Operating Systems

Chapter 2 Operating System Overview

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

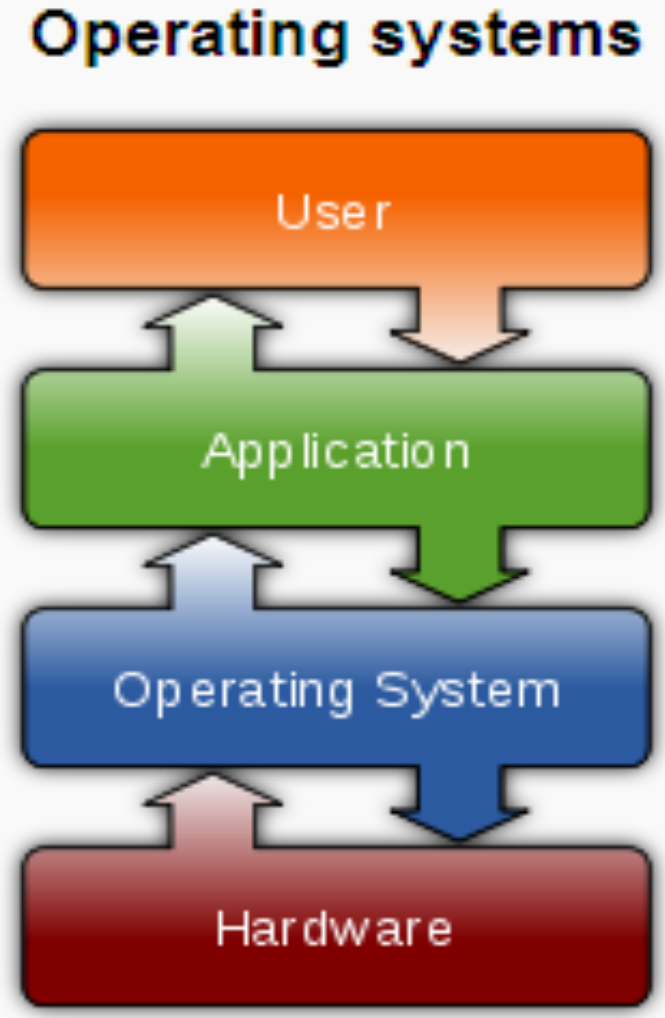
2.3 Major Achievements

2.4 Developments Leading to Modern Operating Systems

2.1 Operating System Objectives and Functions

2.1.0 Introduction (1/2)

- operating System definition:
  A program that
  - controls the execution of application programs
  - Acts as an interface between applications and hardware



2.1.0 Introduction (2/2)

- Operating System Objectives
  - As User/Computer Interface --Convenience/方便
    - Makes the computer more convenient to use
  - As Resource Manager--Efficiency /有效
    - Allows computer system resources to be used in an efficient manner
  - As System Software--Ability to evolve /扩展
    - Permit effective development, testing, and introduction of new system functions without interfering with service

