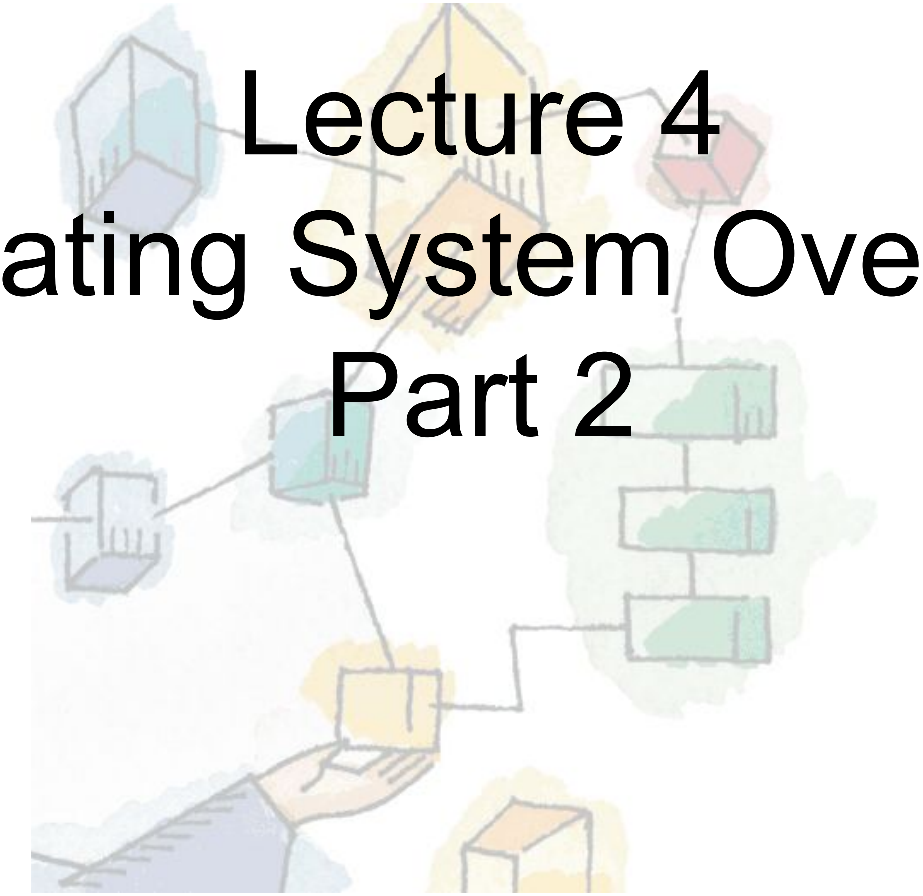
Operating Systems:
Internals and Design Principles, 6/E
William Stallings

Lecture 4
Operating System Overview.
Part 2



Patricia Roy
Manatee Community College, Venice, FL

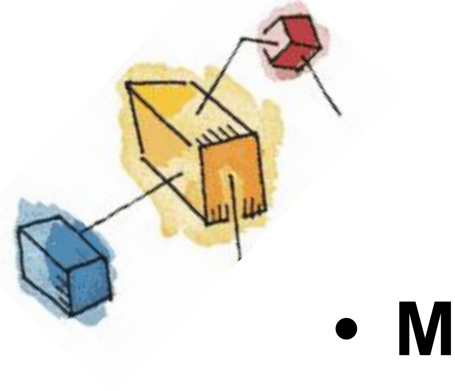
Outline

- **Major achievements**
  - The process
  - Memory management
  - Information protection and security
  - Scheduling and resource management
  - System structure
- Developments leading to modern OSs

Major Achievements

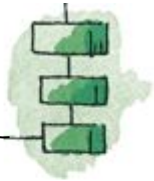
There have been 5 major advances in the development of OSs:

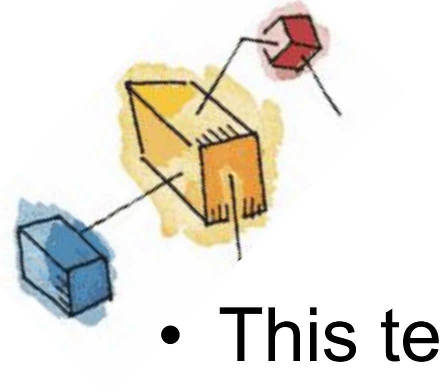
- Processes
- Memory management
- Information protection and security
- Scheduling and resource management
- System structure



Outline

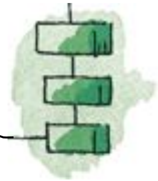
- **Major achievements**
  - **The process**
  - Memory management
  - Information protection and security
  - Scheduling and resource management
  - System structure
- Developments leading to modern OSs





Process

- This term was first used by the designers of Multics in the 1960s
- Definitions
  - A program in execution
  - An instance of a program running on a computer
  - The entity that can be assigned to and executed on a processor
  - A unit of activity characterized by
    - a single sequential thread of execution
    - a current state
    - an associated set of system resources



Difficulties with Designing System Software

Problems in timing and synchronization that contributed to the development of the concept of the process:

- Multiprogramming
  - Key design objective: to keep the processor and I/O devices simultaneously busy to achieve maximum efficiency.
  - Key mechanism: in response to signals indicating the completion of I/O transactions, the processor is switched among the various programs residing in main memory.

