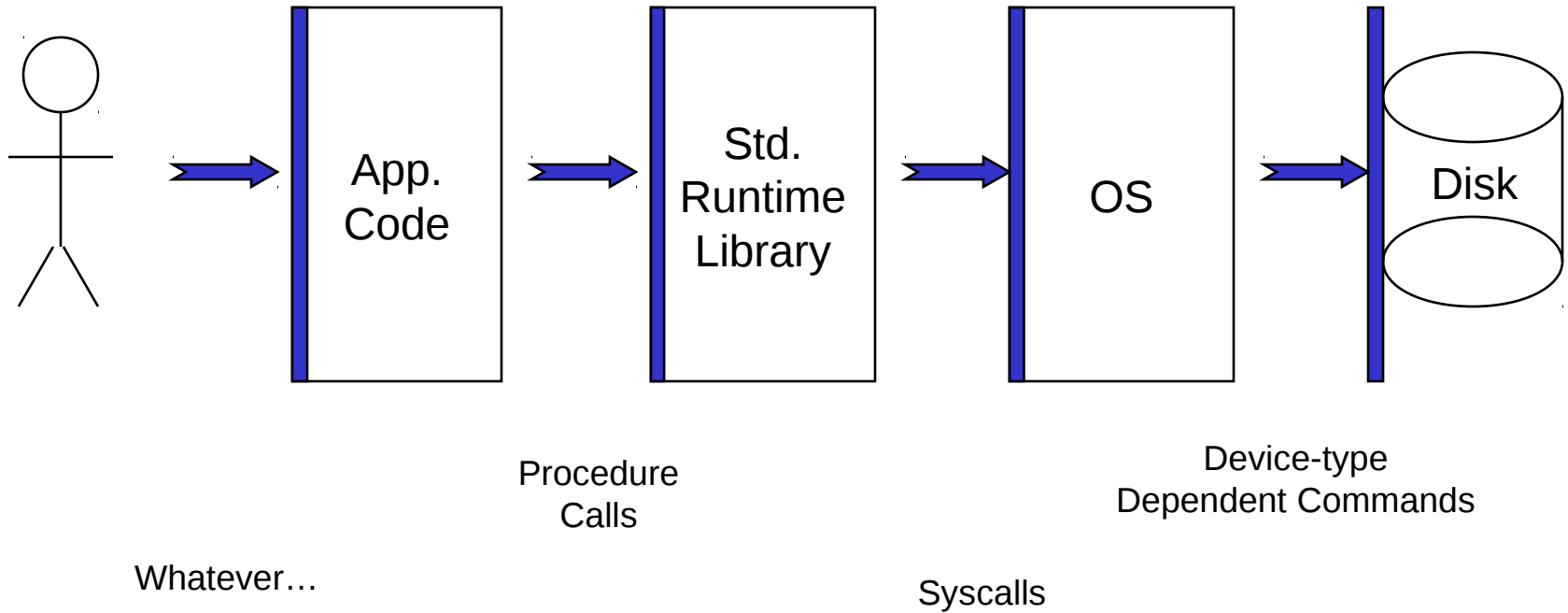


CSE 451: Operating Systems Spring 2011

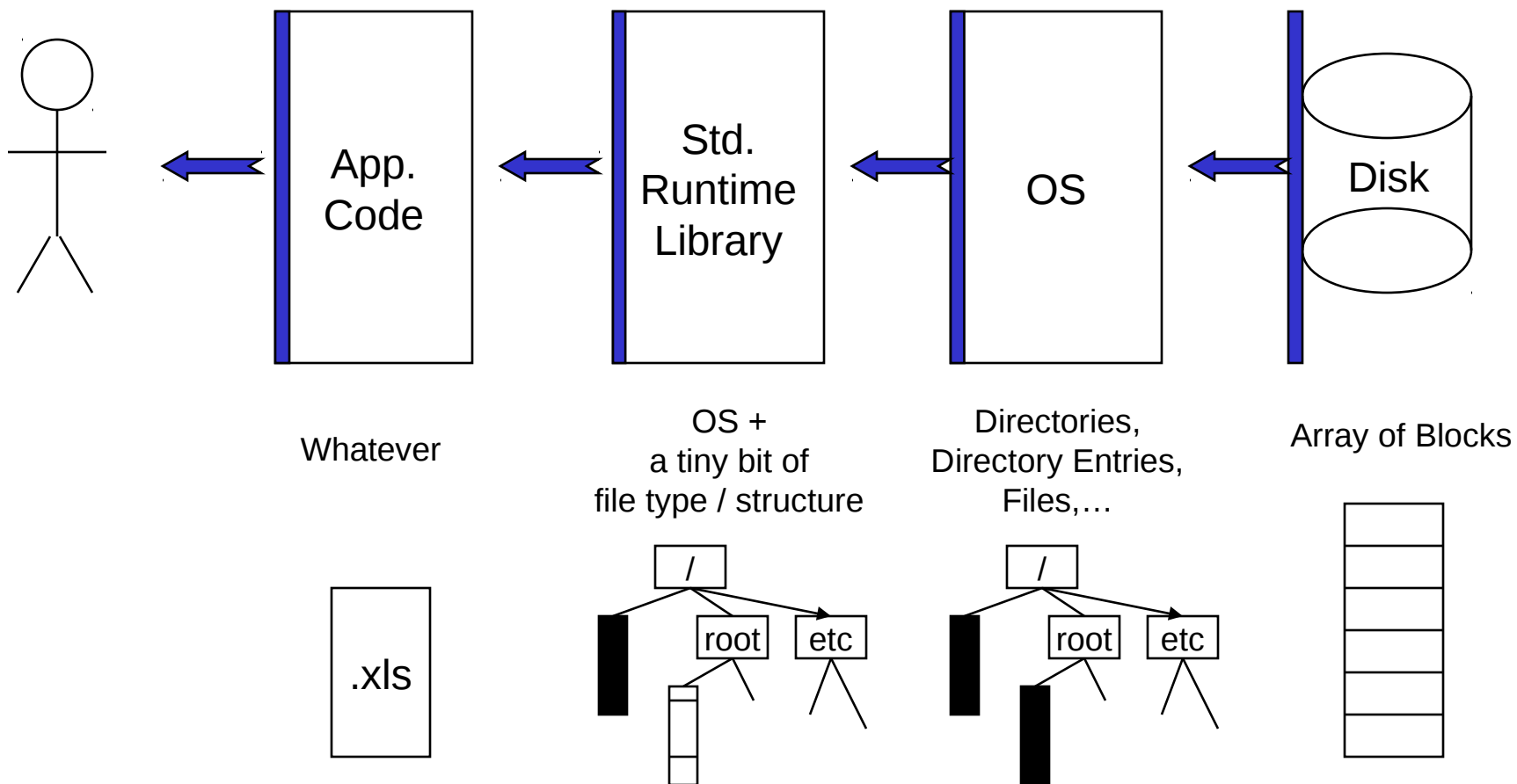
Module 12 Secondary Storage Overview

