Lecture 2: OS Structure

CSC 469H1F / CSC 2208H1F

Fall 2007

Angela Demke Brown

Overview

- Motivation: Why talk about structure?

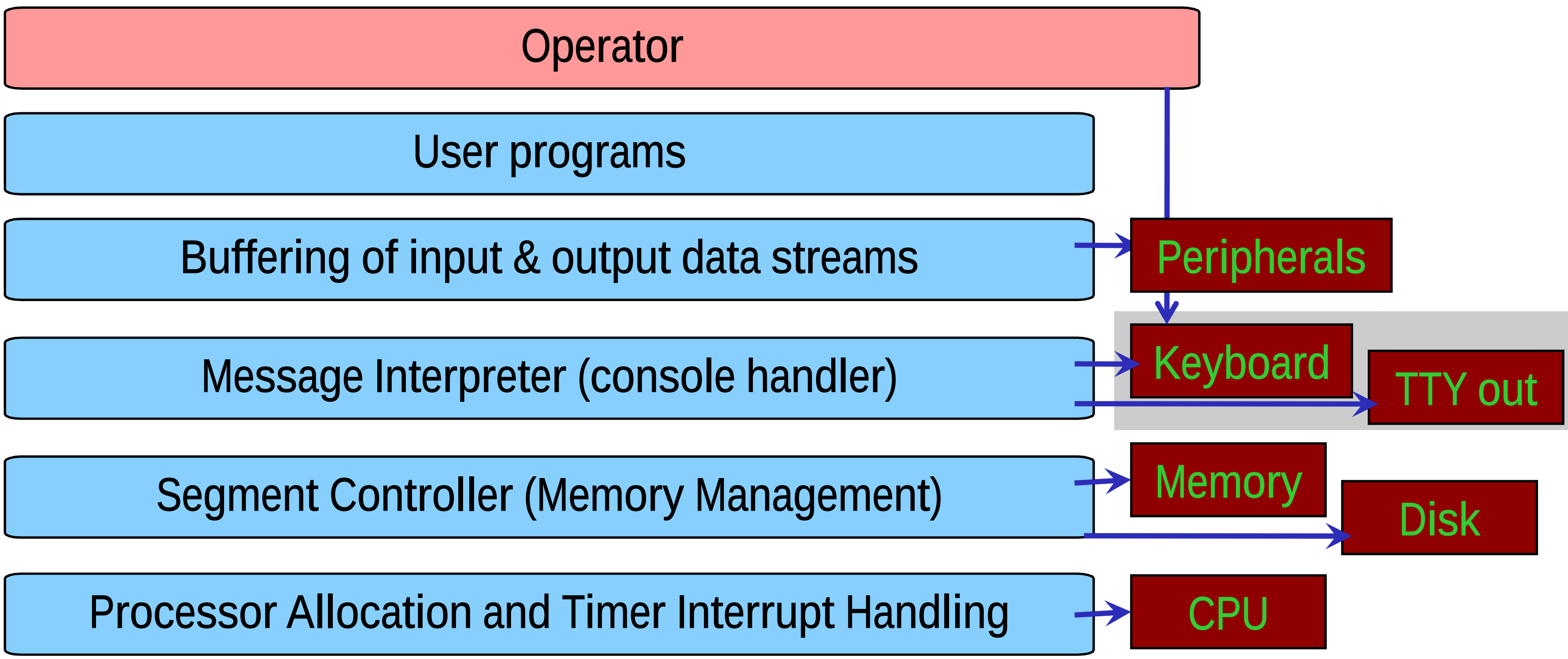
- Kernel structures
  - Layered systems
  - Monolithic kernels
  - Open systems
  - Microkernels
  - Kernel Extensions (Monday)
  - Virtual Machines (Monday)

Motivation

- Let's review what OS provides…
  - Abstraction layers
  - Protection boundaries
  - Resource allocators
  - Resource schedulers
- It's complicated! Size of Windows? Linux?
  - NT ~29 million S.L.O.C.
  - XP ~40 million S.L.O.C.
  - Vista ~50 million S.L.O.C.
  - 2.6.15 (March 2006) - 6.73 million S.L.O.C.