2.1 Operating System Objectives and Functions

2.1.0 Introduction

2.1.1 The Operating System as a User/Computer Interface

2.1.2 The Operating System as Resource Manager

2.1.3 Ease of Evolution of an Operating System

2.1.1 The Operating System as a User/Computer Interface

- Layers of Computer System

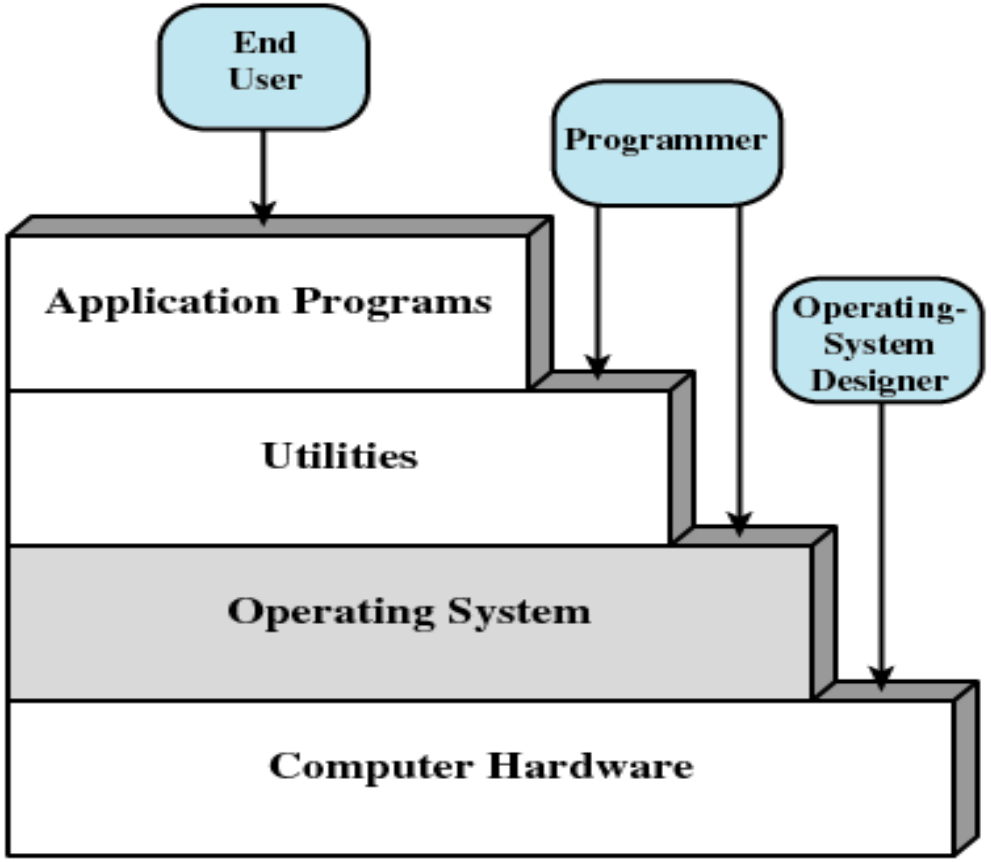


Figure 2.1 Layers and Views of a Computer System

2.1 Operating System Objectives and Functions

2.1.0 Introduction

2.1.1 The Operating System as a User/Computer Interface

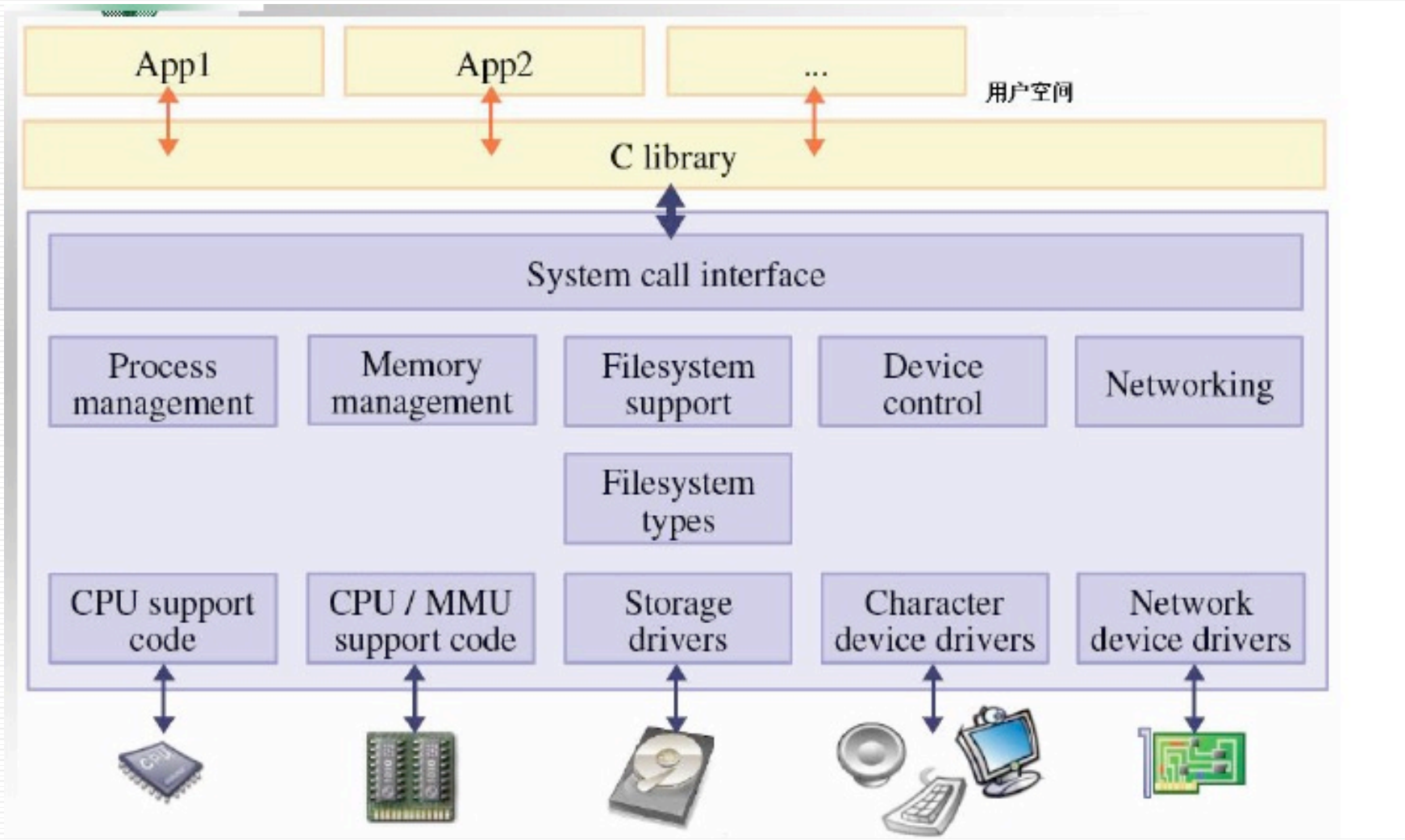
2.1.2 The Operating System as Resource Manager

2.1.3 Ease of Evolution of an Operating System

2.1.2 The Operating System as Resource Manager (1/3)

- The Operating System as Resource Manager
  - Responsible for managing resources/Hardware
  - Work in the same way as ordinary computer software
    - It is program that is executed
  - Operating system frequently relinquishes(放弃) and regains control of the processor

2.1.2 The Operating System as Resource Manager (2/3)



2.1.2 The Operating System as Resource Manager (3/3)

- Kernel(内核)
  - Portion of operating system that is in main memory
  - Contains most frequently used functions
  - Also called the nucleus（核子）

2.1 Operating System Objectives and Functions

2.1.0 Introduction

2.1.1 The Operating System as a User/Computer Interface

2.1.2 The Operating System as Resource Manager

2.1.3 Ease of Evolution of an Operating System

2.1.3 Ease of Evolution of an Operating System (1/1)

- Why ?
  - Hardware upgrades plus new types of hardware
  - New services
  - Fixes Bug

- How ?
  - Modularization
  - OS Architecture [for more detail later on 2.3.5]