**John Zahorjan
zahorjan@cs.washington.edu
Allen Center 534**

Interface Layers

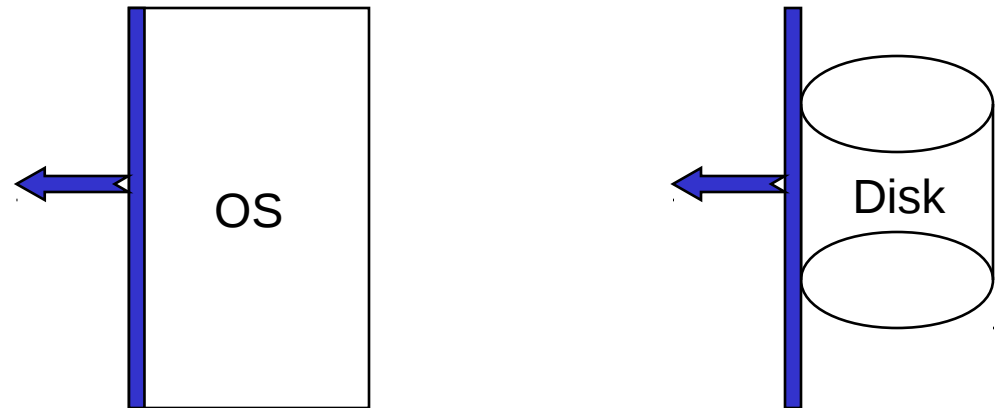


Exported Abstractions



Primary Roles of the OS (file system)