Difficulties with Designing System Software

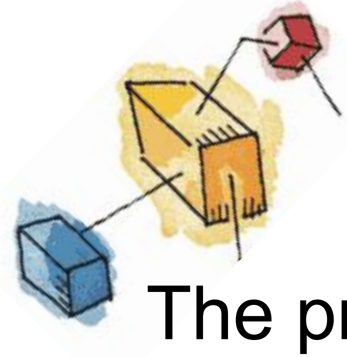
Problems in timing and synchronization that contributed to the development of the concept of the process:

- General purpose time-sharing
  - Key design objective: to be responsive to the needs of the individual user be able to support many users simultaneously
  - Key mechanism: typical user needs an average 2 seconds of processing time per minute, then close to 30 such users should be able to share the same system without noticeable interference

Difficulties with Designing System Software

Problems in timing and synchronization that contributed to the development of the concept of the process:

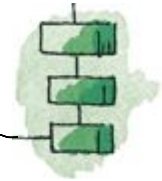
- Real-time transaction processing systems
  - Key design objective: a number of users are entering queries or updates against a database (example: an airline reservation system)
  - Key mechanism: response time is very important

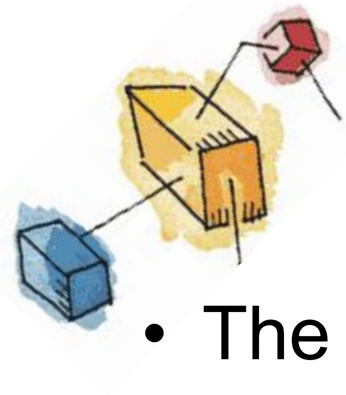


Difficulties with Designing System Software

The principal tool available to system programmers in developing the early multiprogramming and multiuser interactive systems – interrupt

- the activity of any job could be suspended
  - by a definite event or an I/O completion
- the processor would
  - save some sort of context (PC and other registers)
  - branch to an interrupt-handling routine
- an interrupt-handling routine would
  - determine the nature of the interrupt
  - process the interrupt
  - resume user processing with the interrupted job

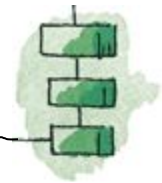




Difficulties with Designing System Software

- The design turned out to be remarkably difficult
  - many jobs in progress at any one time
  - each job involved numerous steps to be performed in sequence

  => it became impossible to analyze all of the possible combinations of sequences of events
- The absence of
  - some systematic means of coordination and cooperation among activities
- Programmers used methods based on their understanding of the environment that the OS had to control



Difficulties with Designing System Software

- Vulnerability to subtle programming errors
  - Effects of these errors could be observed only when certain relatively rare sequences of actions occurred
  - These errors were difficult to diagnose they needed to be distinguished from application software errors and hardware errors
  - Even when the error was detected, it was difficult to determine the cause, because the precise conditions under which the errors appeared were very hard to reproduce
- There 4 main causes of such errors

Difficulties with Designing System Software

- Improper synchronization -
  - a routine must be suspended awaiting an event elsewhere in the system
  - a signal from some other routine is required
  - improper design of the signaling mechanism: lost signals, duplicate signals received
- Failed mutual exclusion
  - multiple programs using shared resources at the same time
  - mechanism that permits only one routine at a time to perform an update against file

Difficulties with Designing System Software

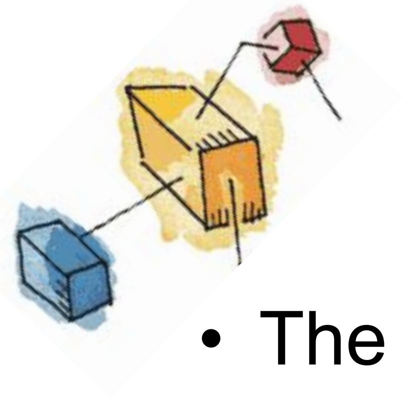
- Nondeterminate program operation
  - the results of a particular program depend on the input to that program
  - not on the activities of other programs in shared memory (overwriting common memory areas)
  - the order in which various programs are scheduled may affect the outcome of any particular program
- Deadlocks
  - two or more programs hung up waiting for each other

Process

- Systematic way to monitor and control the various programs executing on the processor
  - is needed to tackles these problems
- Process consists of three components
  - an executable program
  - the associated data needed by the program
    - variables
    - work space
    - buffers, etc.
  - the execution context of the program

