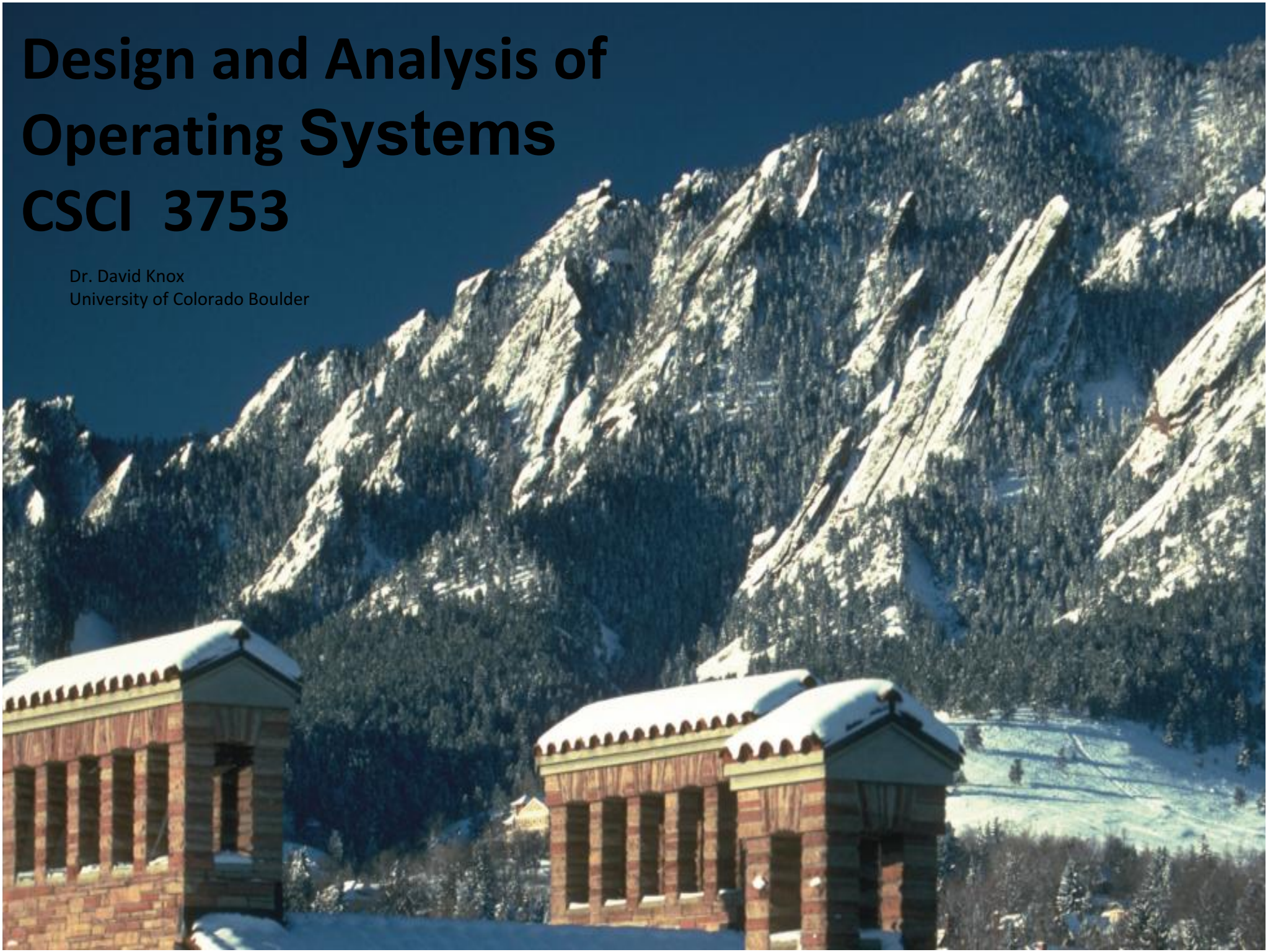
Design and Analysis of Operating Systems CSCI 3753