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

2.3 Major Achievements

2.4 Developments Leading to Modern Operating Systems

2.2 The Evolution of Operating Systems

2.2.1 Serial Processing: No Operating Systems

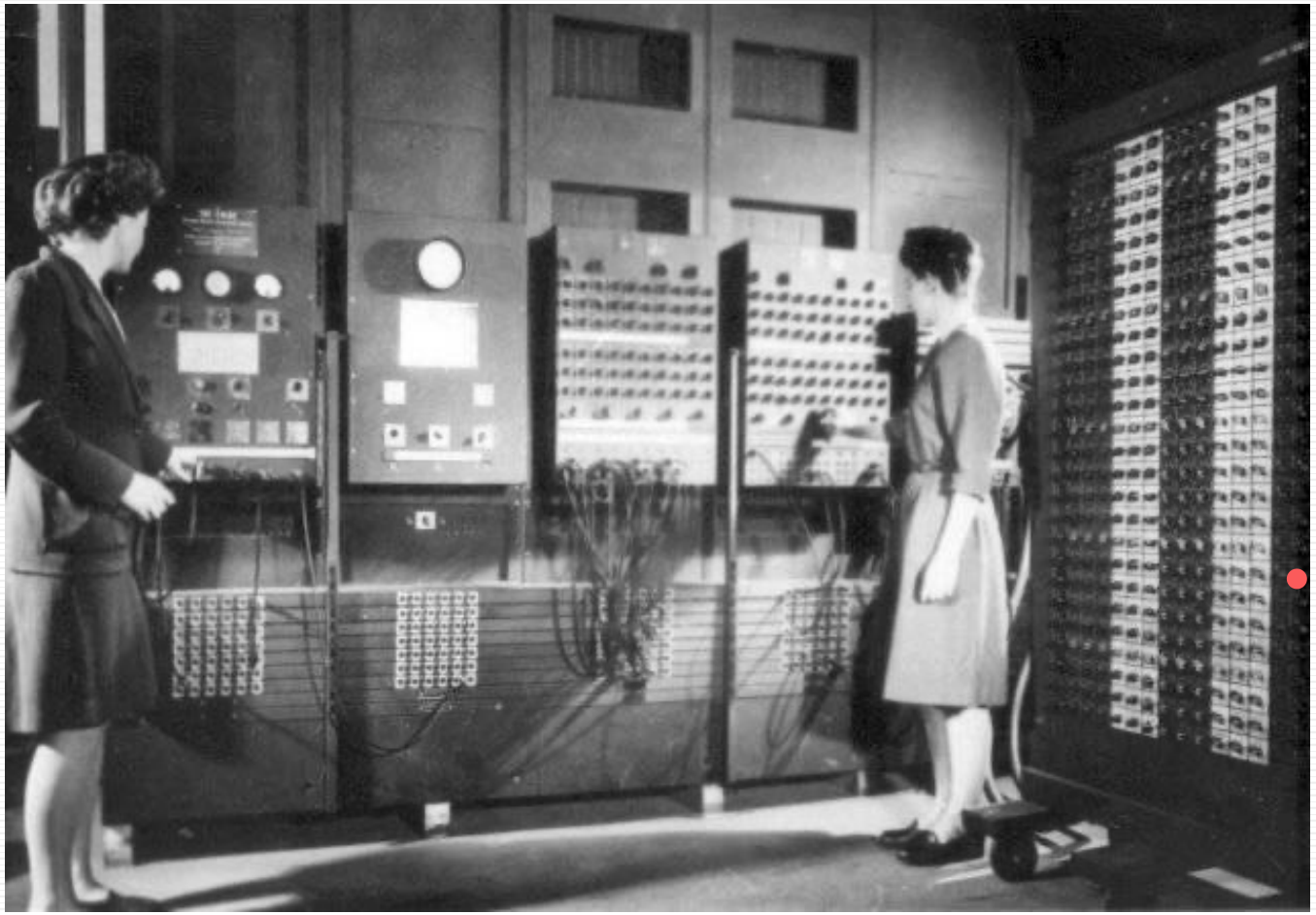
2.2.2 Simple Batch Systems

2.2.3 Multiprogrammed Batch Systems

2.2.4 Time-Sharing Systems

2.2.1 Serial Processing(1/2)

- 操作ENIAC



- Time
  - 1940s-1950s

2.2.1 Serial Processing(2/2)

- 命名起源：the user have access to the system in series
- 特点：the programmer interacted directly with the computer hardware
- 如何操作：Machines run from a console with display lights, toggle switches, input device, and printer
- Problem
  - Scheduling: Most installations装置 used a hardcopy sign-up sheet to reserve computer time
  - Setup准备time: included loading the compiler, source program, saving compiled program, and loading and linking

2.2 The Evolution of Operating Systems

2.2.1 Serial Processing : No Operating Systems

2.2.2 Simple Batch Systems

2.2.3 Multiprogrammed Batch Systems

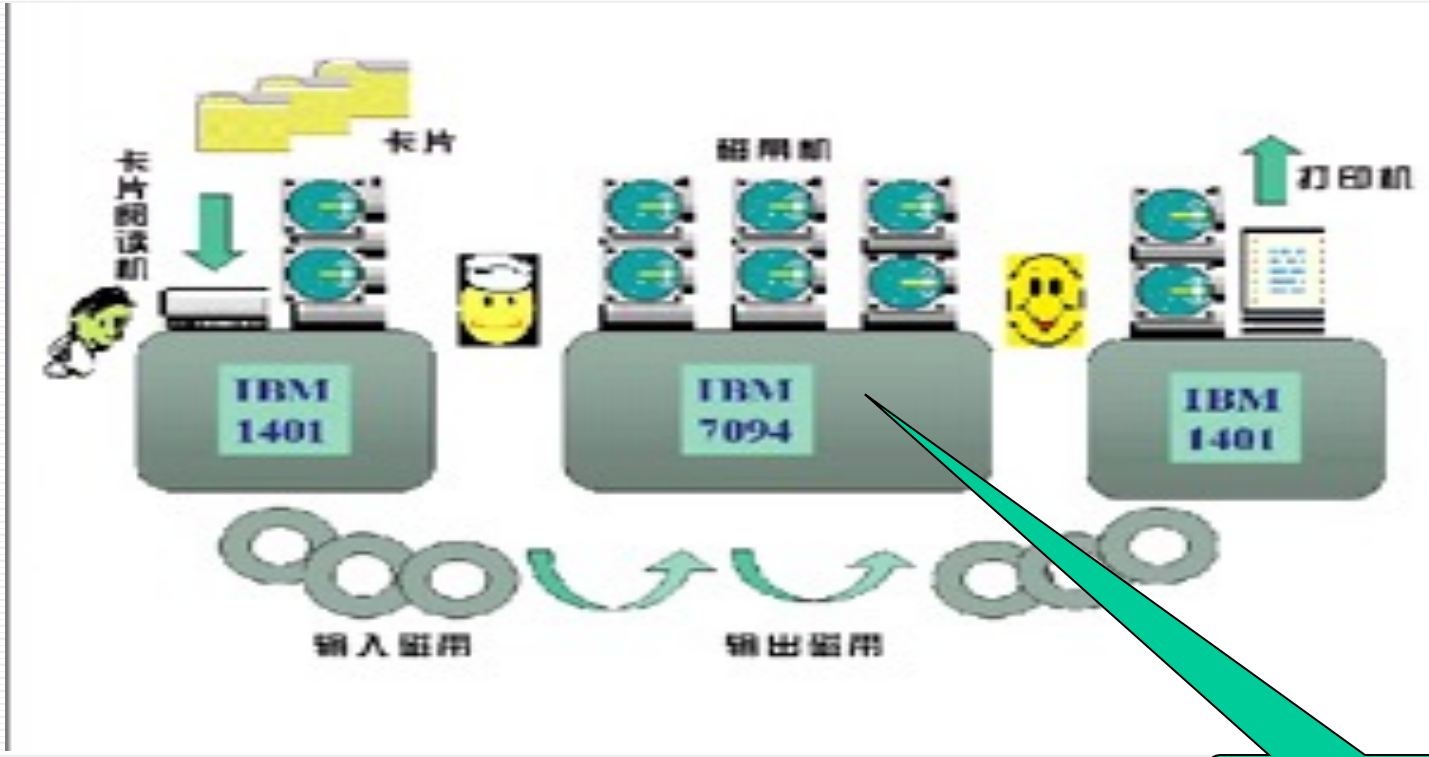
2.2.4 Time-Sharing Systems

2.2.2 Simple Batch Systems (1/7)

- The central idea behind the simple batch processing scheme is
  - Batch: made up by many jobs from users
    - A job may use several program
  - the use of a piece of software known as the monitor that controls the sequence of events.