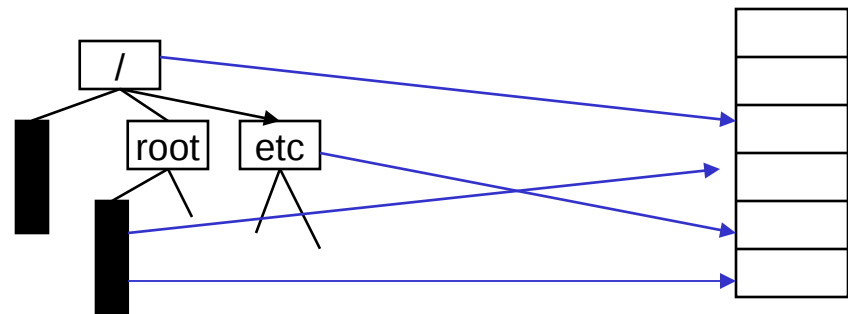
1. Hide hardware specific interface
2. Allocate disk blocks
3. Check permissions
4. Understand directory file structure
5. Maintain *metadata*
6. Performance
7. Flexibility



Why does the OS define directories?

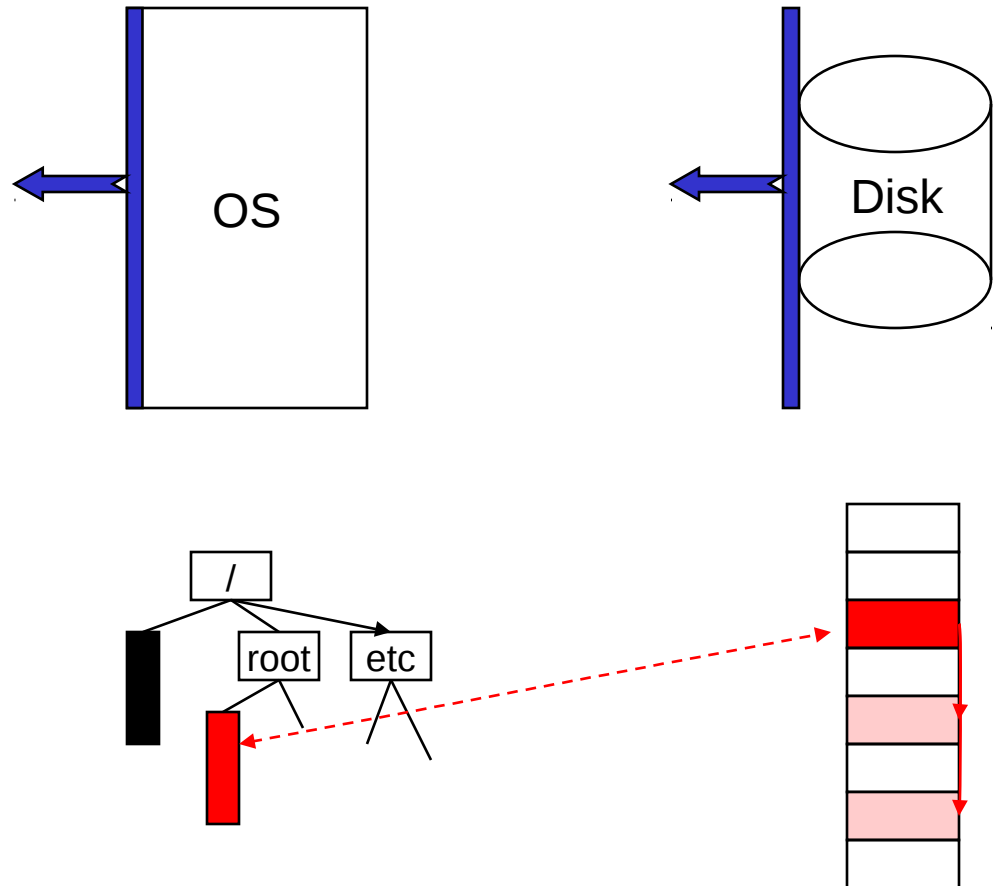
Why not leave that to the library/application layer?

(Why would you want to leave it to the app/library?)

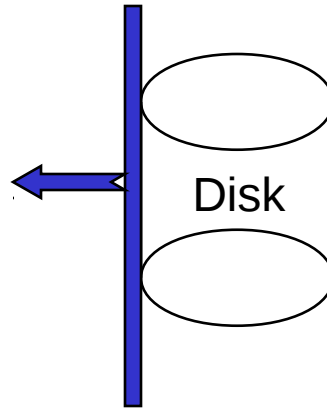
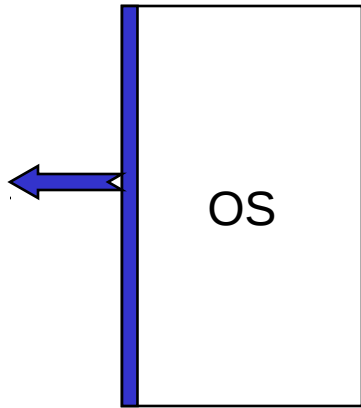


What Is A File?