Dr. David Knox
University of Colorado Boulder



Department of Computer Science
UNIVERSITY OF COLORADO **BOULDER**

Design and Analysis of Operating Systems CSCI 3753

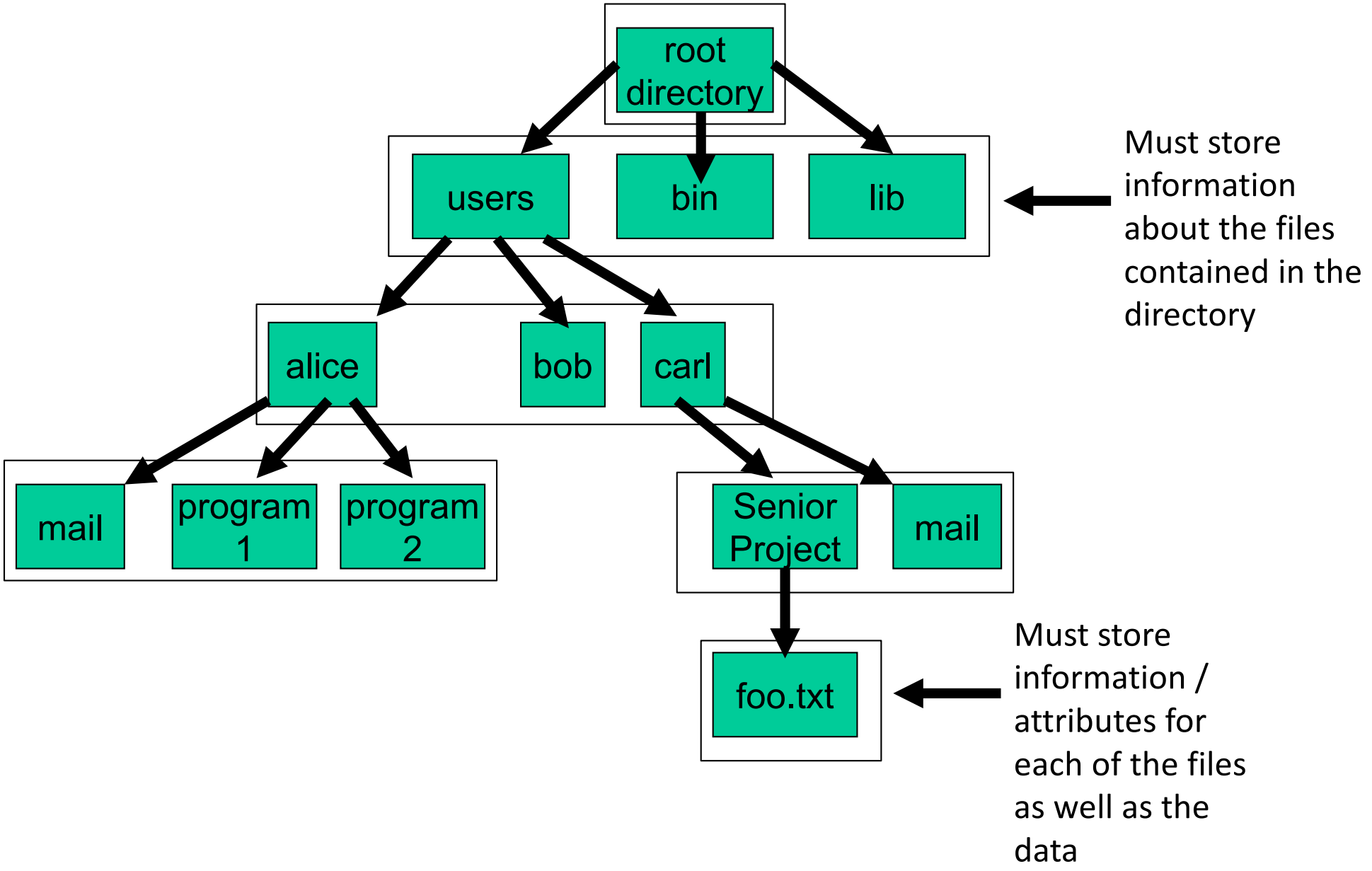
File System Implementation

Dr. David Knox
University of
Colorado Boulder

Material adapted from: Operating Systems: A Modern Perspective : Copyright © 2004 Pearson Education, Inc.