Early Layered System: THE

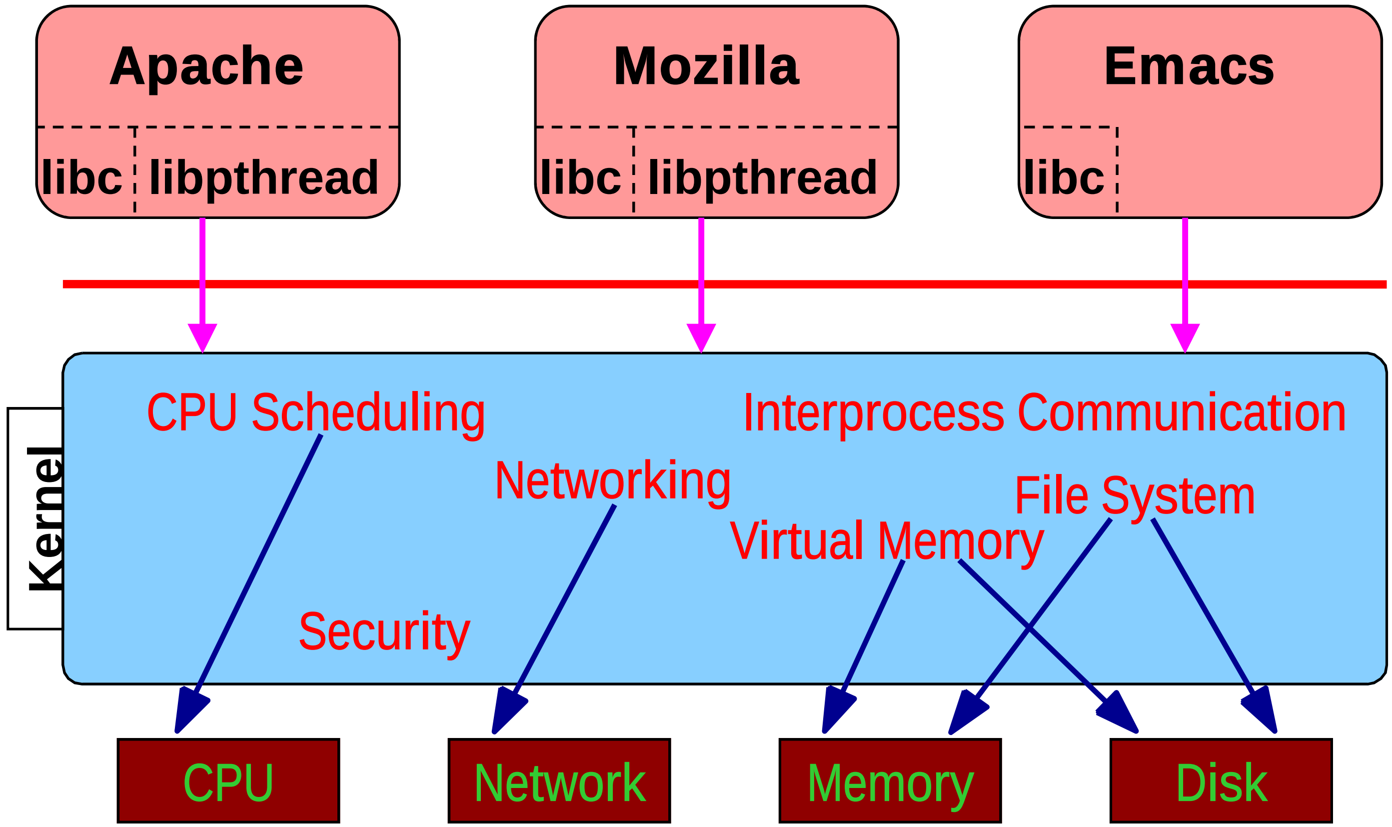
- Djikstra, 1st SOSP, 1967



Properties of Layered Systems

- Each layer has well-defined function and interface to layer above/below
  - Provides easier-to-use abstraction for higher layers
- Other examples: MULTICS (rings)
- Advantages?
  - Processes at any level can only invoke services of level below → no circular wait → no deadlock
  - each layer can be designed, implemented and tested independently
- Disadvantages?
  - Hard to partition functions into this strict hierarchy (why is console below other peripherals?)