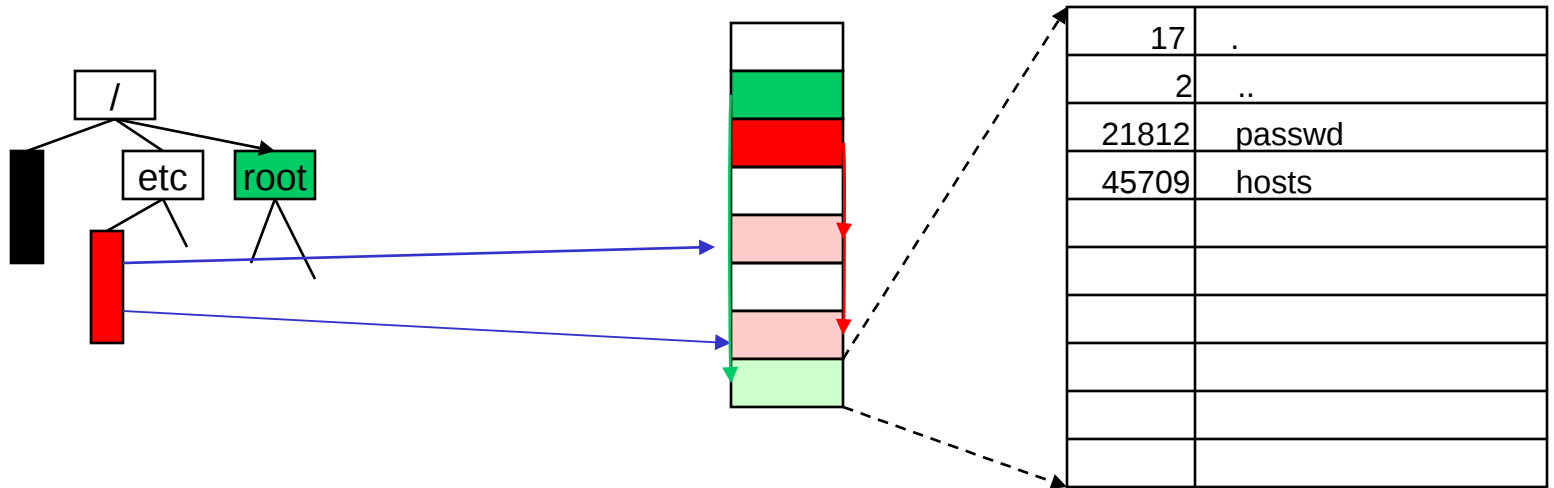
- At this level, what we think of as a file is an “inode”
- An inode keeps track of where the data blocks are
- It also keeps track of metadata (e.g., permissions, last modification time, etc.)