File System Implementation

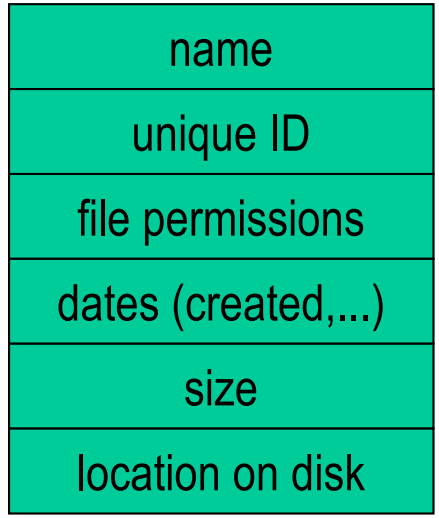
Tree-structured Directory Structure



File System Implementation

- **File system elements are stored on *both*:**
  - Disk/flash – persistent storage
  - Main memory/RAM – volatile storage
- **On *disk/flash*, the entire file system is stored, including 5 main elements:**
  1. its entire directory tree structure
  2. each file's attributes are in a File Control Block
  3. each file's data
  4. a *boot block*, typically the first block of a volume, that contains info needed to boot an operating system from this volume. Empty if no OS to boot.
  5. a *volume control block* that contains volume or partition details, e.g. tracks free blocks on disk, the number of blocks in a partition, size of a block, etc.