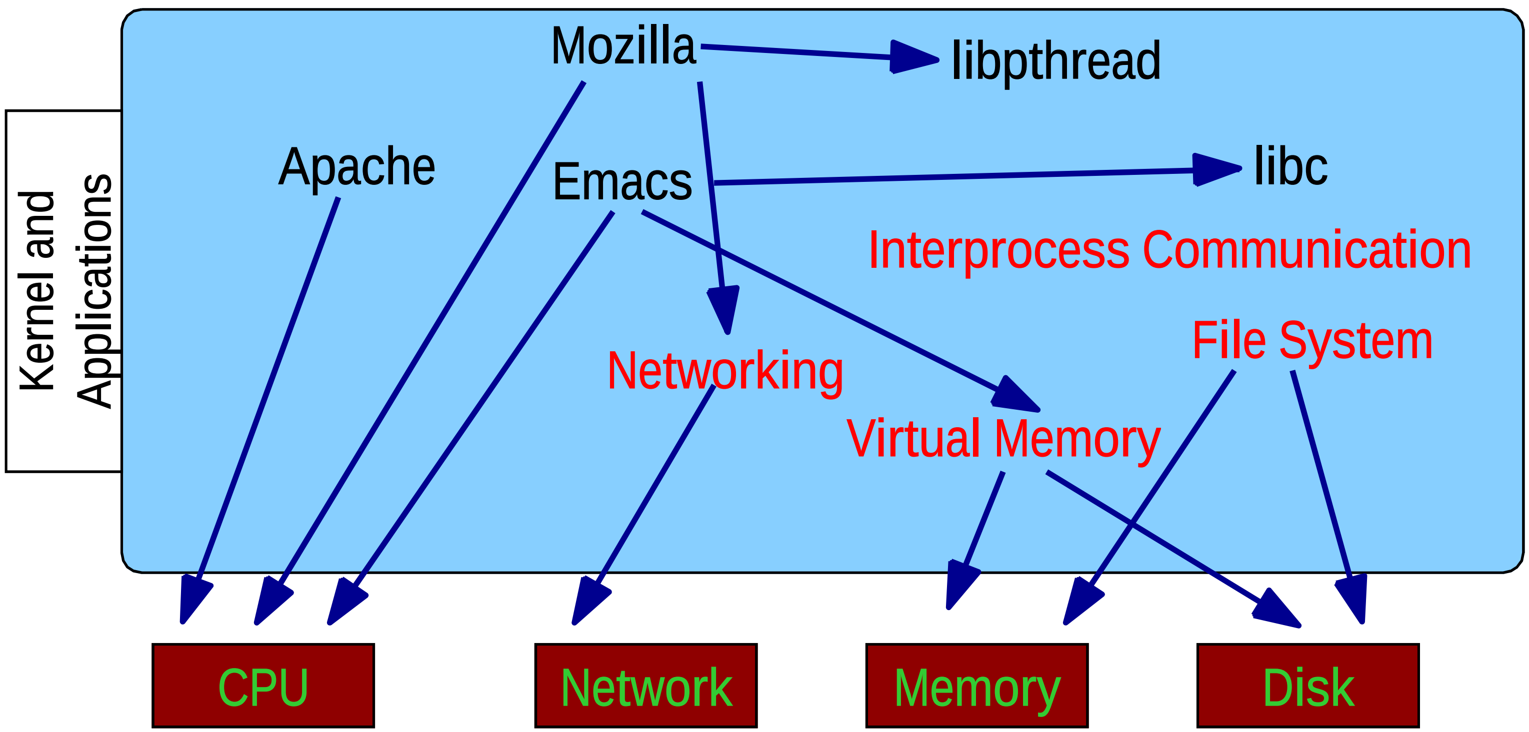
Monolithic OS



Properties of Monolithic Kernels

- OS is all in one place, below the “red line”
- Applications use a well-defined system call interface to interact with kernel
- Examples: Unix, Windows NT/XP, Linux, BSD, OS/161
  - Common in commercial systems
- Advantages?
  - Good performance, well-understood, easy for kernel developers, high level of protection between applications
- Disadvantages?
  - No protection between kernel components, not (safely, easily) extensible, overall structure becomes complicated (no clear boundaries between modules)

