Chapter 15: File System Internals

3423 Operating Systems

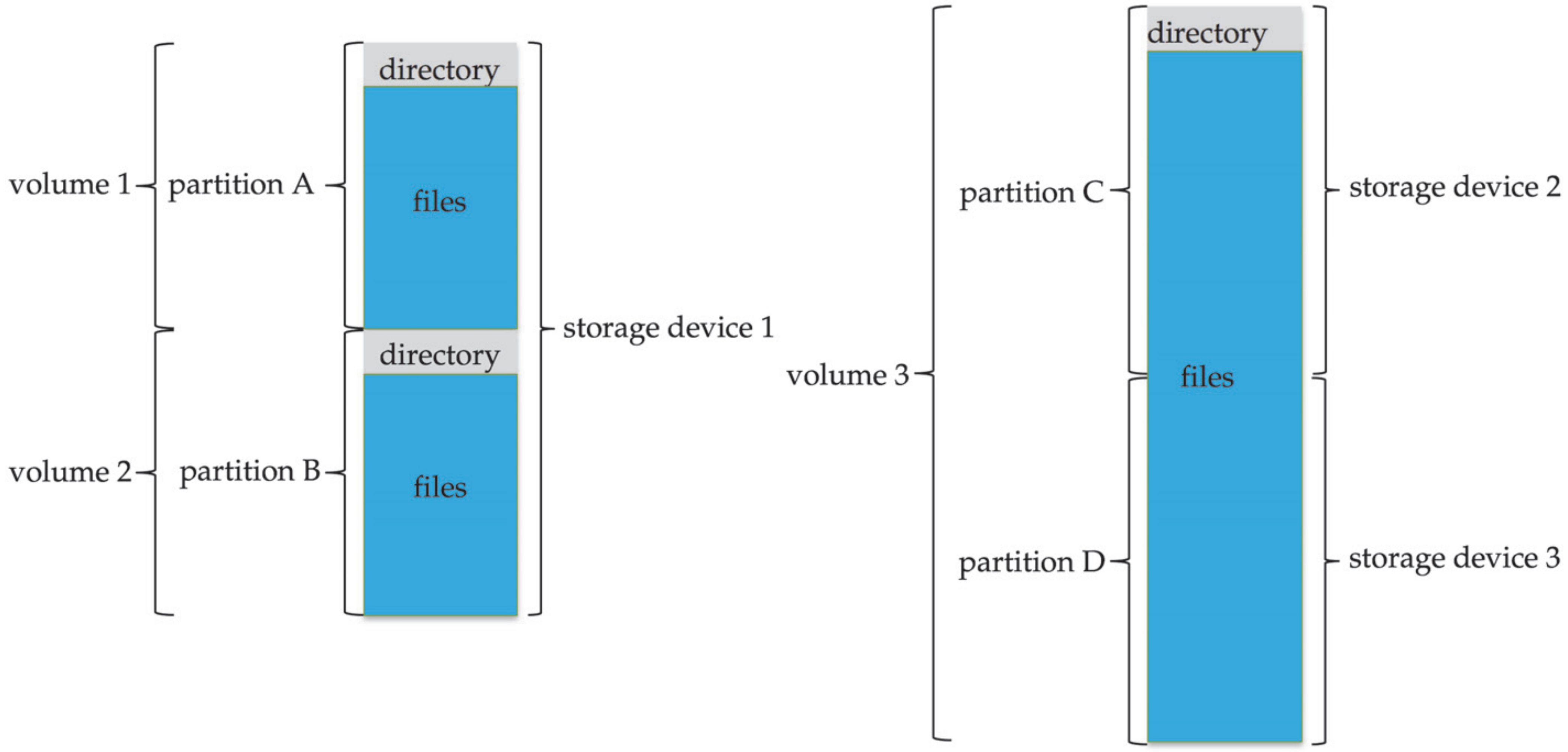
Fall 2019

National Tsing Hua University

Types of File Systems

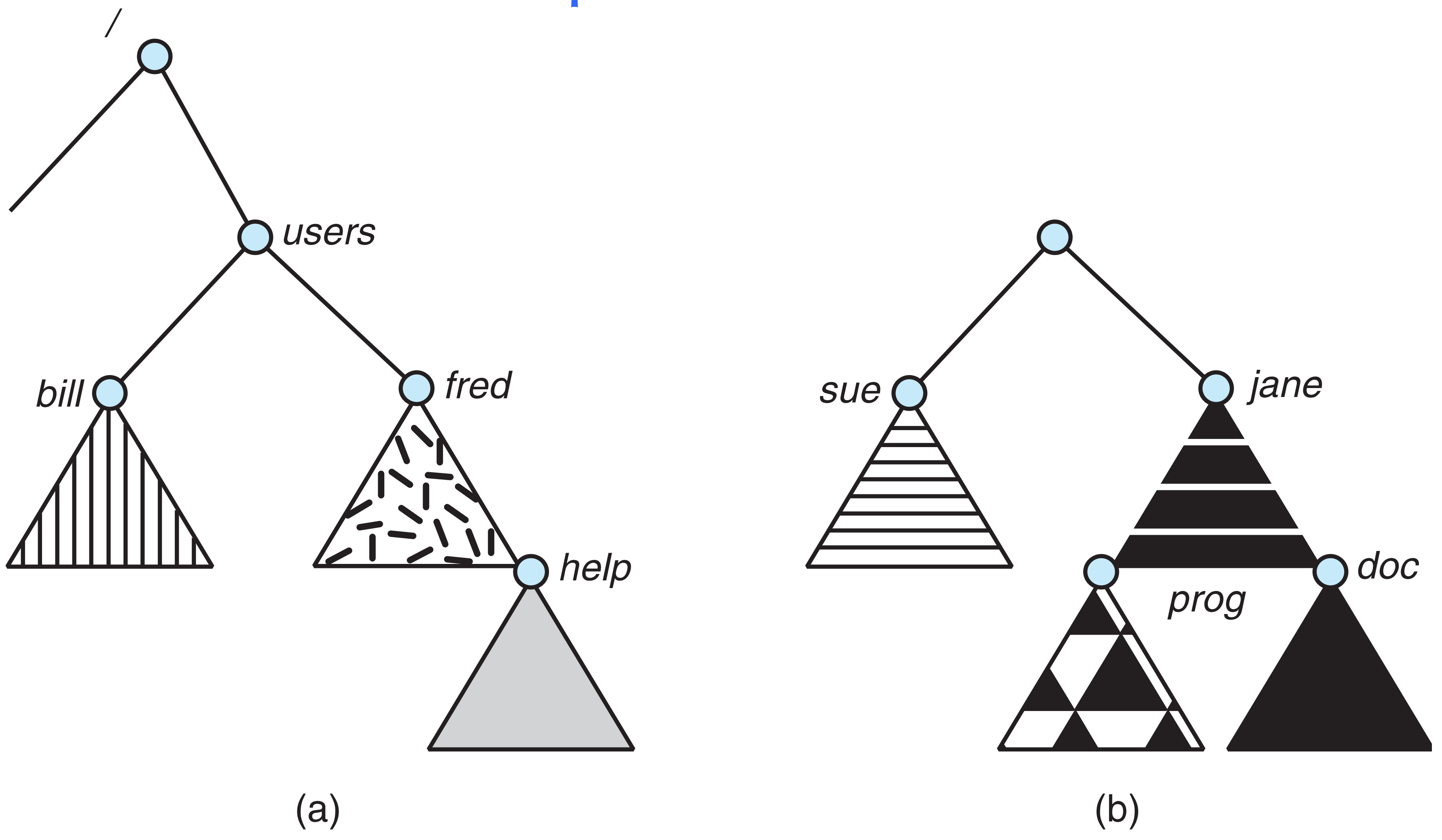
- General-purpose file systems vs Special Purpose
- Example: Solaris
  - `tmpfs` – memory-based volatile FS for fast, temporary I/O
  - `objfs` – interface into kernel memory to get kernel symbols for debugging
  - `ctfs` – "contract" file system for managing daemons - i.e., processes that are started on boot up and continue running
  - `lofs` – loopback file system allows one FS to be accessed in place of another
  - `procfs` – kernel interface to process structures as a file system
  - `ufs, zfs` – general purpose file systems

