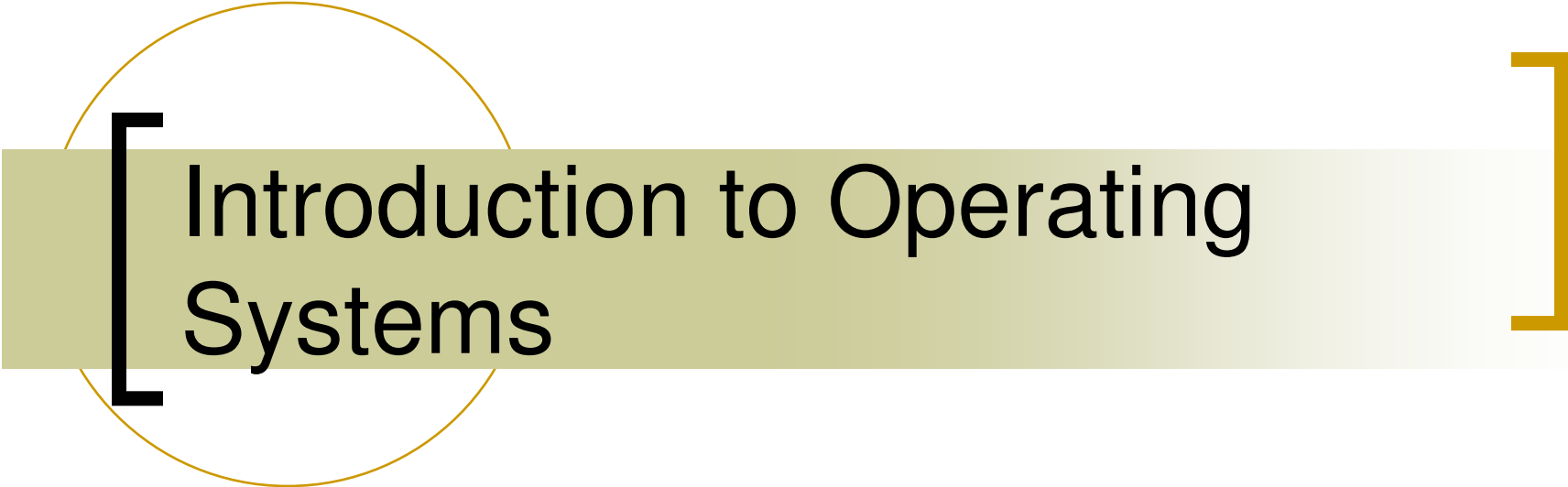




Introduction to Operating Systems

CS 447, Lecture 1

Monday, Aug 4, 2003

[What does a system consist of?]

- Components
- Coordination
- Behavior
 - Static Structure
 - Dynamic behavior
- External Interface – high level abstraction separated from system's implementation