2.2.2 Simple Batch Systems (2/7)

- 举例：IBM 7094 简单/单任务批处理系统
  - Batch / 批: 多个用户作业



2.2.2 Simple Batch Systems (3/7)

- The Process
  - The monitor reads a job from the batch
  - Then control is passed to this job
  - When the job is completed, it returns control to the monitor
  - The monitor reads in the next job

2.2.2 Simple Batch Systems (4/7)

- Job Control Language (JCL)
  - Special type of programming language
  - Provides instruction to the monitor
    - What compiler to use
    - What data to use
    - …….

2.2.2 Simple Batch Systems (5/7)

- Hardware Features
  - Memory protection
    - Do not allow the memory area containing the monitor to be altered
  - Timer
    - Prevents a job from monopolizing独占 the system

2.2.2 Simple Batch Systems (6/7)

- Hardware Features(cont.)
  - Privileged instructions
    - Certain machine level instructions can only be executed by the monitor
  - Interrupts
    - This feature gives the OS more flexibility in relinquishing control to and regaining control from user programs

2.2.2 Simple Batch Systems (7/7)

- CPU mode
  - User program executes in **user mode(用户模式)**
    - Certain instructions may not be executed
    - Some memory can not be accessed
  - Monitor executes in **system mode(系统模式)**
    - **Kernel mode(内核模式)**
    - Privileged instructions are executed
    - Protected areas of memory may be accessed