A Typical File-system Organization

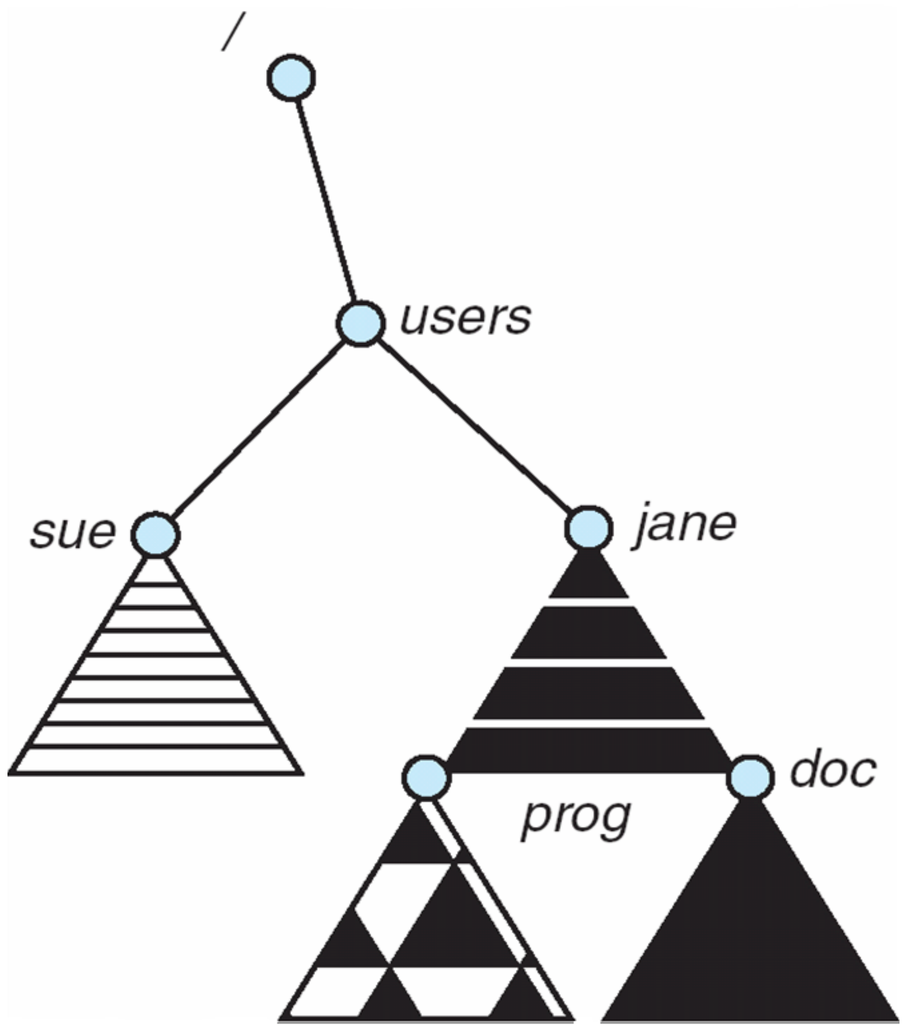


File System Mounting

- 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



Mount Point



Partitions

- Partition can be
  - volume containing a file system ("cooked") or
  - raw – just a sequence of blocks with no file system
- Root partition
  - contains the OS
  - Mounted at boot time
- other partitions
  - can hold other OSs, other file systems, or raw
  - can mount automatically or manually