example FCB



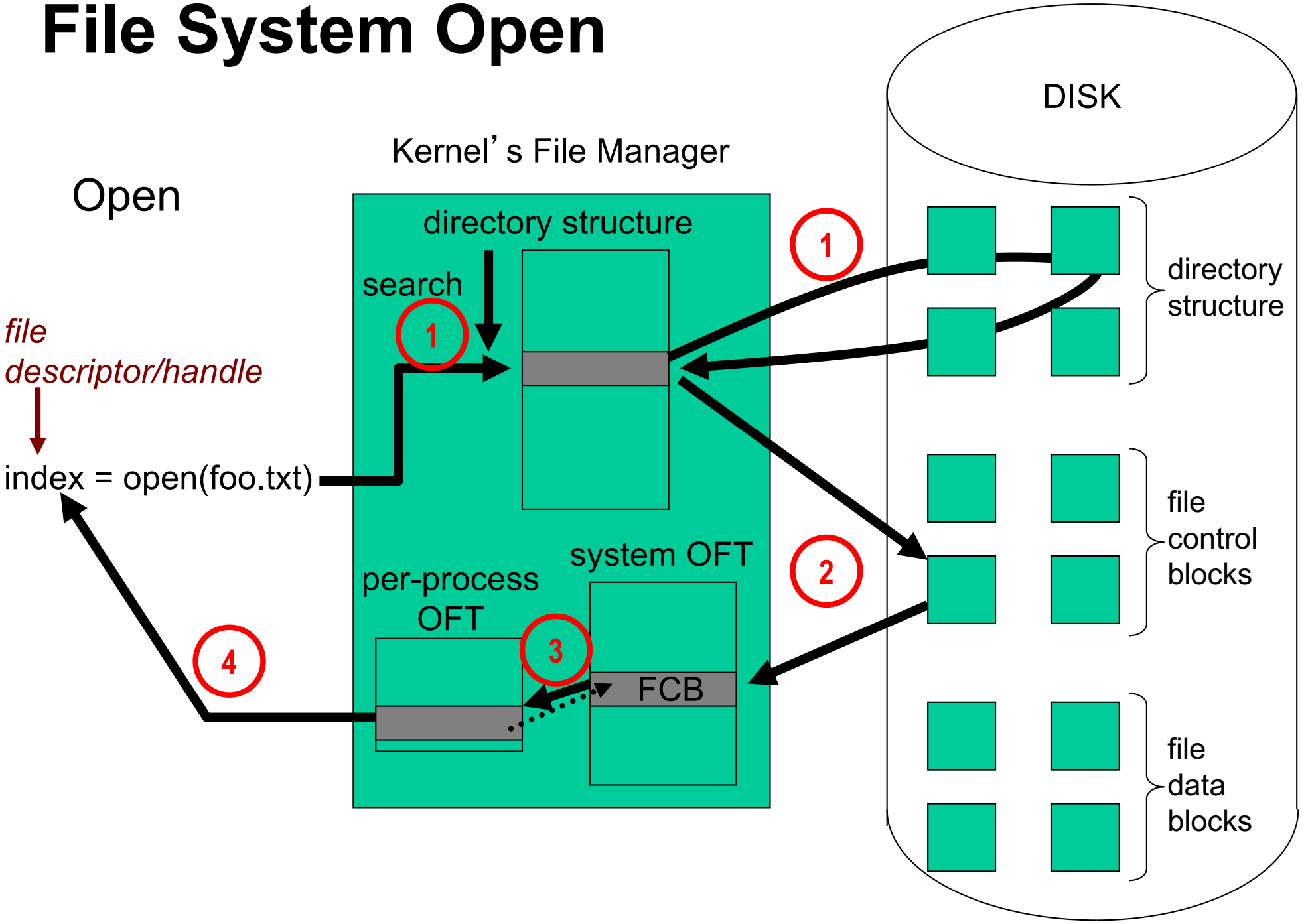
File System Implementation

- **In memory/RAM, the OS file manager maintains only a subset of open files and recently accessed directories**
  - memory is used as a cache to improve performance.
  - All the information is available for a fast search of memory, rather than a slow search of disk, e.g. for a file's FCB.
- **The four main file system components in memory are:**
  1. recently accessed parts of the directory structure tree are stored in memory

The four main file system components in memory are:

1. Recently accessed parts of the directory structure tree are stored in memory
2. OS also maintains a *system-wide open file table* (let's abbreviate it OFT) that tracks process-independent info of open files
   - the file header containing attributes about the open file is stored here
   - an open count of the number of processes that have a file open is stored here
3. OS also maintains a *per-process OFT* - tracks all files that have been opened by a particular process, may store access rights, etc.
   - also keeps a current-file-position pointer, i.e. where in the file the process is currently reading/writing
4. OS keeps a mount table of devices with file systems that have been mounted as volumes
   - we'll use the terms "volumes" and "partitions" interchangeably, though technically a volume may be spread across different disk partitions on different disks, e.g. in RAID disk systems