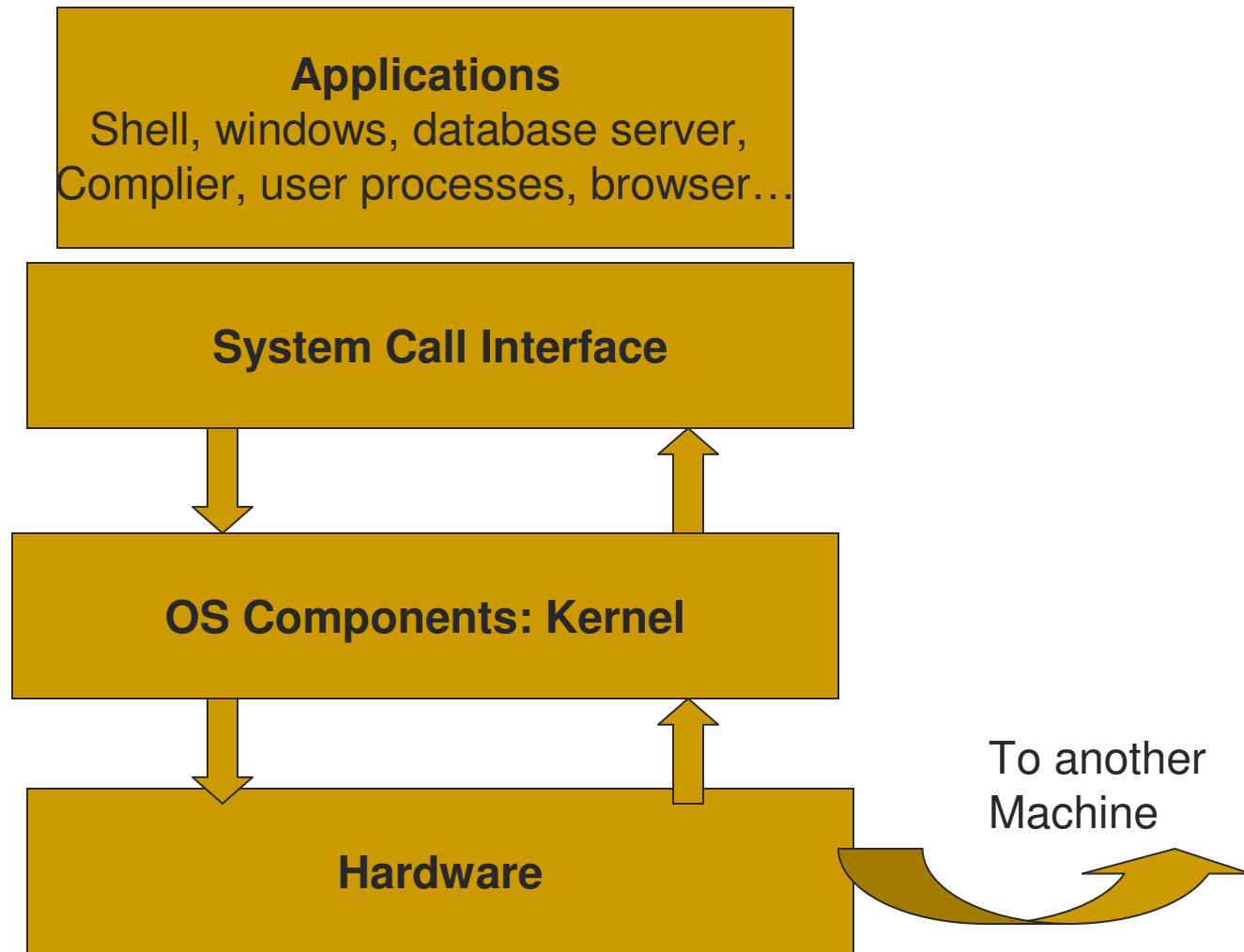
[Example Systems]

- Database Systems
- Operating Systems
- File Systems
- Multiprocessor Systems
- Distributed Systems
- A Document Processing System
- An Academic System

[Why study operating systems]

- Commonly used (everyday) system
- OS – An Important system in itself for CS curriculum
- As an example large system
 - What goes in making of such systems
 - Many algorithms, data structures, policies
 - New and interesting problems and their solutions
 - Architecture of such a system

[Overall Architecture - Layering]



[Trace the sequence of events:
keystroke → display of a char]

- Key hit → interrupt generated → stroke read in input buffer → waiting process signaled → waiting process wakes up → it completes its blocked read call → takes action after reading (e.g. display the character)

[What are the functions of an OS?]

- Manage Resources for users
 - CPU
 - Memory
 - Disk
 - Peripheral devices: keyboard, display, mouse, printer, ..
 - Networking
- Provide abstractions to use the system
 - Users, processes, files, synchronizers, messaging

[CPU requirements]

- Multiprogramming
- Fairness
- Fast response
- High utilization

[Memory Requirements]

- Efficient and fair Allocation
- Protection
- Memory hierarchies and consistency
- Kernel vs. User space
- Virtual memory

[Requirements for Disk Management]

- Allocation and Deallocation
- Fragmentation
- Structuring
- Protection
- Navigation
- Ability to work in coordination with memory

[Peripherals]

- Installation and deinstallation
- Interrupts and events/messaging
- Priorities
- Communication
- Uniform interface – device drivers
- Special requirements
 - Keyboard and display buffers

[OS Abstractions]

- Users
- Processes
- Files and Directories
- Memory Pages
- Synchronizers
- Messaging Primitives
- I/O

[System Calls]

- For all of the above abstractions
- The only way to interact with an OS
- All applications developed on top of system call layer
- Calls are made in user space, in user mode
- OS executes in kernel space and may also in user space but in kernel mode

[Users]

- Identity
- Ownerships
- Abilities
- Accounting

[Processes]

- Space allocation
- Allocation of Computational power
- Protection
- Kernel vs. user tasks
- Execution cycle and Multiprogramming
- Interconnectibility – devices and other processes
- Accounting

[Files]

- Create, Destroy, Read, Write, change modes,...
- Block sizes
- Protection
- Modes
- Uniform interface for multiple types of file systems

[Memory]

- Pages
- Allocation and deallocation
- accounting

[Synchronizers and messaging]

- Create
- Connect
- Send/receive/operate
- Close/destroy
- accounting

[Course Book: Get hold of any
of the below books]

- Silberschatz, Galvin
- Dhamdhere
- Stallings