Mounting

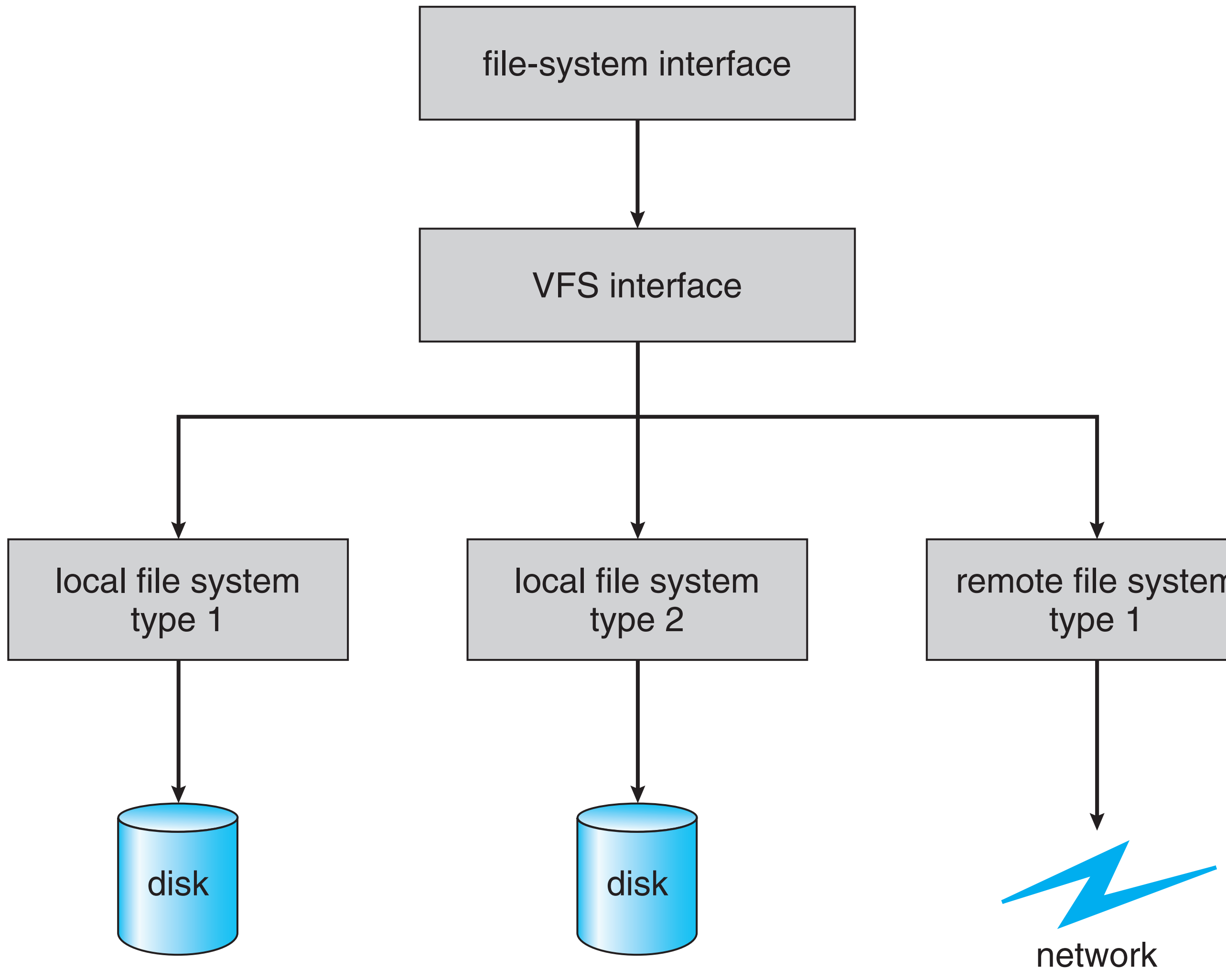
- Boot block can
  - point to boot volume or
  - contain *boot loader* = set of blocks that contain enough code to know how to load the kernel from the file system
  - Or a boot management program for multi-OS booting
- At mount time, file system consistency checked
  - if not all metadata correct, fix it, try again
  - If all correct, add to mount table, allow access