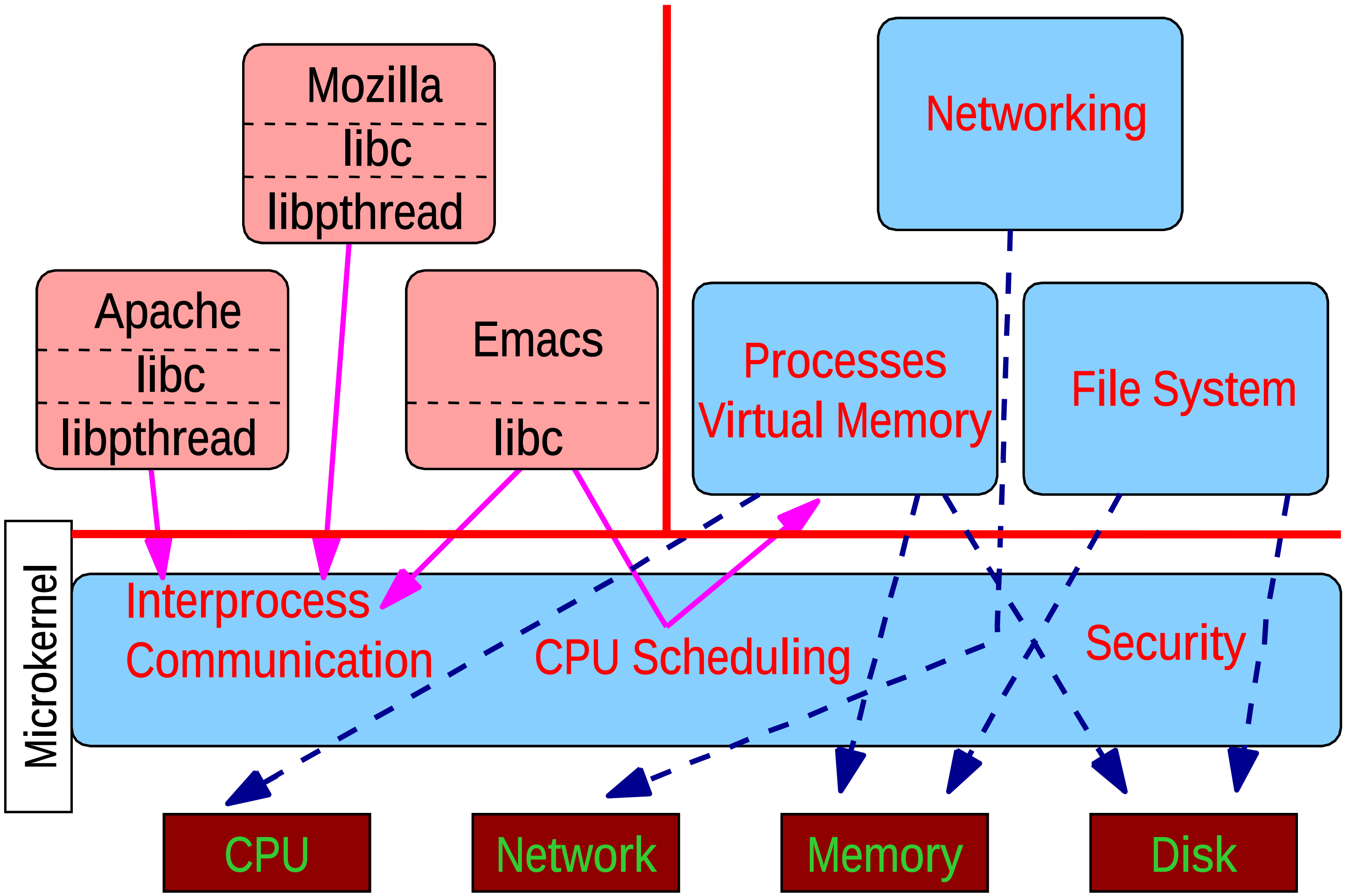
Open Systems



Properties of Open Systems

- Applications, libraries, kernel all in the same address space
- Crazy?
  - Idea first described by Lampson & Sproull, 7th SOSP, 1979 "An open operating system for a single-user machine"
  - MS-DOS; Mac OS 9 and earlier; Windows ME, 98, 95, 3.1
  - Palm OS and some embedded systems
- Used to be *very* common
- Advantages?
  - *Very* good performance, very extensible, works well for single-user
- Disadvantages?
  - No protection btwn kernel and/or apps, not very stable, composing extensions can lead to unpredictable behavior

Microkernel OS



Properties of Microkernels

- Design Philosophy: protected kernel code provides minimal "small, clean, logical" set of abstractions
  - Tasks and threads
  - Virtual memory
  - Interprocess communication
- Everything else is a *server process* running at user-level
- Early examples: Nucleus (1970),
- Later examples: Mach, Chorus, QNX, L4, GNU Hurd
- Mixed results …