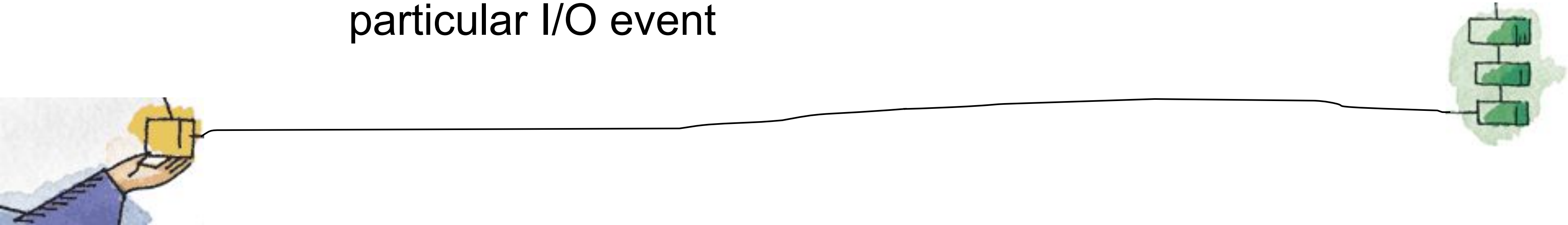
Process

- The **execution context**, or **process state**
  - is the internal data by which the OS is able to supervise and control the process
- This internal information is separated from the process
  - because the OS has information not permitted to the process



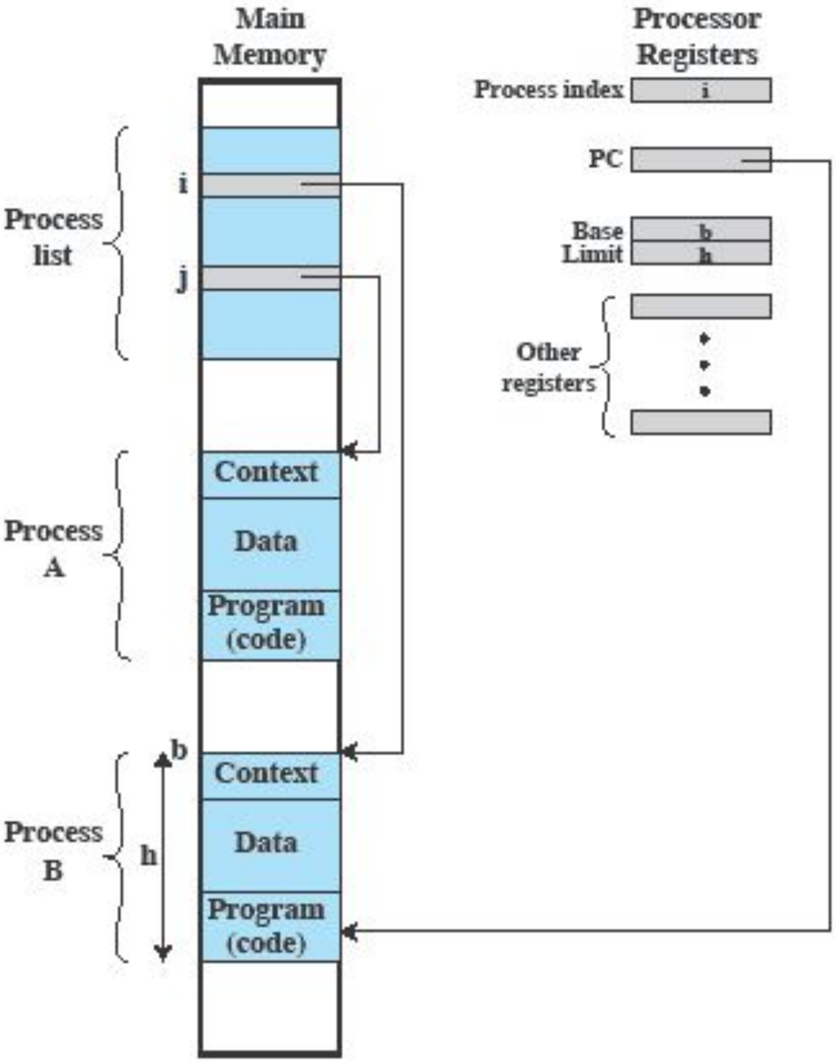
Process

- The context includes all the information that
  - the OS needs to manage the process
  - the processor needs to execute the process properly
- The context includes
  - the contents of the various processor registers (PC, data registers)
  - information of use to the OS
    - priority of the process
    - whether the process is waiting for the completion of a particular I/O event



