Chapter 1 (Part 2)
Introduction to Operating System

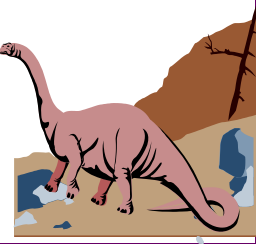
张竞慧

办公室：计算机楼366室

电邮：jhzhang@seu.edu.cn

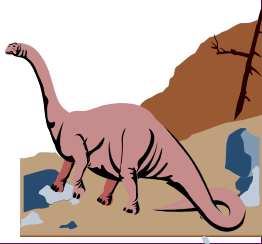
主页：http://cse.seu.edu.cn/PersonalPage/zjh/

电话：025-52091017

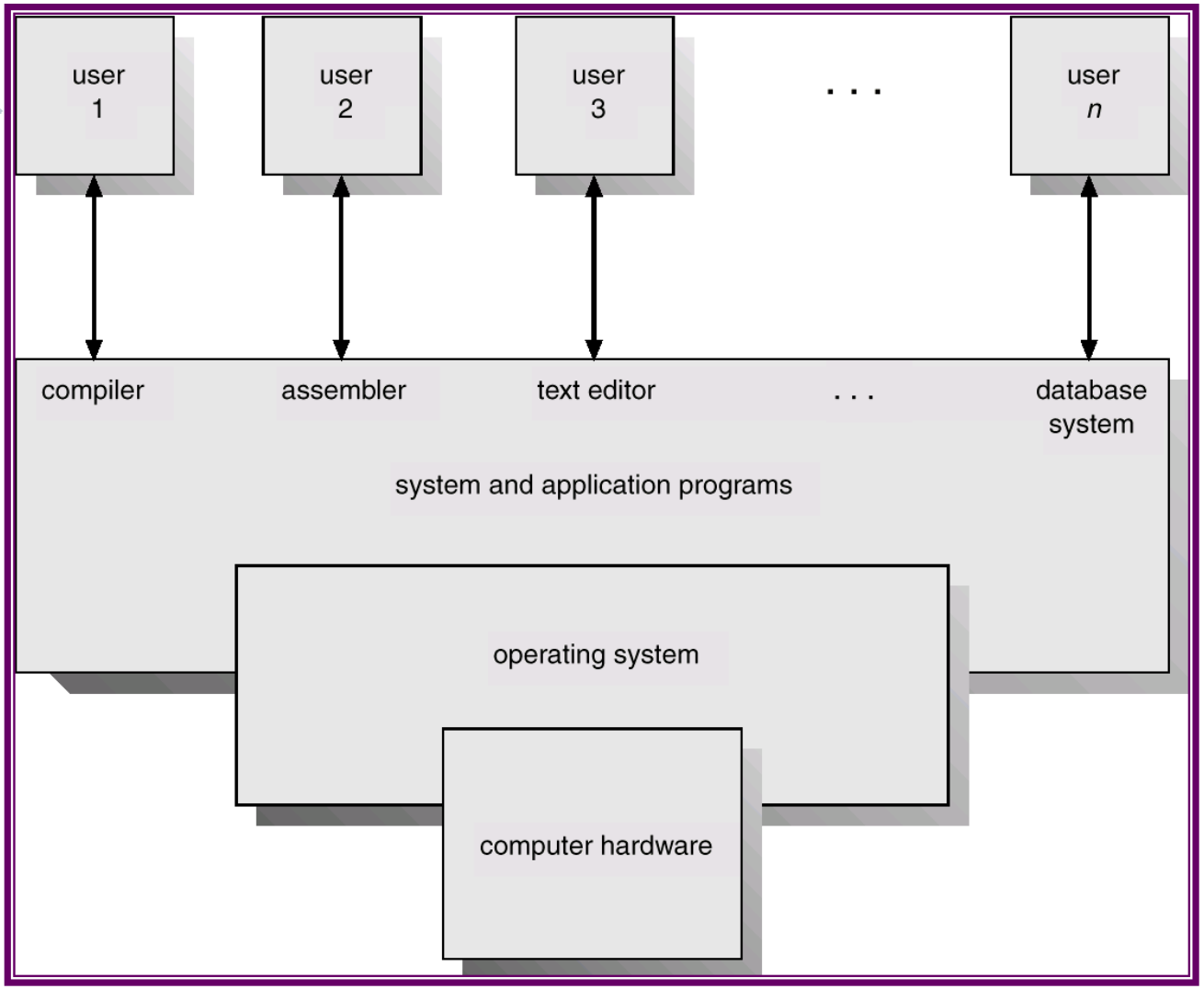


Computer System Components

1. Hardware – provides basic computing resources (CPU, memory, I/O devices).
2. Operating system – controls and coordinates the use of the hardware among the various application programs for the various users.
3. Applications programs – (compilers, database systems, video games, business programs) are used to solve user problems.
4. Users (people, machines, other computers).