File System Open



File System Open

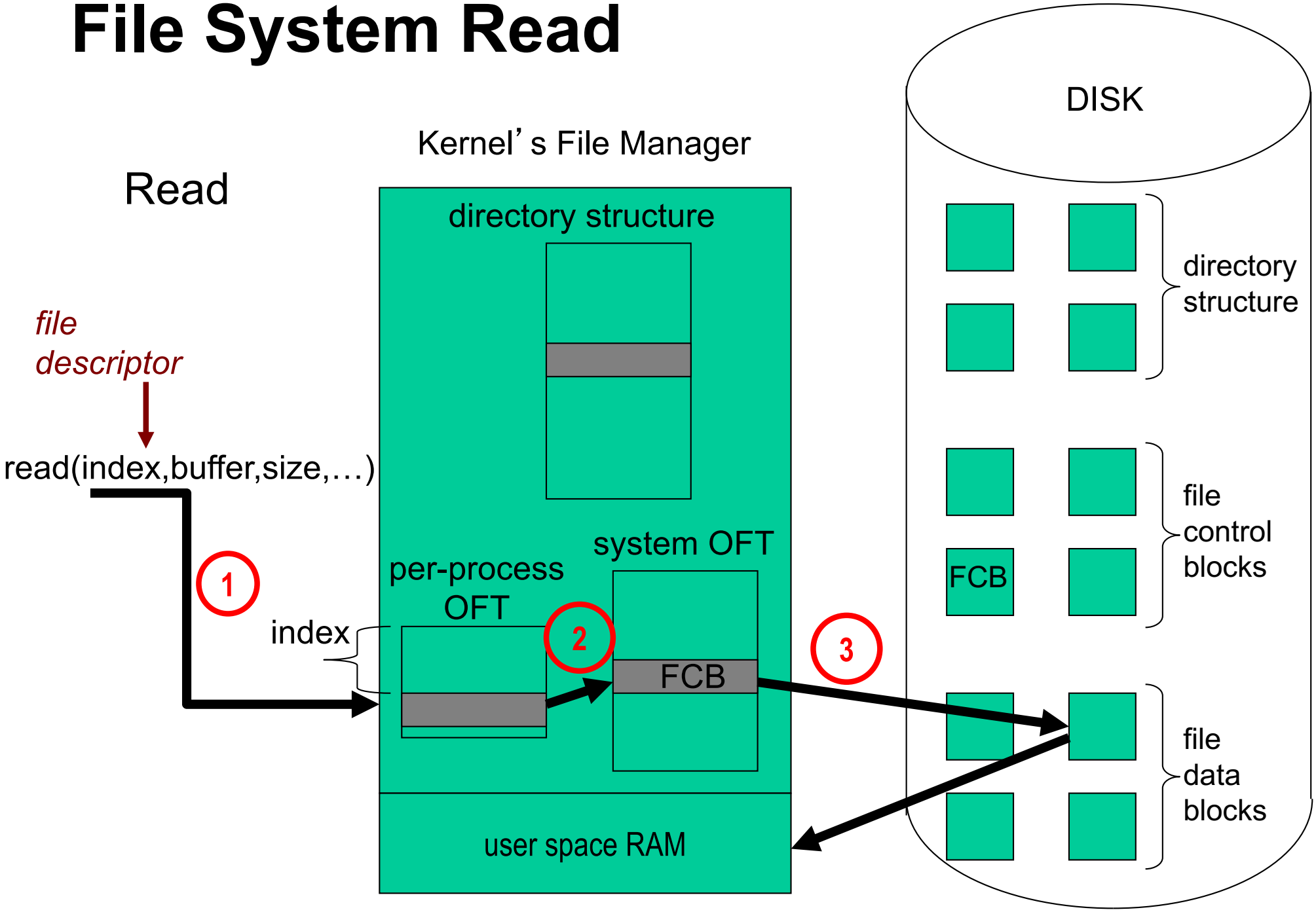
- **When a process calls *open(foo.txt)*, the following procedural steps are followed:**
  1. Directory structure is searched for the file name foo.txt
     - if the directory entries are in memory, then the search is fast
     - otherwise, directories and directory entries have to be retrieved from disk and cached for later accesses
  2. Once the file name is found, the directory entry contains a pointer to the FCB on disk
     - retrieve the FCB from disk
     - copy the FCB into the system OFT. This acts as a cache for future file opens.
     - Increment the open file counter for this file in the system OFT
  3. Add an entry to the per-process OFT that points to the file's FCB in the system OFT
  4. Return a file descriptor or handle to the process that called open()

File System Open

- **Some OS's employ a mandatory lock on an open file**
  - only one process at a time can use an open file
  - Windows policy

- **Other OS's allow optional or advisory locks**
  - UNIX policy
  - It's up to users to synchronize access to files

File System Read



File System Close

- **on a close() system call**
  1. remove the entry from the per-process OFT
  2. decrement the open file counter for this file in the system OFT
  3. if counter = 0, then write back to disk any metadata changes to the FCB, e.g. its modification date
     - Note: there may be a temporary inconsistency between the FCB stored in memory and the FCB on disk
     - Designers of file systems need to be aware of this.
     - A similar inconsistency occurred for modified memory-mapped file data in RAM that had not yet been written to disk.

Department of Computer Science
UNIVERSITY OF COLORADO **BOULDER**

Design and Analysis of Operating Systems CSCI 3753

Dr. David Knox
University of
Colorado Boulder

Material adapted from: Operating Systems: A Modern Perspective : Copyright © 2004 Pearson Education, Inc.