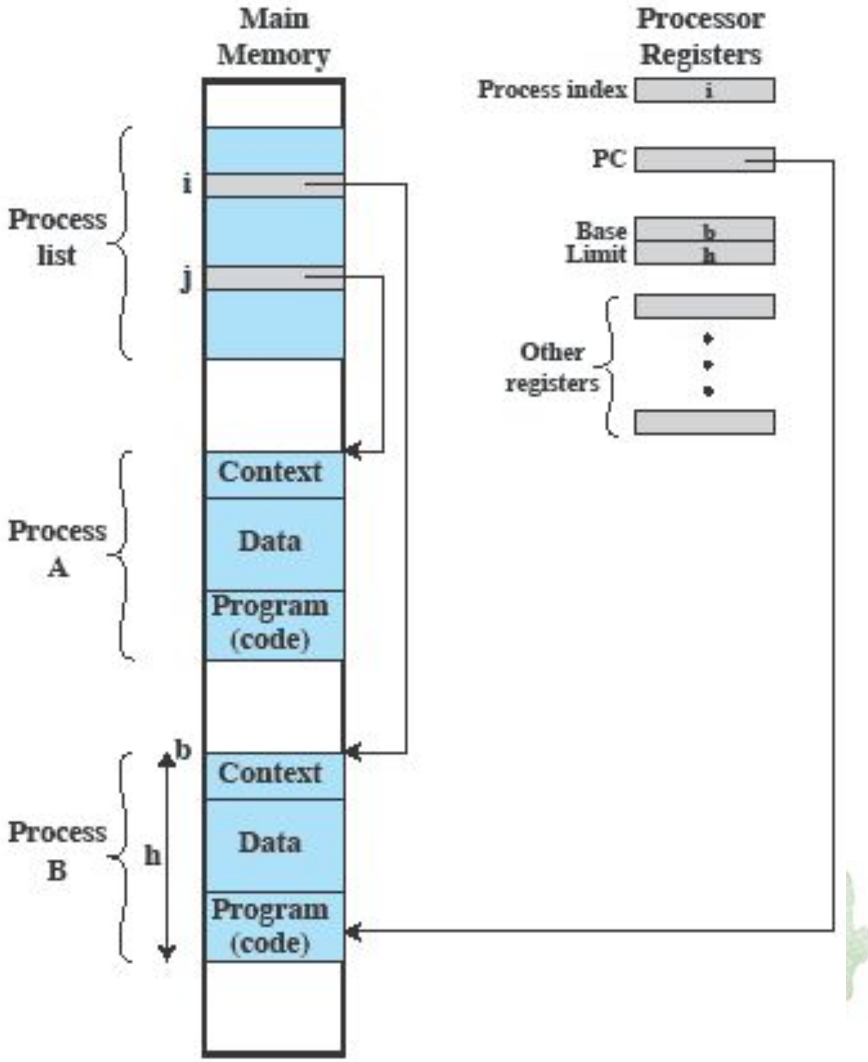
Typical process implementation

- Two processes A and B exist in the portions of main memory
- A block of memory is allocated to each process, contains
  - program
  - data
  - context
- An OS builds and maintains a process list with one entry for each process
  - a pointer to the location of the block of memory that contains the process
  - a part or all of the execution context of the process