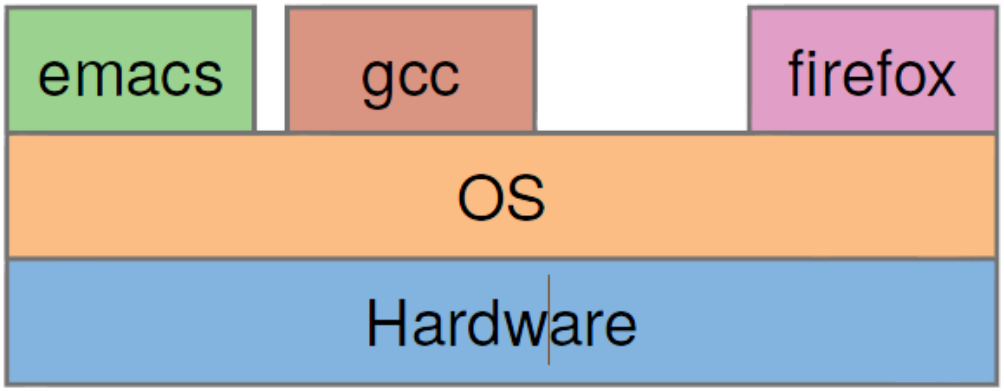


Abstract View of Computer System Components



OS from User View(1/2)

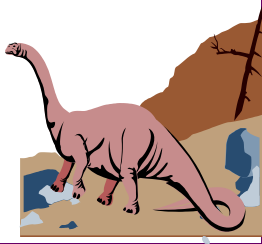
- Layer between applications and hardware



- Makes hardware useful to the programmer
- Provides abstractions for applications
  - Manages and hides details of hardware
  - Accesses hardware through low/level interfaces unavailable to applications
- Provides protection
  - Prevents one process/user from clobbering another

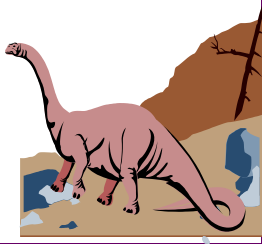
OS from User View(2/2)

- PC Users: an operating system is a program that provides an easy-to-use interface for using the hardware
- Mainframe/Minicomputer Users: an operating system is a program that helps maximize the system resource utilization
- Workstation Users: an operating system is a program designed to compromise between individual usability and resource utilization



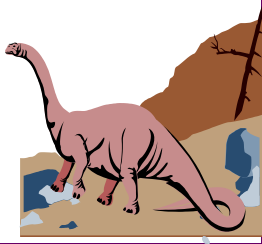
What is an Operating System?

- A program that acts as an intermediary between the application programs and the computer hardware.
  - Execute user programs and make solving user problems easier.
  - Make the computer system convenient to use.
  - Use the computer hardware in an efficient manner.
- From the computer's point of view:
  - A resource allocator, a control program, a kernel



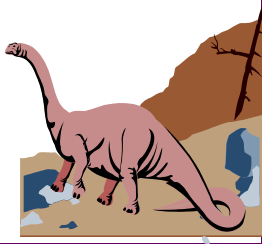
OS from System View(1/2)

- *A Resource allocator*: the operating system allocates and reclaims the system hardware resources to and from user programs
- *A Control Program*: the operating system controls the execution of user programs to prevent errors and improper use of the computer



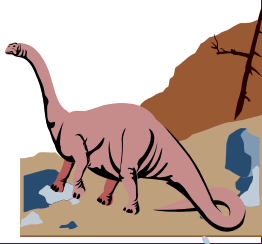
OS from System View(2/2)

- There is *no* universal definition of what is an operating system.
- A common definition is that the operating system is the one program running at all times on the computer (*i.e.*, the *kernel*). All other programs are application programs



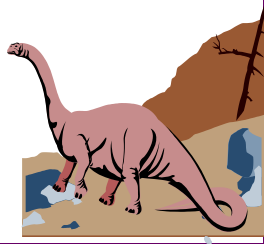
The Goals of an Operating System

- Primary Goal: *efficient* operation of the computer system.
- In some systems, it is *convenience*.
- In the past, efficiency is far more important than convenience.