Microkernel Advantages

- Extensible: add a new server to add new OS functionality
- Kernel does not determine operating system environment
  - Allows support for multiple OS *personalities*
  - Need an emulation server for each system (e.g. Mac, Windows, Unix)
  - All applications run on same microkernel
  - Applications can use customized OS (e.g. for databases)

More Advantages

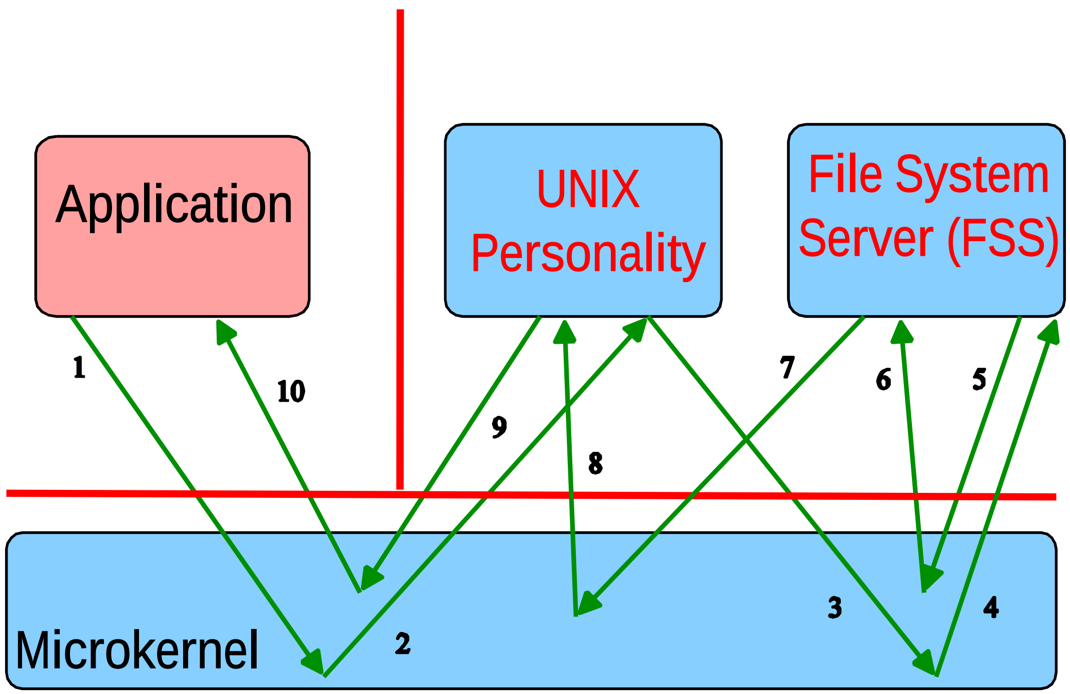
- Mostly hardware agnostic
  - Threads, IPC, user-level servers don't need to worry about underlying hardware
- Strong protection
  - Even of the OS against itself (i.e., the parts of the OS that are implemented as servers)
- Easy extension to multiprocessor and distributed systems

Microkernel Disadvantages

- Performance
  - System calls can require a lot of protection mode changes (next slide)
- Expensive to reimplement everything with a new model
  - OS personalities are easier to port to new hardware *after* porting to microkernel, but porting to microkernel may be *harder* than porting to new hardware
- Bad past history
  - See IBM Workplace OS story

Microkernel System Call Example

1. Application calls read(), traps to microkernel
2. microkernel sends message to Unix Personality requesting read
3. Unix personality sends message to File System Server (FSS) asking for data
4. FSS receives message and begins processing
5. FSS sends message to microkernel asking for disk blocks
6. Microkernel sends data back to FSS
7. FSS sends message to UNIX Personality with results
8. Unix Personality receives message with data
9. Unix Personality sends data to Application
10. Application receives data



The Mach Microkernel

- CMU Research Project
- The Plan:
  - Step 1: Proof of Concept
    - Take BSD 4.3 and "fix" VM, threads, IPC
  - Step 2: Microkernel and "single-server" Unix emulation
    - Take unix kernel and "saw it in half"
  - Step 3: Microkernel and multiple servers (for FS, paging, network, etc.)
    - Servers glued together by modules that catch system calls