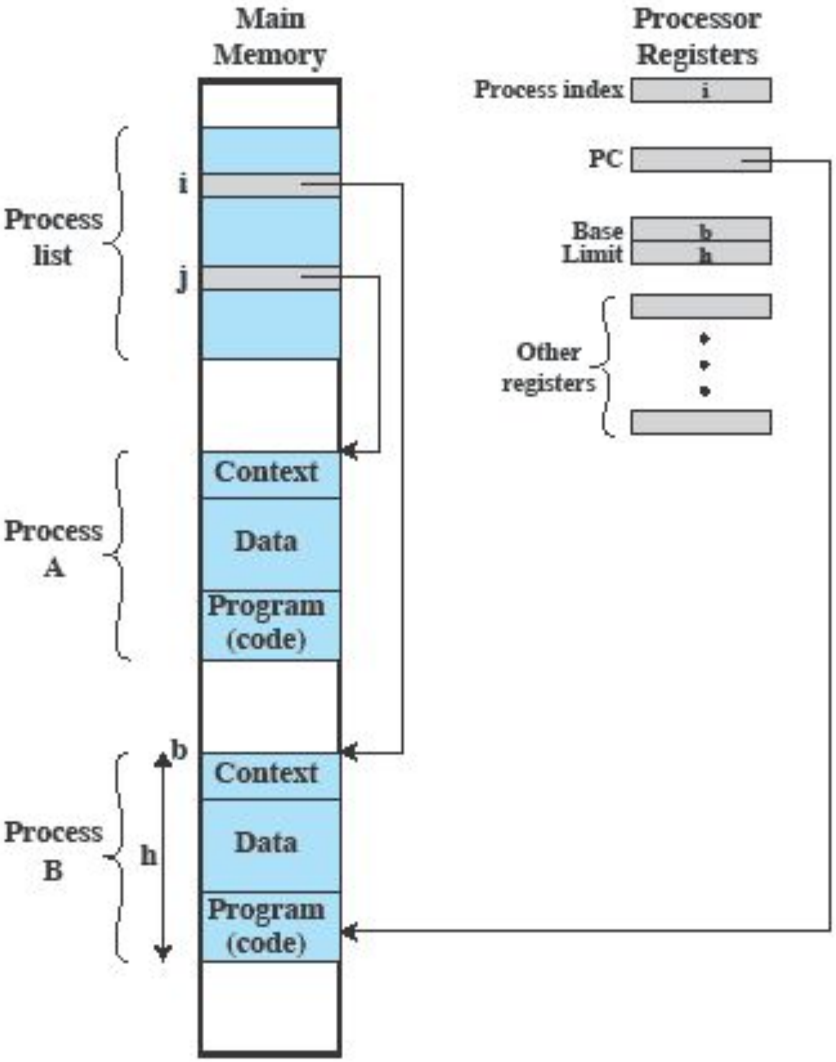
Typical process implementation

- The process index register contains the index into the process currently controlling the processor
- The program counter points to the next instruction in the process to be executed
- The base register and limit register define the region in memory occupied by the process



Typical process implementation

- The base register is the starting address of the region in the memory
- The limit register is the size of the region (in bytes or words)
- The program counter and all data references are
  - interpreted relative to the base register
  - must not exceed the value in the limit register
  - this prevents interprocess interference



Process

- Process is realized as a data structure
- Process can be executing or awaiting execution
- The entire state of the process at any instant is contained in its context
- This structure allows the development of powerful techniques for ensuring coordination and cooperation among processes

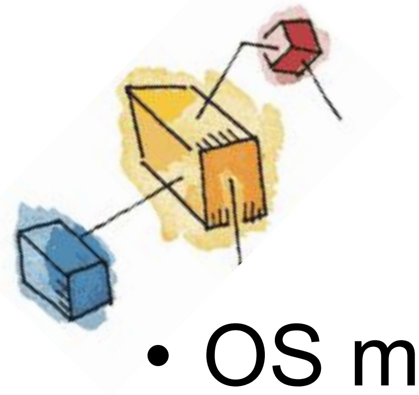
Outline

- **Major achievements**
  - The process
  - **Memory management**
  - Information protection and security
  - Scheduling and resource management
  - System structure
- Developments leading to modern OSs

Memory Management

Storage management responsibilities:

- Process isolation
- Automatic allocation and management
- Support of modular programming
- Protection and access control
- Long-term storage



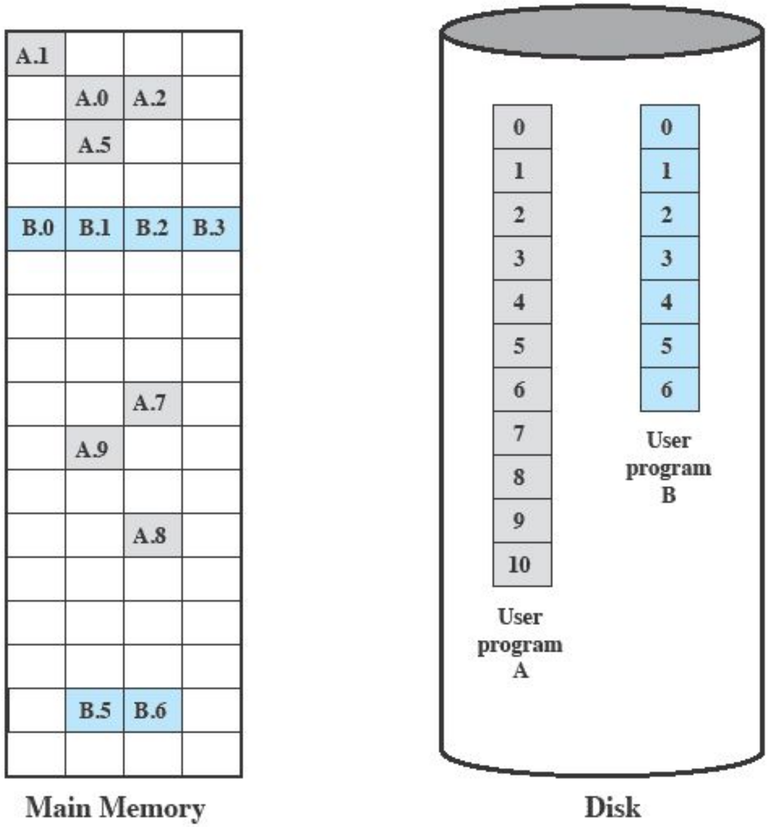
Virtual Memory

- OS meets these requirements with virtual memory and file system facilities.
- File system implements long-term store
- Information stored in named objects called files
- Virtual memory allows programs to address memory from a logical point of view, without regard to the amount of main memory physically available

Paging

- Processes vary in size
- If the processor switches among a number of processes, it is difficult to pack them compactly into main memory
- Paging systems allow process to be comprised of a number of fixed-size blocks, called pages
- Virtual address is a page number and an offset within the page
- Each page may be located anywhere in main memory
- Real address or physical address is the main memory address

Virtual Memory



A.1			
	A.0	A.2	
	A.5		
B.0	B.1	B.2	B.3
		A.7	
	A.9		
		A.8	
	B.5	B.6	

Main memory consists of a number of fixed-length frames, each equal to the size of a page. For a program to execute, some or all of its pages must be in main memory.