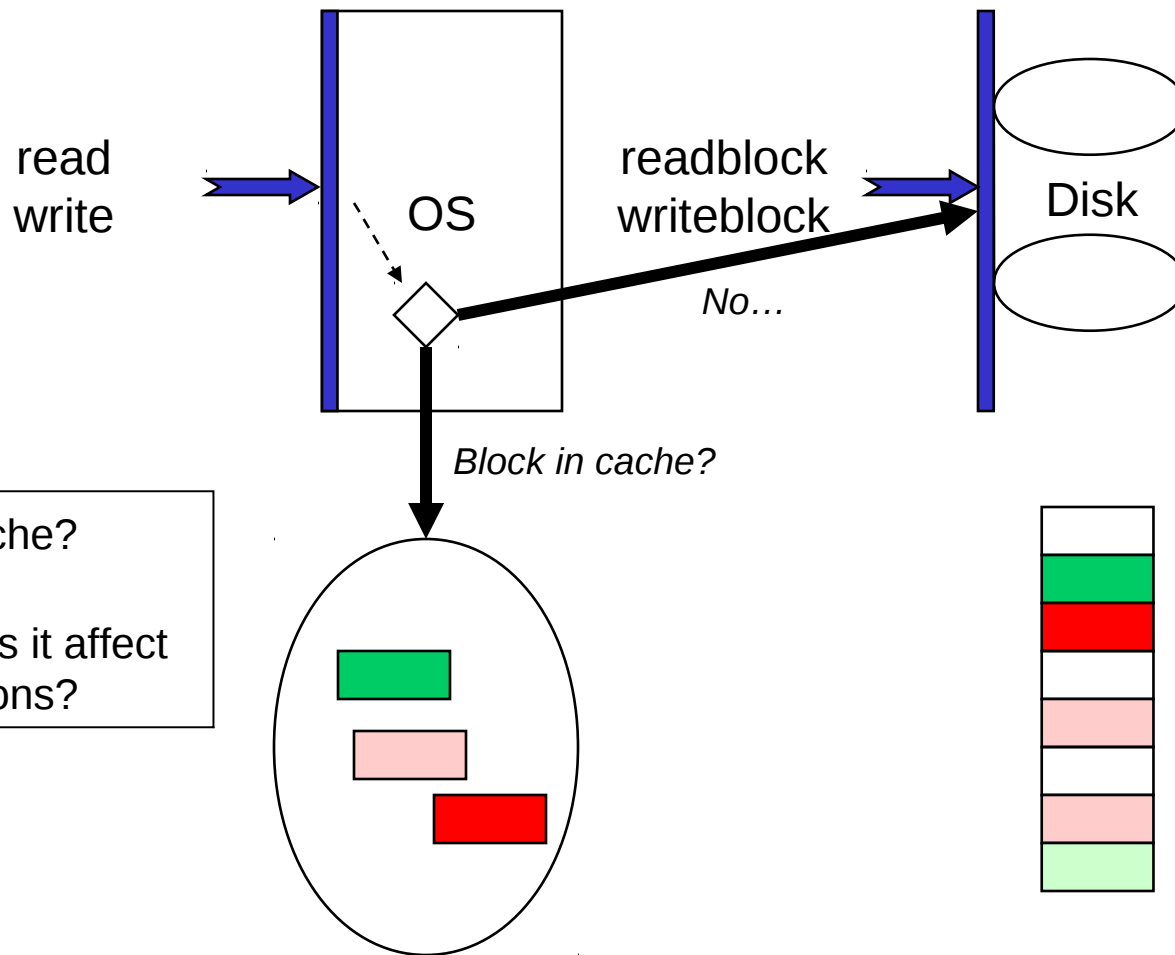
What Is A Directory?



Ok, so what is a "hard link"?

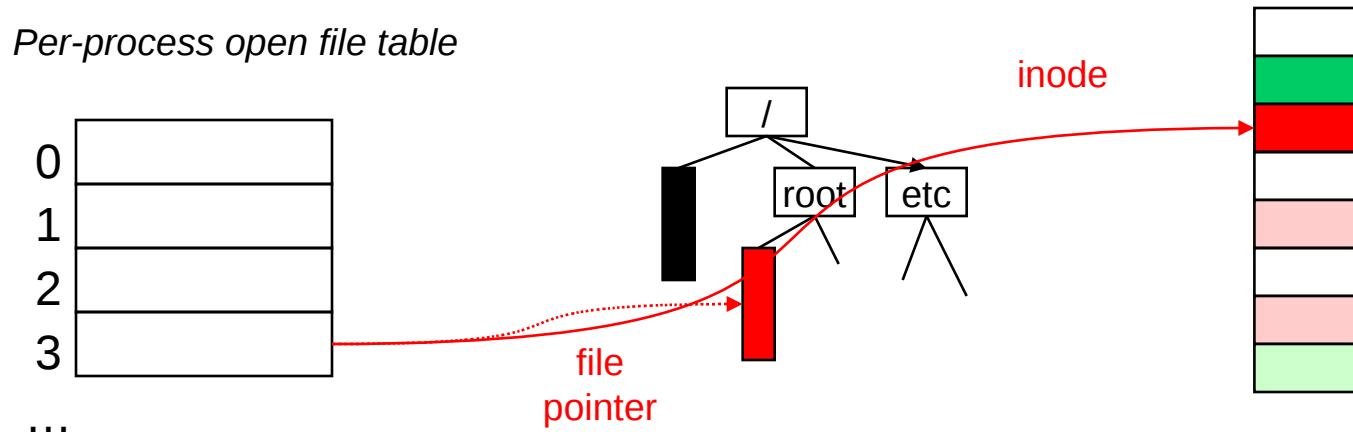
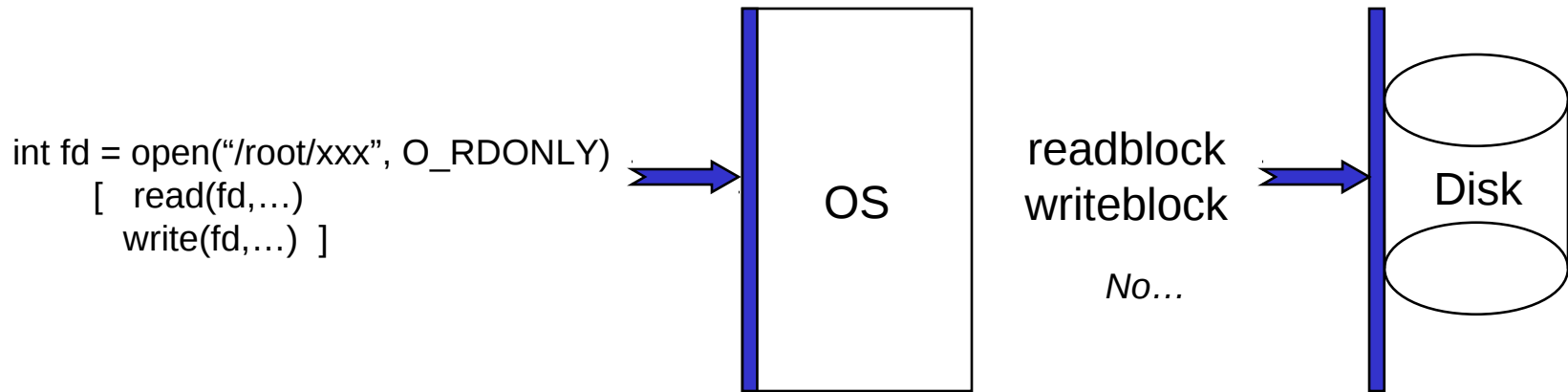