Mach

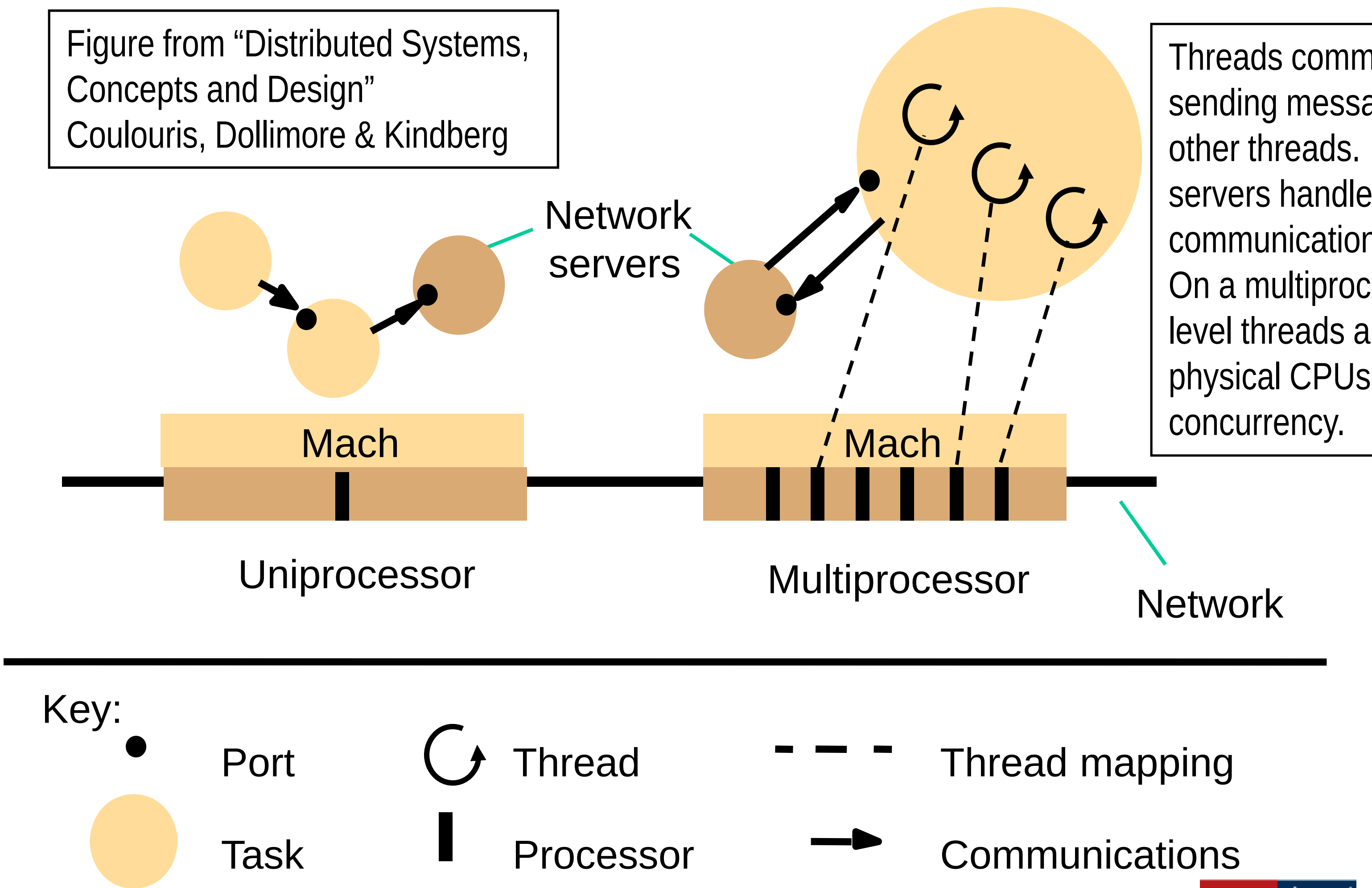
- Reality:
  - Proof of concept completed in 1989
    - Unix server, SMP support, kernel threads, 5 HW architectures
    - Commercial deployment: Encore Multimax, Convex Exemplar, OSF/1, NeXT (and eventually to OS X)
  - Microkernel and single-server completed and deployed to 10's of machines
  - Multi-server never fully completed
- Hugely influential

Key Mach Abstractions

- Tasks/threads
  - Tasks are passive (address space + resources)
  - Threads are active, perform computation
- Ports
  - Message origin / destination
  - Have access rights (embodied as capabilities)
  - Essentially an object reference mechanism
- Messages
  - Basis of all communication in Mach
- Devices
- Memory objects and memory cache objects