Secondary memory (disk) can hold many fixed-length pages. A user program consists of some number of pages. Pages for all programs plus the operating system are on disk, as are files.

Figure 2.9 Virtual Memory Concepts

Virtual Memory Addressing

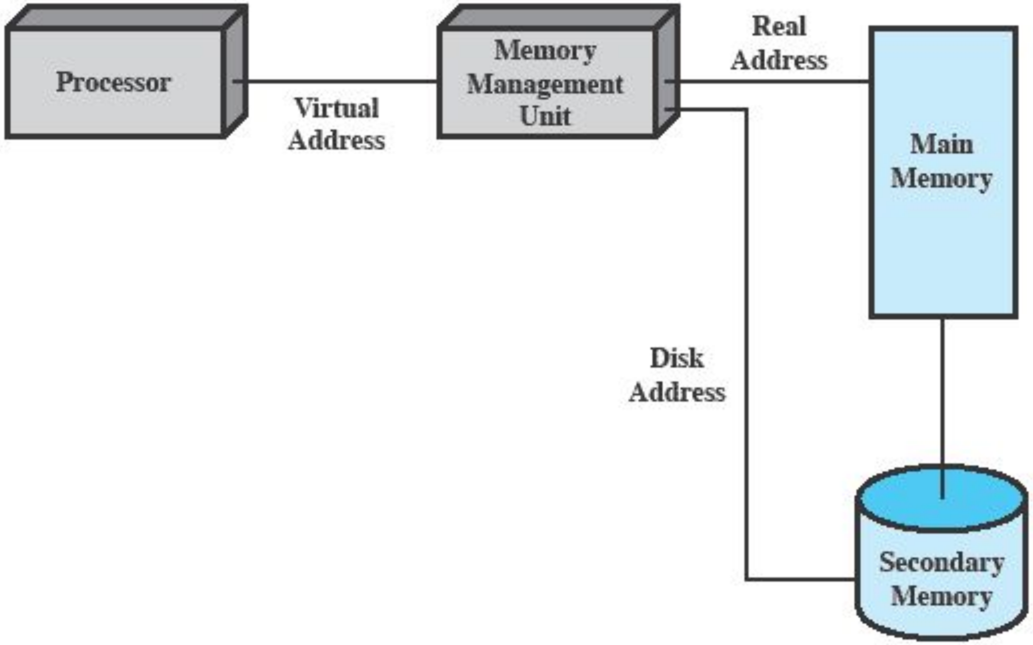


Figure 2.10 Virtual Memory Addressing

Outline

- **Major achievements**
  - The process
  - Memory management
  - **Information protection and security**
  - Scheduling and resource management
  - System structure
- Developments leading to modern OSs

Information Protection and Security

- Availability
  - Concerned with protecting the system against interruption
- Confidentiality
  - Assuring that users cannot read data for which access is unauthorized

Information Protection and Security

- Data integrity
  - Protection of data from unauthorized modification
- Authenticity
  - Concerned with the proper verification of the identity of users and the validity of messages or data

Outline

- **Major achievements**
  - The process
  - Memory management
  - Information protection and security
  - **Scheduling and resource management**
  - System structure
- Developments leading to modern OSs

Scheduling and Resource Management

- Fairness
  - Give equal and fair access to resources
- Differential responsiveness
  - Discriminate among different classes of jobs

Scheduling and Resource Management

- Efficiency
  - Maximize throughput, minimize response time, and accommodate as many uses as possible