Obtaining Performance: Caching

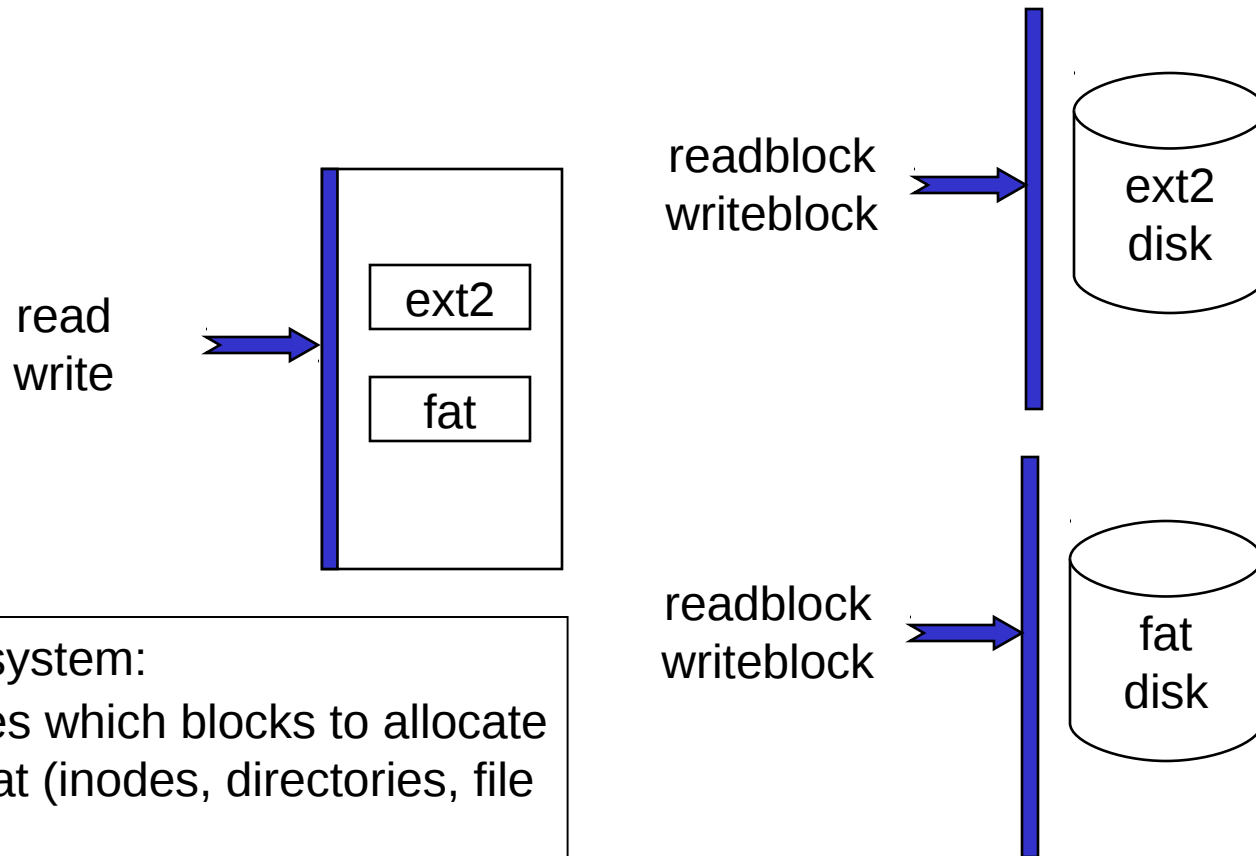


- Why cache?
- How does it affect applications?

