounting per File System
  - Automatic Mounting
    - Device-letter:\path\file
  - Mounting over a Directory that Contains Files



File Sharing

- Issues for Multi-User File Sharing
  - Access Privileges of Shared Files
    - Owner, Group, Other, etc.
    - A subset of allowed operations
- Remote File Systems
  - Manual File Transfers – ftp
    - Anonymous and Authenticated Access
  - Remote Directory Access – Distributed File Systems; Tight Integration
  - A Browser-Based Access – World Wide Web

File Sharing

- The Client-Server Model
  - Servers – Machines that Contains Files
  - Clients – Machines that Seek Access
    - Security versus Compatibility
    - Network File System (NFS) of UNIX lets the server trust the user ID presented by a client.
- Distributed Information Systems/Distributed Naming Services
  - The Domain Name System (DNS) provides host-name-to-network-address translation

File Sharing

- Network Information Service (NIS or yellow pages from SUN) centralizes the storage of user names, host names, printer info, etc.
- Common Internet File System (CIFS from Microsoft) is used with authentication to create a network login for file access.
- Lightweight Directory-Access Protocol (LDAP) provides a secure distributed naming mechanism.
  - Secure Single Sign-On

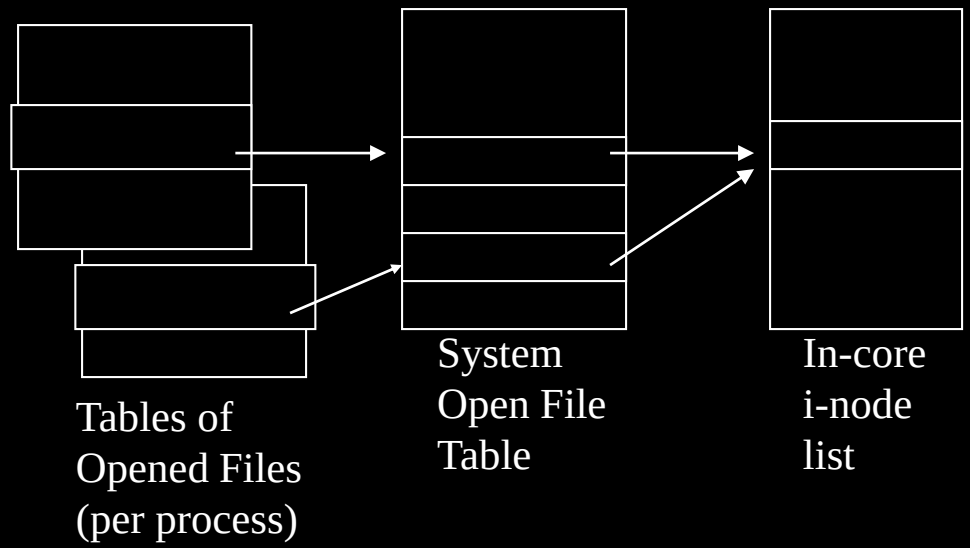
File Sharing

- Failure Modes
  - Disk failure, directory/metadata corruption, disk-controller failure, cable failure, host-adaptor failure, user/systems administrator failure, network failure, etc.
  - Stateless DFS ←→ Security
- Consistency Semantics
  - They specify how multiple users of a system are to access a shared file simultaneously.
  - Their enforcement needs light-weighted implementation of process synchronization.
    - A file session is the series of accesses between open() and close().

File Sharing

- UNIX Semantics
  - Writes to an open file are visible immediately to other users.
  - Some sharing mode allows the sharing of a file pointer.
- Session Semantics – Andrew File System
  - Writes to an open file are not visible immediately to other users.
  - Once a file is close, the changes made to it are visible only to sessions starting later.
- Immutable-Shared-Files Semantics
  - No name reusing and contents altering.

File Sharing – UNIX Semantics



- Each “independently opened file” has its offset.
- Examples
  - Write → offset is incremented!
  - O_APPEND → offset = current file size before each write
  - lseek() causes no I/O (only on the system open file table)
  - dup() and fork causes the sharing of entries in the (system open) file table.
  - filedes flags versus file status flags

Protection

- Motivation:
  - Keep stored information safe from physical damage (reliability) and improper access (protection).
    - Reliability versus Redundancy
    - Protection versus Controlled Access
- Types of Access:
  - Read, Write, Execute, Append, Delete, Listing of File Name and Attributes, etc
  - Higher-Level Functions such as renaming, copying, editing
    - Protection is often offered at a lower level.

Protection – Access Control

- Access Control List (ACL)
  - Access rights per user for each file/directory
  - Advantage:
    - Enable complex access methodologies.
  - Disadvantages:
    - List Length
    - Tedious Construction
    - A Variable Directory Entry

Protection

- A Condensed Version of ACL
  - Owner, Group, and Universe
  - Rights and the Owner/Group Info for Each File
  - Group List for Each User
- A password per file/subdirectory/path
  - Disadvantages:
    - A number of passwords to remember versus one password for a set of files
      - Passwords for Different Operations?

File/Storage System Implementations

- Directory Implementations
  - A Linear List <File Name, Pointers to Data Blocks>
    - Sorted Data Structures, e.g., a B-Tree, and Cache
  - Hash Table
    - Collision Problems
- Storage Systems
  - Space Allocation
    - Contiguous Allocation
    - Linked Allocation
    - Indexed Allocation