Key Elements of an Operating System

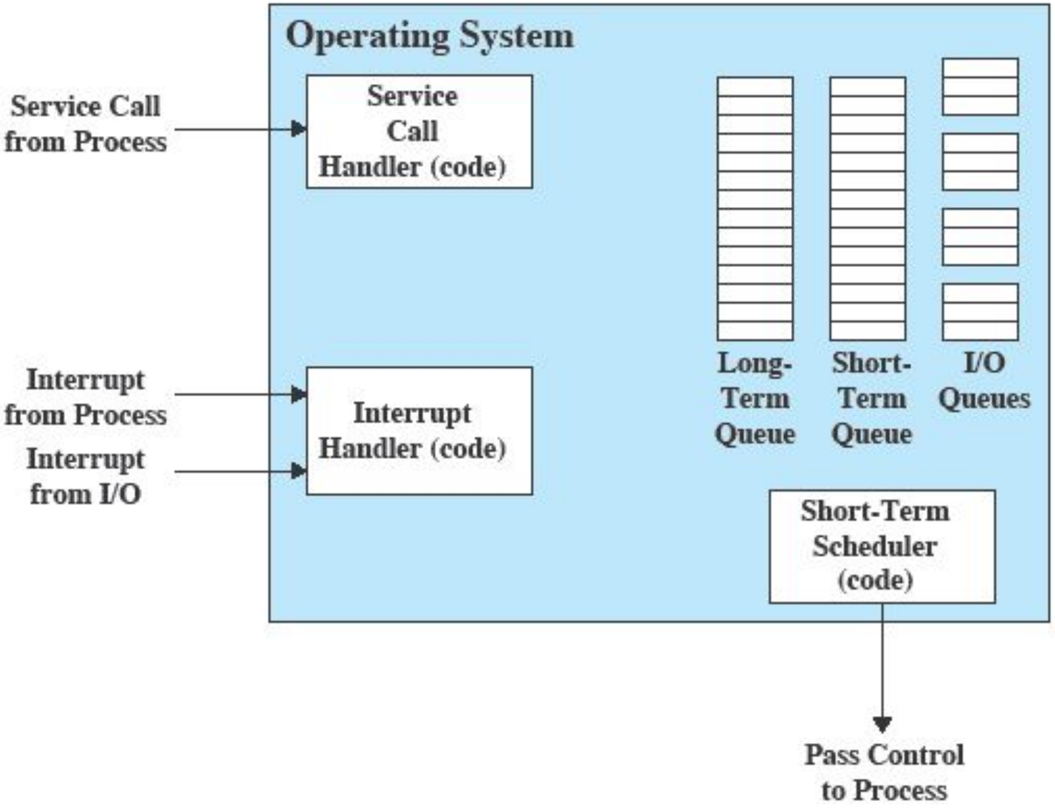


Figure 2.11 Key Elements of an Operating System for Multiprogramming

Outline

- **Major achievements**
  - The process
  - Memory management
  - Information protection and security
  - Scheduling and resource management
  - **System structure**
- Developments leading to modern OSs

System Structure

- View the system as a series of levels
- Each level performs a related subset of functions
- Each level relies on the next lower level to perform more primitive functions
- This decomposes a problem into a number of more manageable subproblems

Levels

- Level 1
  - – Electronic circuits
  - – Objects are registers, memory cells, and logic gates
  - – Operations are clearing a register or reading a memory location
- Level 2
  - – Processor’s instruction set
  - – Operations such as add, subtract, load, and store

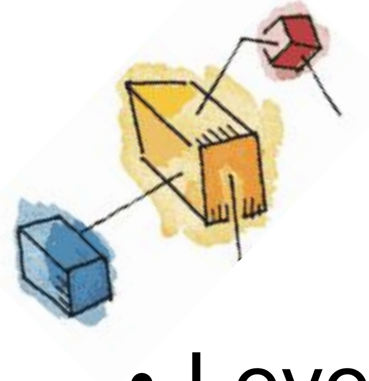
Levels

- Level 3
  - Adds the concept of a procedure or subroutine, plus call/return operations
- Level 4
  - Interrupts

Not part of the OS, constitute the processor hardware

Concepts Related to Multiprogramming

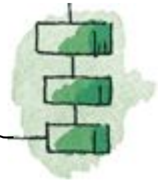
- Level 5
  - Process as a program in execution
  - Suspend and resume processes
- Level 6
  - Secondary storage devices
  - Transfer of blocks of data

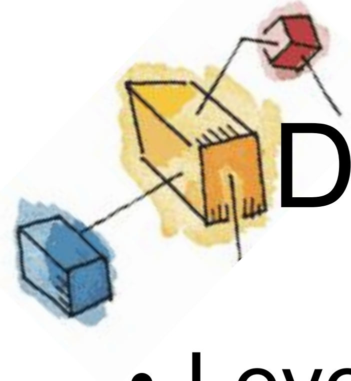


Concepts Related to Multiprogramming

- Level 7
  - – Creates logical address space for processes
  - – Organizes virtual address space into blocks

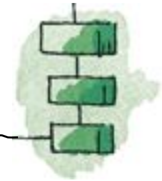
OS deals with the resources of a single processor





Deal with External Objects

- Level 8
  - Communication of information and messages between processes
- Level 9
  - Supports long-term storage of named files
- Level 10
  - Provides access to external devices using standardized interfaces



Deal with External Objects

- Level 11
  - Responsible for maintaining the association between the external and internal identifiers
- Level 12
  - Provides full-featured facility for the support of processes
- Level 13
  - Provides an interface to the OS for the user
- Hypothetical model of an OS:
  - ❑ provides useful descriptive structure
  - ❑ serves as an implementation guideline