Tasks, threads and communication

Figure from "Distributed Systems, Concepts and Design"
Coulouris, Dollimore & Kindberg

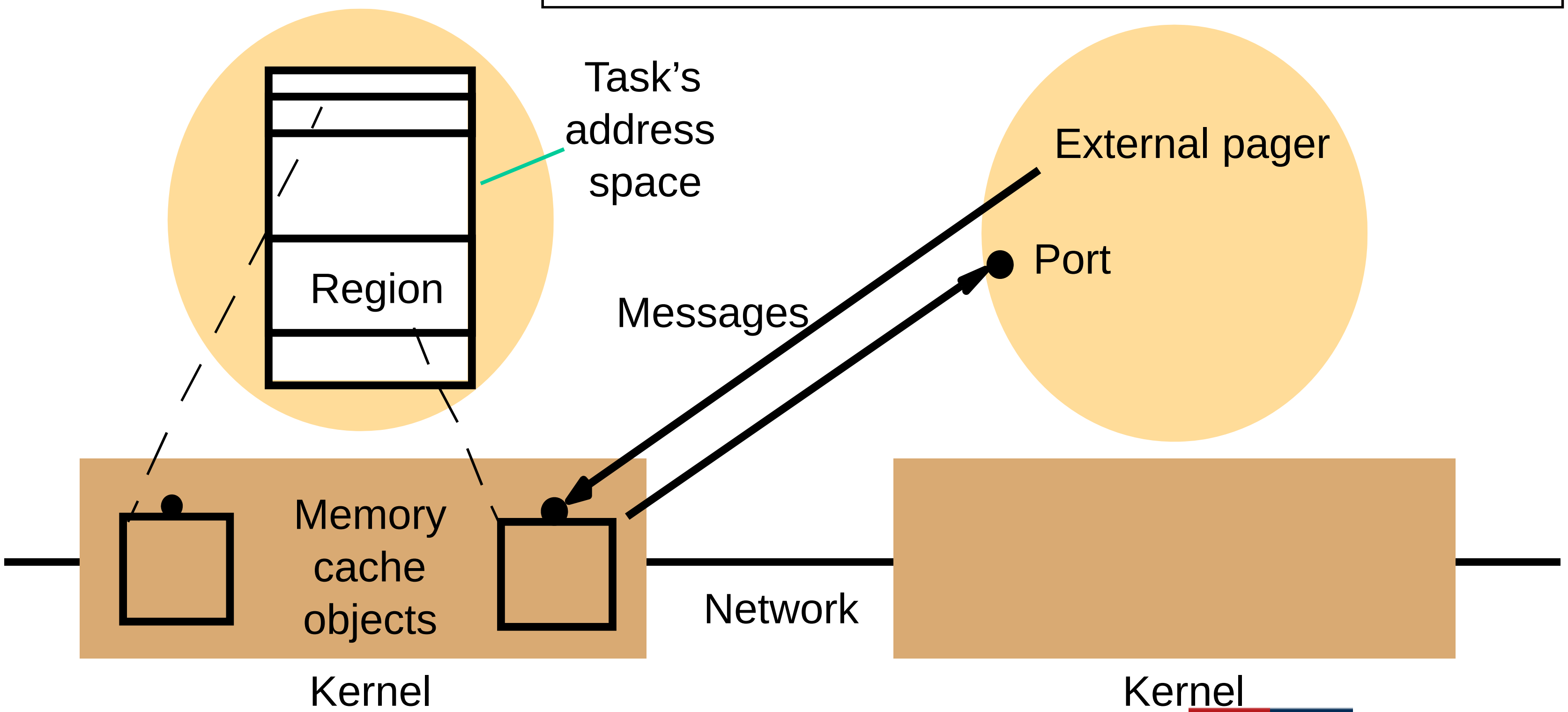


Threads communicate by sending messages to ports of other threads. Network servers handle distributed communication transparently. On a multiprocessor user-level threads are mapped to physical CPUs, providing true concurrency.

Mach External pager

Figure from "Distributed Systems, Concepts and Design"
Coulouris, Dollimore & Kindberg

Address space maps memory objects; microkernel maintains cache of memory object contents in physical memory while a user-level pager manages the backing store for each object. External pager may be on same, or different machine.



Next Time...

- OS Extensions
- Virtual machines

- Tutorial tomorrow: self-assessments