2.2 The Evolution of Operating Systems

2.2.1 Serial Processing : No Operating Systems

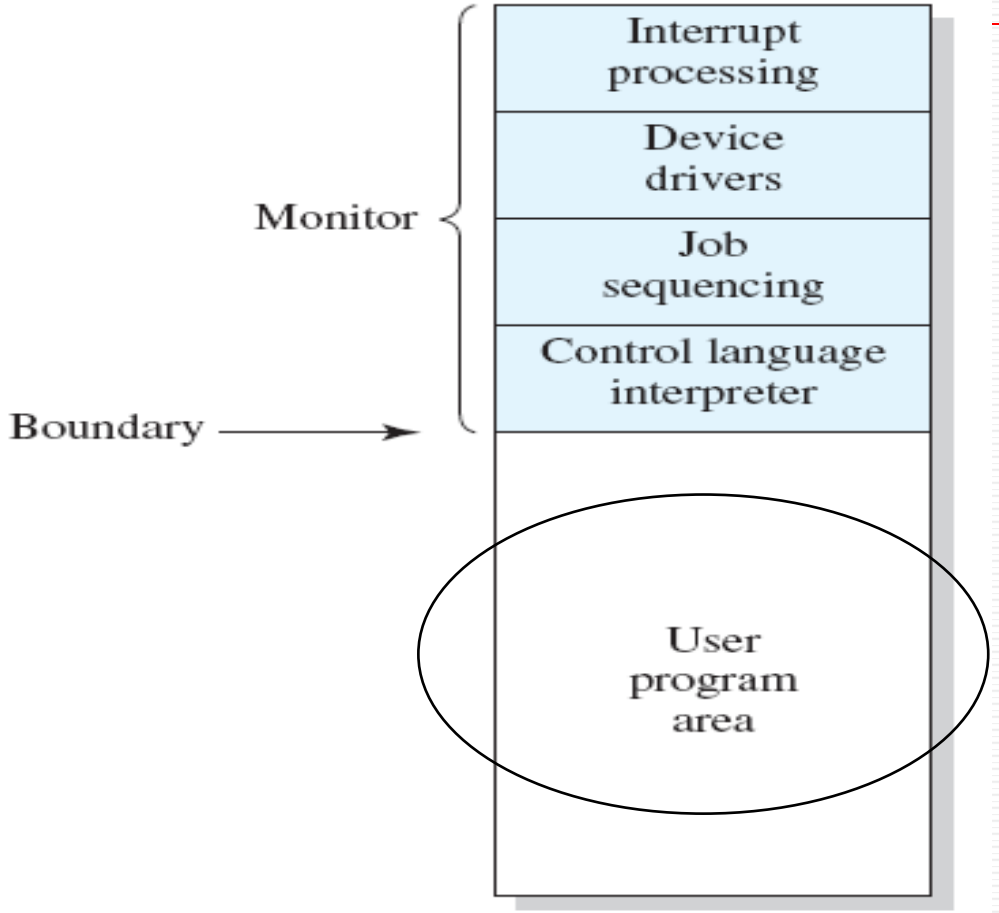
2.2.2 Simple Batch Systems

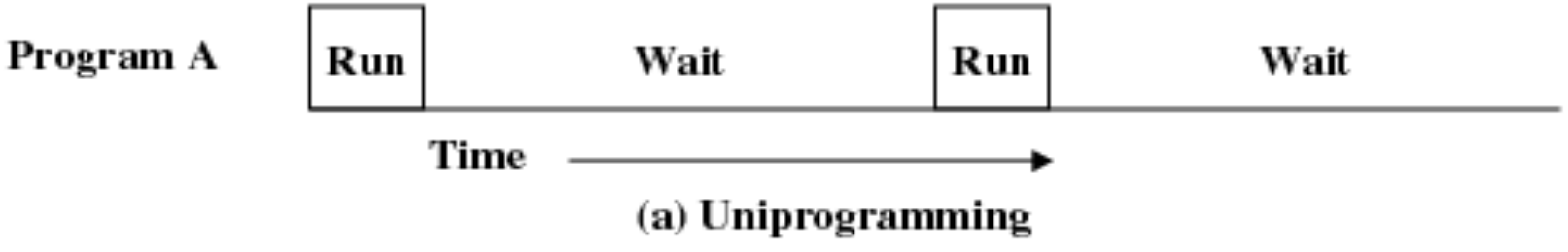
2.2.3 Multiprogrammed Batch Systems

2.2.4 Time-Sharing Systems

2.2.3 Multiprogrammed Batch Systems (1/6)

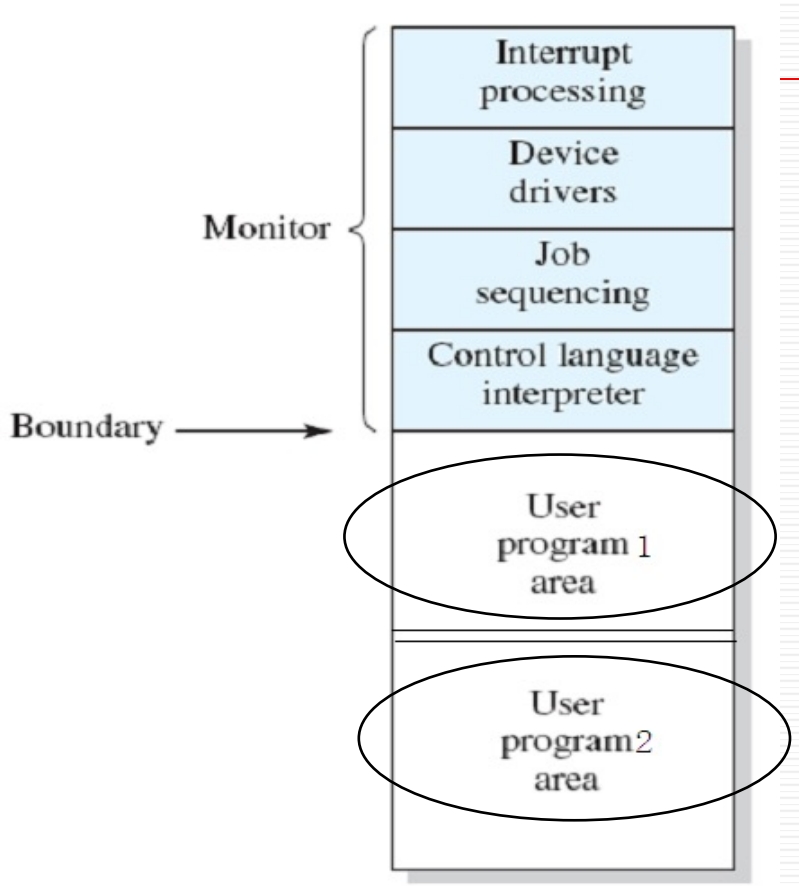
- Simple Batch Systems
  - Only one job in memory at a time

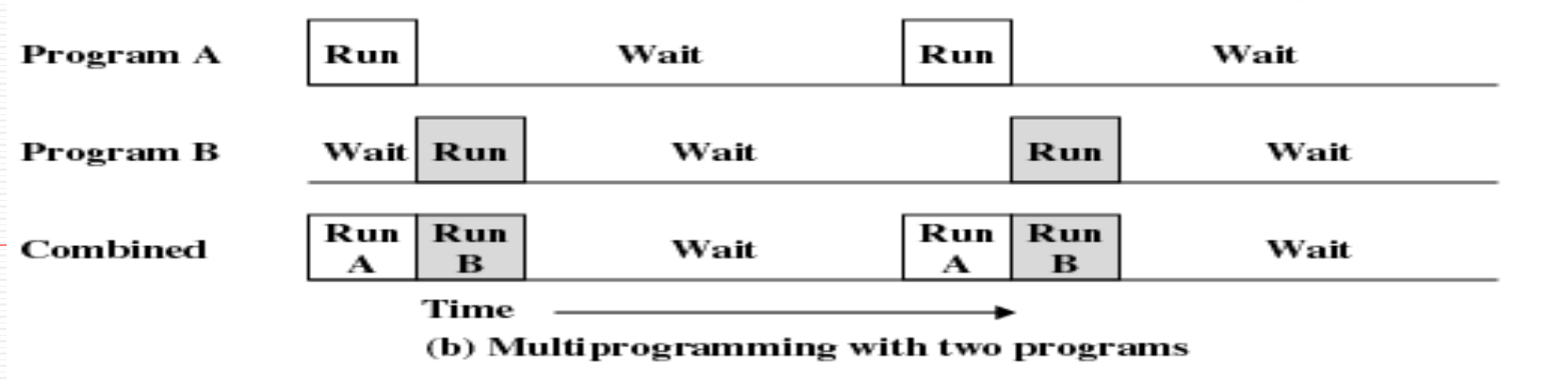




2.2.3 Multiprogrammed Batch Systems (2/6)

- Multiprogrammed Batch Systems
  - Multiple jobs in memory at a time
- Why ?
  - I/O Devices Slow
    - CPU not Busy





2.2.3 Multiprogrammed Batch Systems (3/6)

- Uniprogramming(单道程序设计)
  - Processor must wait for I/O instruction to complete before preceding
- Multiprogramming(多道程序设计)
  - When one job needs to wait for I/O, the processor can switch to the other job
  - Also known as Multitasking(多任务处理)

2.2.3 Multiprogrammed Batch Systems (4/6)

- Example: 3个作业/程序
  - 各自情况描述

Table 2.1 Sample Program Execution Attributes

	JOB1	JOB2	JOB3
Type of job	Heavy compute	Heavy I/O	Heavy I/O
Duration	5 min	15 min	10 min
Memory required	50 M	100 M	75 M
Need disk?	No	No	Yes
Need terminal?	No	Yes	No
Need printer?	No	No	Yes

2.2.3 Multiprogrammed Batch Systems (5/6)

- Example: 3个作业/程序 (cont.)
  - Utilization Histograms(利用率直方图)