Outline

- Major achievements
  - The process
  - Memory management
  - Information protection and security
  - Scheduling and resource management
  - System structure
- **Developments leading to modern OSs**

Modern Operating Systems

- Microkernel architecture
  - Assigns only a few essential functions to the kernel
    - Address spaces
    - Interprocess communication (IPC)
    - Basic scheduling

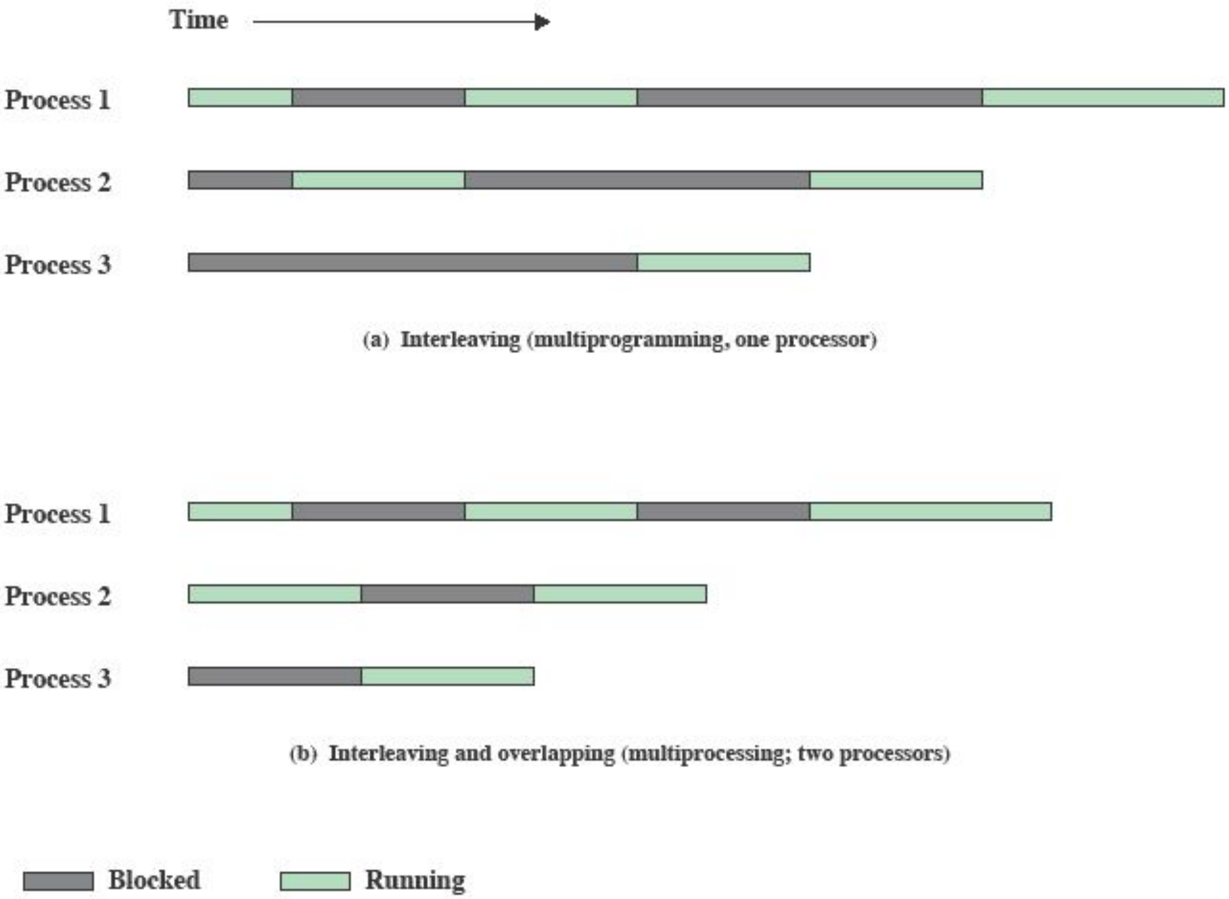
Modern Operating Systems

- Multithreading
  - Process is divided into threads that can run concurrently
    - Thread
      - Dispatchable unit of work
      - executes sequentially and is interruptable
    - Process is a collection of one or more threads

Modern Operating Systems

- Symmetric multiprocessing (SMP)
  - There are multiple processors
  - These processors share same main memory and I/O facilities
  - All processors can perform the same functions

Multiprogramming and Multiprocessing



(a) Interleaving (multiprogramming, one processor)

(b) Interleaving and overlapping (multiprocessing; two processors)

Figure 2.12 Multiprogramming and Multiprocessing

Modern Operating Systems

- Distributed operating systems
  - Provides the illusion of a single main memory space and single secondary memory space

Modern Operating Systems

- Object-oriented design
  - Used for adding modular extensions to a small kernel
  - Enables programmers to customize an operating system without disrupting system integrity