Virtual File Systems

- Virtual File Systems (VFS) on Unix
  - provide an object-oriented way of implementing file systems
- Same system call API for different types of file systems
  - Separates file-system generic operations from implementation details
  - Implementation can be a file system or network file system
    - Implements vnodes which hold either inodes (for a local file system) or network file details (for network file system)
  - Then dispatches operation to appropriate file system implementation routines

Virtual File Systems (Cont.)

- The API is to the VFS interface, rather than any specific type of file system



For object types in Linux VFS

- inode object:
  - an individual file
- file object:
  - an open file
- superblock:
  - entire file system
- dentry:
  - "directory entry"

File Sharing on Distributed Systems

- Several approaches
  - Manually via FTP, anonymous or authenticated
  - Automatically using distributed file systems
  - Via the world wide web, anonymous
- If distributed systems, shared across a network
  - Network File System (NFS) is a common distributed file-sharing method
  - trickier to authenticate across machines! Same user may have different user IDs on different machines. how to trust them?

File Sharing – Remote File Systems

- Client-server model
  - allows clients to mount remote file systems from servers
  - Client and user-on-client identification is insecure or complicated
  - Standard operating system file calls are translated into remote calls
- NFS (network file system) on UNIX
  - server must trust client; user IDs must match
  - uses NIS (network info. service) to authenticate. NIS+ secure.
- CIFS (common Internet file system) on Windows
  - creates network login
  - Active Directory for distributed naming structure

Distributed Naming Service

- Examples
  - DNS, NIS, Active Directory
  - implement unified access to information needed for remote computing
- Trend: moving towards LDAP
  - LDAP = lightweight directory-access protocol
  - single sign-on for an organization
  - Active Directory is based on LDAP

File Sharing – Failure Modes

- All file systems have failure modes
- Local file system
  - corruption of directory structures or metadata
- Remote file systems
  - new failure modes, due to network failure, server failure
- Recovery from failure
  - involve state information about status of each remote request
  - Stateless protocols such as NFS v3 include all information in each request, allowing <u>easy recovery</u>, but <u>less security</u>

File Sharing – Consistency Semantics

- Specify simultaneous access to file by multiple users
  - Similar to process synchronization algorithms
- Unix file system (UFS) implements:
  - Writes to an open file visible immediately to other users
  - Sharing file pointer to allow multiple users to read and write concurrently
- Andrew File System (AFS)
  - implemented complex remote file sharing semantics
  - session semantics: writes only visible to sessions after the file is closed
  - advantage: local access speed; but multiple versions exist!

Network File System (NFS) by Sun Microsystems

- NFS
  - an implementation
  - a specification of a software system
- Purpose
  - for accessing remote files across LANs (or WANs)
- The implementation
  - part of the Solaris and SunOS operating systems running on Sun workstations
  - uses unreliable datagram (UDP/IP protocol) and Ethernet

NFS Architecture

- Interconnected workstations
  - a set of independent machines with independent file systems
  - allows sharing among these file systems in a transparent manner
- Remote directory is mounted over a local FS directory
  - mounted directory looks like an integral subtree of the local FS
  - mount operation needs the host name of the remote directory
  - Subject to access-rights accreditation,
  - potentially any FS (or directory within a FS), can be mounted remotely on top of any local directory

NFS (Cont.)

- Heterogeneous environment
  - independent of machines, OSs, and network architectures
- mechanism: RPC primitives
  - built on top of an External Data Representation (XDR) protocol used between two implementation-independent interfaces
- NFS specification distinguishes between
  - the services provided by a mount mechanism and
  - the actual remote-file-access services