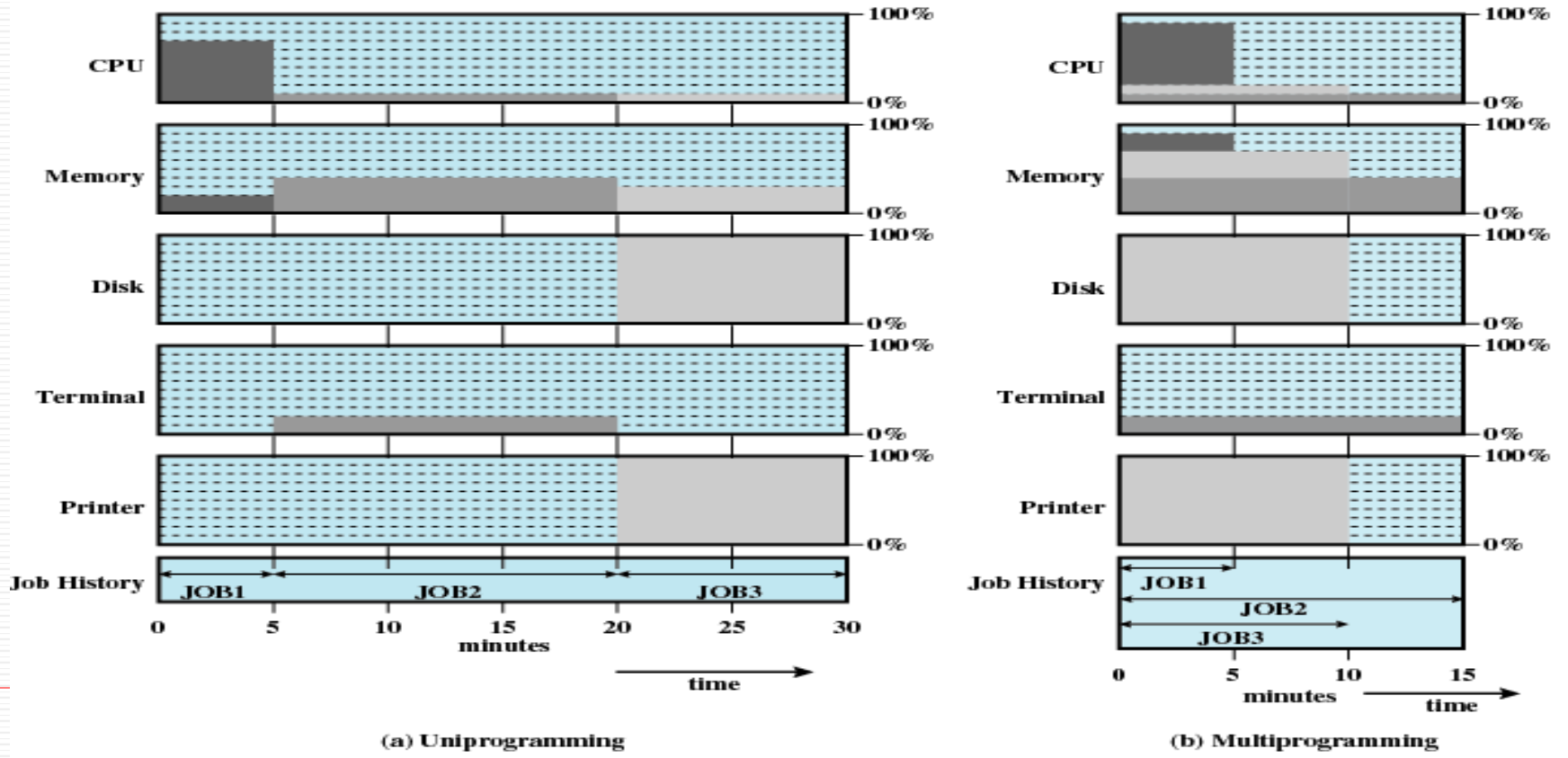


Figure 2.6 Utilization Histograms

2.2.3 Multiprogrammed Batch Systems (6/6)

- Example: 3个作业/程序(cont.)
  - Effects of Multiprogramming on Resource Utilization

Table 2.2 Effects of Multiprogramming on Resource Utilization

	Uniprogramming	Multiprogramming
Processor use	20%	40%
Memory use	33%	67%
Disk use	33%	67%
Printer use	33%	67%
Elapsed time	30 min	15 min
Throughput	6 jobs/hr	12 jobs/hr
Mean response time	18 min	10 min

2.2 The Evolution of Operating Systems

2.2.1 Serial Processing : No Operating Systems

2.2.2 Simple Batch Systems

2.2.3 Multiprogrammed Batch Systems

2.2.4 Time-Sharing Systems

2.2.4 Time-Sharing Systems (1/2)

- Why?
  - Background: 1960s
    - Mainframe Computer主计算机 System:
    - Multiple users simultaneously access the system through terminals
  - Requirement: handle multiple interactive users/jobs
- Idea
  - Processor's time is shared among multiple users/jobs

2.2.4 Time-Sharing Systems (2/2)

- Batch Multiprogramming VS Time Sharing

Table 2.3 Batch Multiprogramming versus Time Sharing

	Batch Multiprogramming	Time Sharing
Principal objective	Maximize processor use	Minimize response time
Source of directives to operating system	Job control language commands provided with the job	Commands entered at the terminal

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

2.3 Major Achievements

2.4 Developments Leading to Modern Operating Systems

2.3 Major Achievements

2.3.1 The Process

2.3.2 Memory Management

2.3.3 Information Protection and Security

2.3.4 Scheduling and Resource Management

2.3.5 System Structure

2.3.1 The Processes (1/2)

- Processes consists of three components
  - An executable program /code
  - Associated data needed by the program
  - Execution context of the program (the core)
    - All information the operating system needs to manage the process
- The process is realized/implemented as a data structure
  - All information the operating system needs to manage the process is stored in the data structure