Obtaining Performance: open



Multiple File Systems

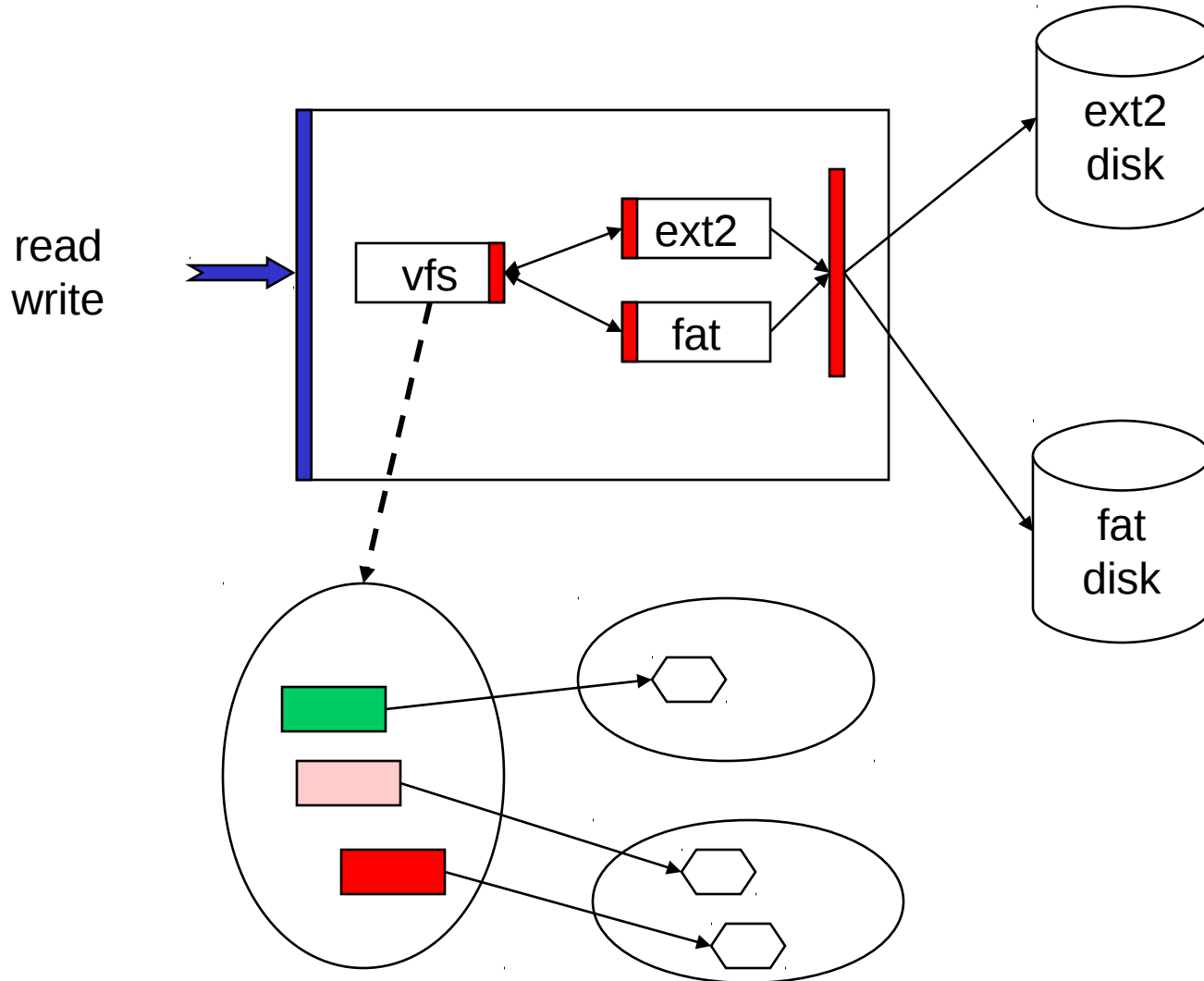


Each filesystem:

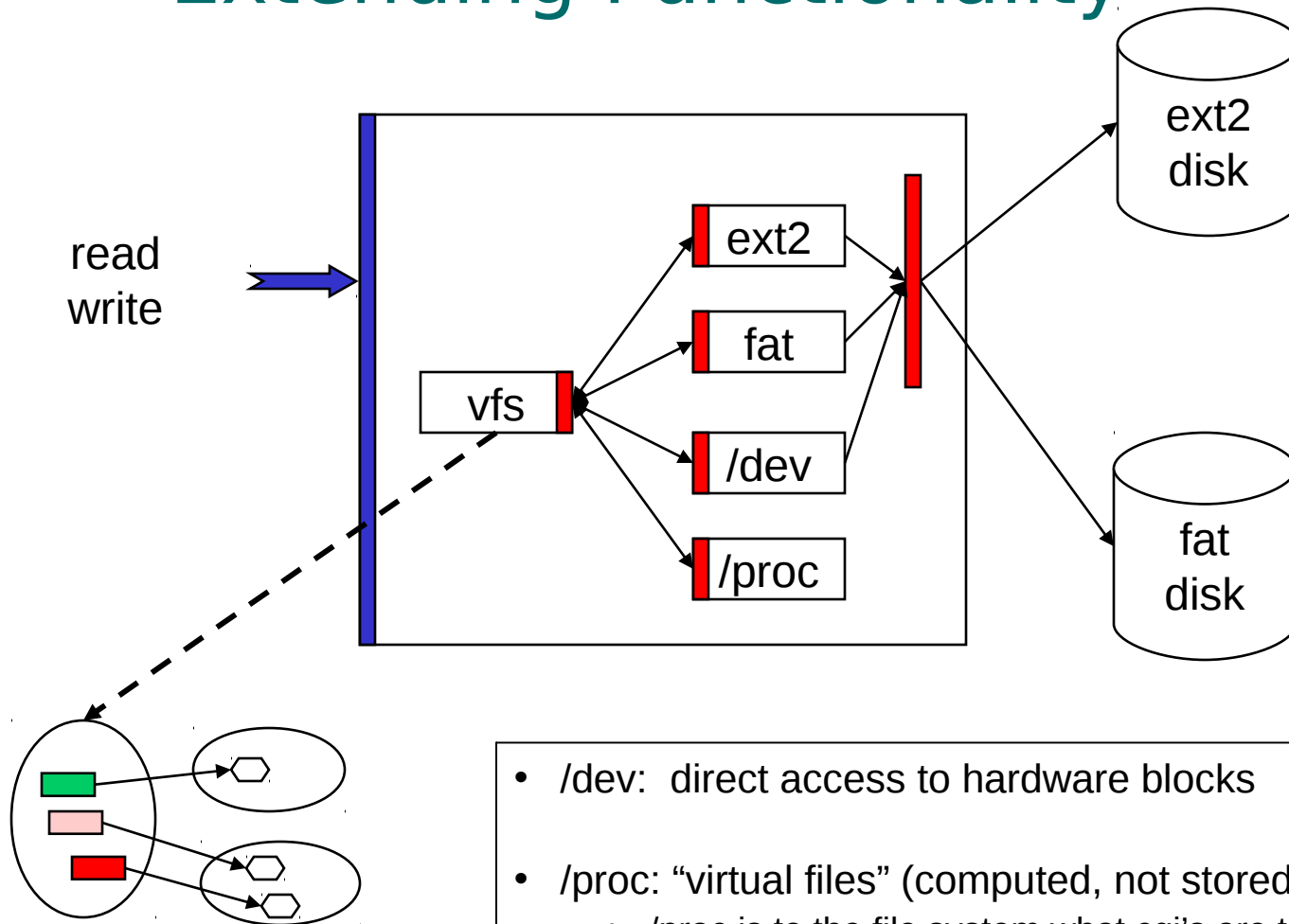
- Decides which blocks to allocate for what (inodes, directories, file data)
- Decides what the format of a directory is

*Not really disks, but partitions.
We'll get to that later...*

Supporting Multiple File Systems: vfs



Extending Functionality



- /dev: direct access to hardware blocks
- /proc: "virtual files" (computed, not stored, results)
 - /proc is to the file system what cgi's are to a web server (sort of)

Are Directories Fundamental?

What is the logical role of the directories?

What practical role(s) do they have?

Are there alternatives?