2.3.1 The Processes (2/2)

- An example

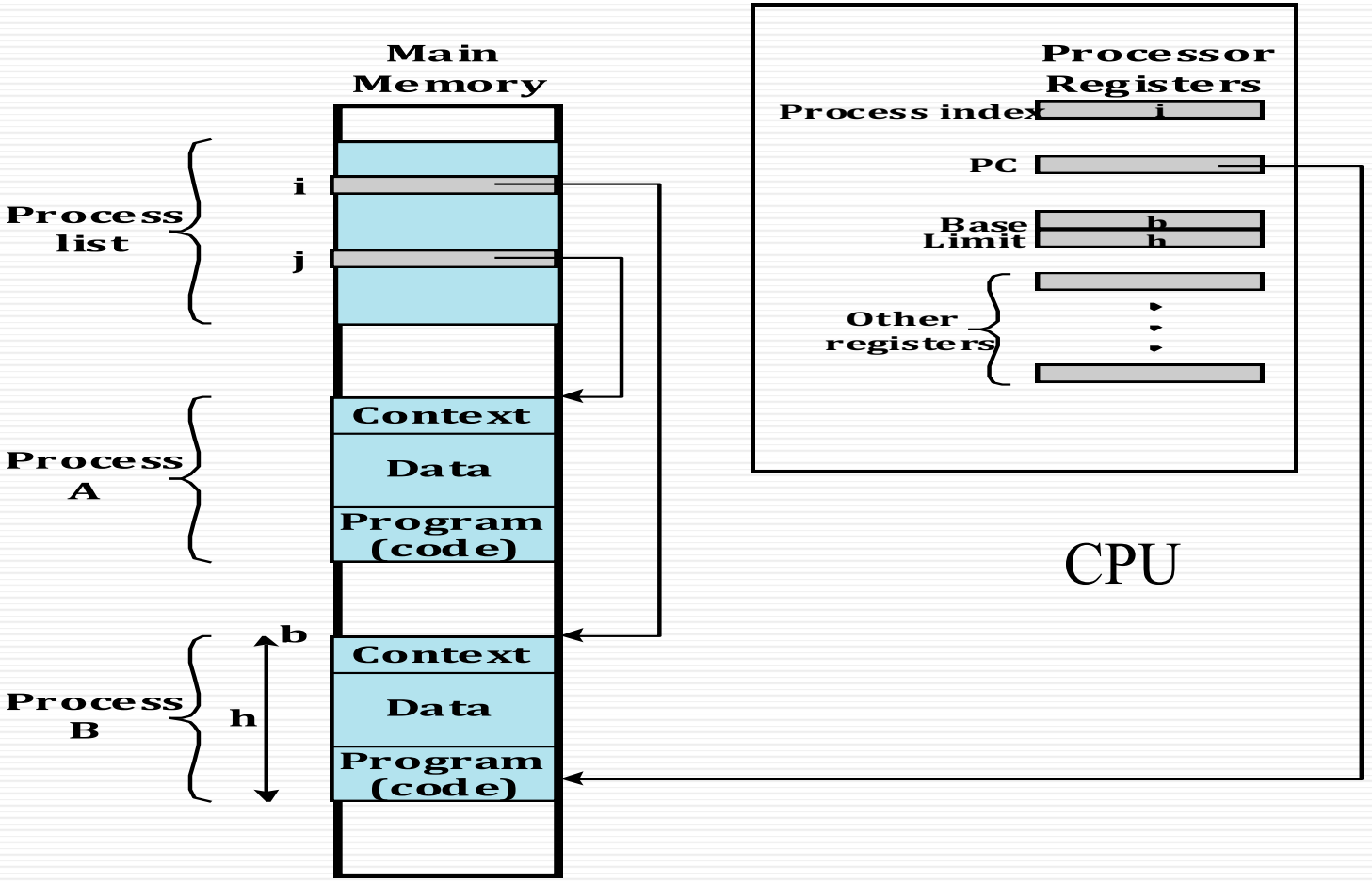


Figure 2.8 Typical Process Implementati

2.3 Major Achievements

2.3.0 Introduction

2.3.1 The Process

2.3.2 Memory Management

2.3.3 Information Protection and Security

2.3.4 Scheduling and Resource Management

2.3.5 System Structure

2.3.2 Memory Management (1/3)

- OS has five storage management responsibilities:
  - Process isolation(进程隔离)
  - Automatic allocation and management(自动分配和管理)
  - Support of modular programming(模块化程序设计)
  - Protection and access control (保护与存取控制)
  - Long-term storage(长期存储)

2.3.2 Memory Management (2/3)

- Virtual Memory / VM
  - Allows programmers to address memory from a logical point of view
  - No hiatus(脱节) between the execution of successive processes while one process was written out to secondary store and the successor process was read in
- Page
  - Allows process to be comprised of a number of fixed-size blocks, called pages
  - Each page may be located any where in main memory

2.3.2 Memory Management (3/3)

- Virtual Memory Addressing
  - Storage system consists of main memory and auxiliary memory

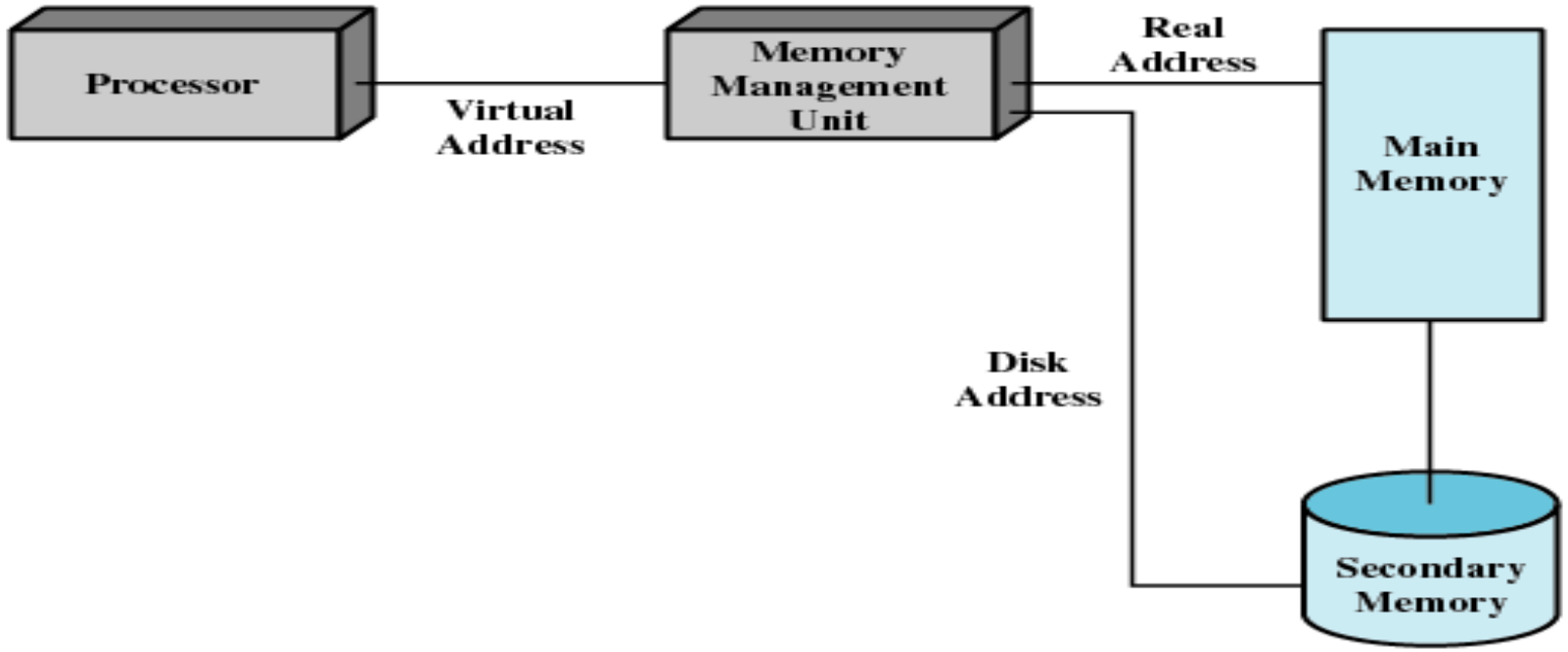


Figure 2.10 Virtual Memory Addressing

2.3 Major Achievements

2.3.0 Introduction

2.3.1 The Process

2.3.2 Memory Management

2.3.3 Information Protection and Security

2.3.4 Scheduling and Resource Management

2.3.5 System Structure

2.3.3 Information Protection and Security(1/1)

- Availability可用性
- Confidentiality保密性
- Data integrity数据完整性
- Authenticity授权性

2.3 Major Achievements

2.3.0 Introduction

2.3.1 The Process

2.3.2 Memory Management

2.3.3 Information Protection and Security

2.3.4 Scheduling and Resource Management

2.3.5 System Structure

2.3.4 Scheduling and Resource Management (1/2)

- Manage the resources (such as CPU, Main Memory, I/O devices)
- Schedule/调度 these resources to various active processes/进程

2.3.4 Scheduling and Resource Management (2/2)

- The factors that should be considered
  - Fairness(公平性)
    - Give equal and fair access to resources
  - Differential responsiveness(有差别的响应性)
    - Discriminate among different classes of processes
  - Efficiency(高效性)
    - Maximize throughput(吞吐量), minimize response time(响应时间), and accommodate as many users as possible

2.3 Major Achievements

2.3.0 Introduction

2.3.1 The Process

2.3.2 Memory Management

2.3.3 Information Protection and Security

2.3.4 Scheduling and Resource Management

2.3.5 System Structure

2.3.5 System Structure (1/4)

- The size and complexity of operating systems has grown.
  - CTSS: 32,000 36-bit words
  - OS/360: more than a million machine instructions
  - Multics: more than 20 million instructions.
  - Windows
    - Windows NT 4.0: 16 million lines of code
    - Windows 2000: more than 32 million lines of code
    - Windows XP: 35 million lines of code
    - Windows Vista: 50 million lines of code
  - Linux 2.6.27: 10 million lines of code

2.3.5 System Structure (2/4)

- Led to four unfortunate but all-too-common problems
  - chronically late习惯性落后in being delivered
  - performance is often not what was expected
  - have latent bugs
  - it has proved impossible to deploy a complex OS that is not vulnerable不易to a variety of security attacks
- Solution：
  - The software must be modular and modules must have well-defined interfaces
  - Levels

2.3.5 System Structure (3/4)

- 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
  - 效果：This decomposes a problem into a number of more manageable subproblems

2.3.5 S

Table 2.4 Operating System Design Hierarchy

Level	Name	Objects	Example Operations
13	Shell	User programming environment	Statements in shell language
12	User processes	User processes	Quit, kill, suspend, resume
11	Directories	Directories	Create, destroy, attach, detach, search, list
10	Devices	External devices, such as printers, displays, and keyboards	Open, close, read, write
9	File system	Files	Create, destroy, open, close, read, write
8	Communications	Pipes	Create, destroy, open, close, read, write
7	Virtual memory	Segments, pages	Read, write, fetch
6	Local secondary store	Blocks of data, device channels	Read, write, allocate, free
5	Primitive processes	Primitive processes, semaphores, ready list	Suspend, resume, wait, signal
4	Interrupts	Interrupt-handling programs	Invoke, mask, unmask, retry
3	Procedures	Procedures, call stack, display	Mark stack, call, return
2	Instruction set	Evaluation stack, microprogram interpreter, scalar and array data	Load, store, add, subtract, branch
1	Electronic circuits	Registers, gates, buses, etc.	Clear, transfer, activate, complement

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

2.3 Major Achievements

2.4 Developments Leading to Modern Operating Systems

2.4 Modern Operating Systems (1/8)

- Background
  - new developments in hardware
  - new applications
  - new security threats
- Five features are mentioned in this chapter
  - Microkernel architecture : more on chapter 4
  - Multithreading : more on chapter 4
  - Symmetric multiprocessing (SMP) : more on chapter 4
  - Distributed operating systems : chapter 14~15: 不讲解
  - Object-oriented design

2.4 Modern Operating Systems (2/8)

- Microkernel(微内核) architecture
  - VS monolithic kernel(单体内核) architecture
  - Assigns only a few essential functions to the kernel
    - Address spaces地址空间
    - Interprocess communication (IPC)进程间通信
    - Basic scheduling基础调度

2.4 Modern Operating Systems (3/8)

- 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 and other resources

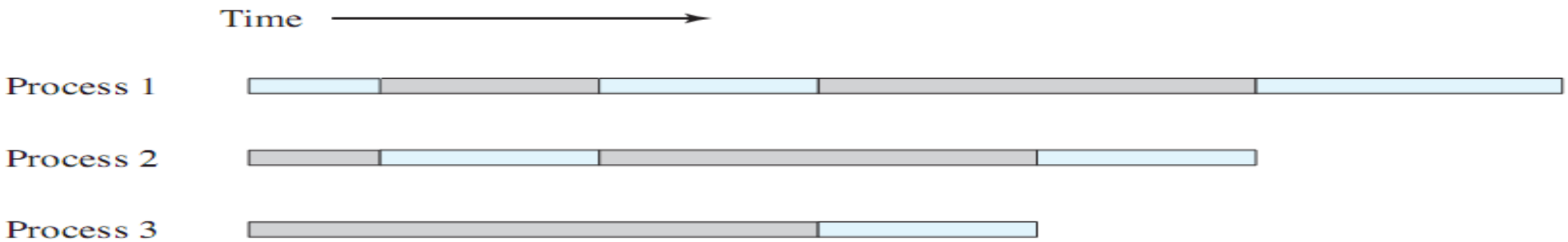
2.4 Modern Operating Systems (4/8)

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

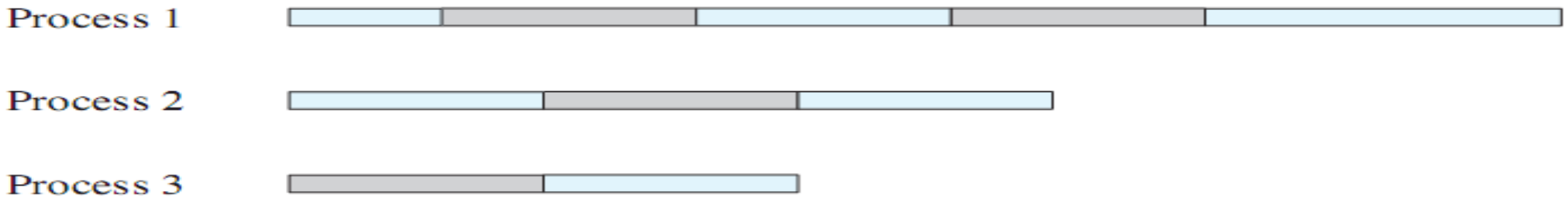
2.4 Modern Operating Systems (5/8)

- Symmetric multiprocessing (SMP)
  - The OS of an SMP schedules processes or threads across all of the processors.
  - SMP has a number of potential advantages over uniprocessor单处理机 architecture, including the following:
    - Performance 性能
    - Availability 可用性
    - Incremental growth 可增长
    - Scaling可扩展，多型号

2.4 Modern Operating Systems (6/8)



(a) Interleaving (multiprogramming, one processor)



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

2.4 Modern Operating Systems (7/8)

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

2.4 Modern Operating Systems (8/8)

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