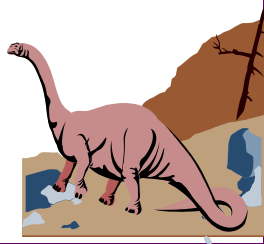
Evolution of Operating Systems

- It may be easier to understand the key requirements of an OS by considering the evolution of Operating Systems
  - Simple Batch Systems
  - Multiprogrammed batch systems
  - Time Sharing Systems



Simple batch system

- Early computers were extremely expensive
  - Important to maximize processor utilization
- Monitor
  - Software that controls the sequence of events
  - Batch jobs together
  - Program returns control to monitor when finished



Monitor's perspective

- *Resident Monitor* is software always in memory
- Monitor controls the sequence of events
- Monitor reads in job and gives control
- Job returns control to monitor

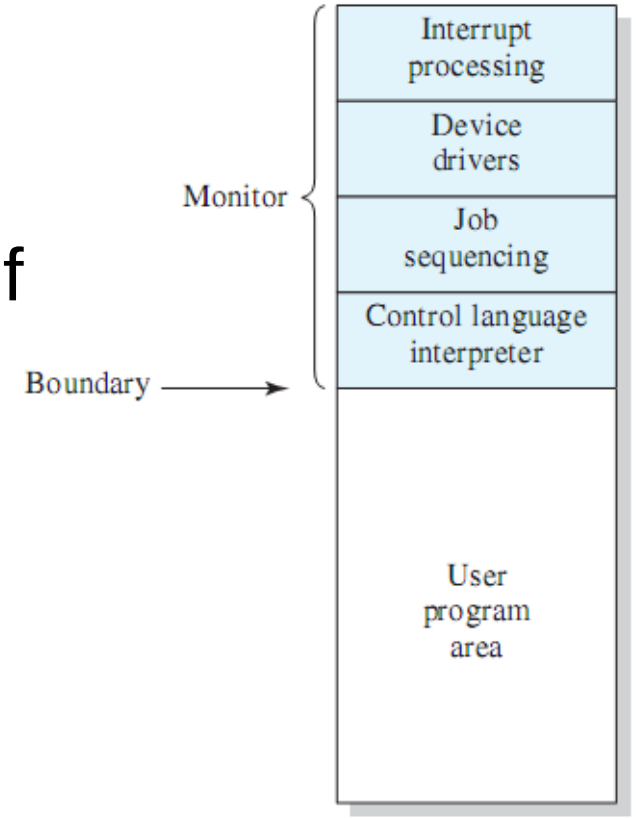
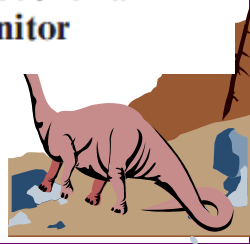
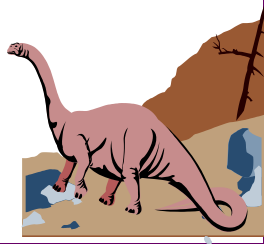


Figure 2.3 **Memory Layout for a Resident Monitor**



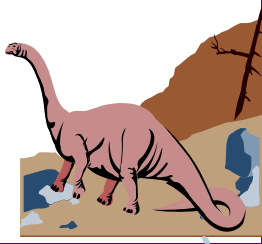
Desirable Hardware Features

- Memory protection for monitor
  - Jobs cannot overwrite or alter
- Timer
  - Prevent a job from monopolizing system
- Privileged instructions
  - Only executed by the monitor



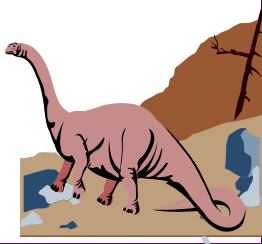
Modes of Operation

- User Mode
  - User program executes in user mode
  - Certain areas of memory protected from user access
  - Certain instructions may not be executed
- Kernel Mode
  - Monitor executes in kernel mode
  - Privileged instructions may be executed, all memory accessible



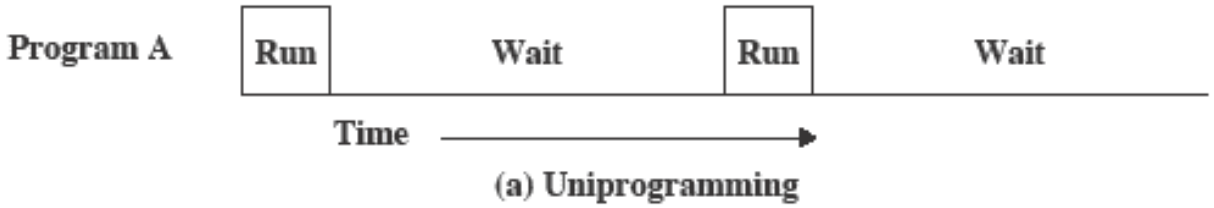
Multiprogrammed Batch Systems

- CPU is often idle
  - Even with automatic job sequencing.
  - I/O devices are slow compared to processor

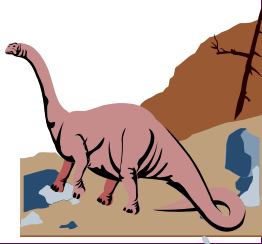


Uniprogramming

- Processor must wait for I/O instruction to complete before preceding

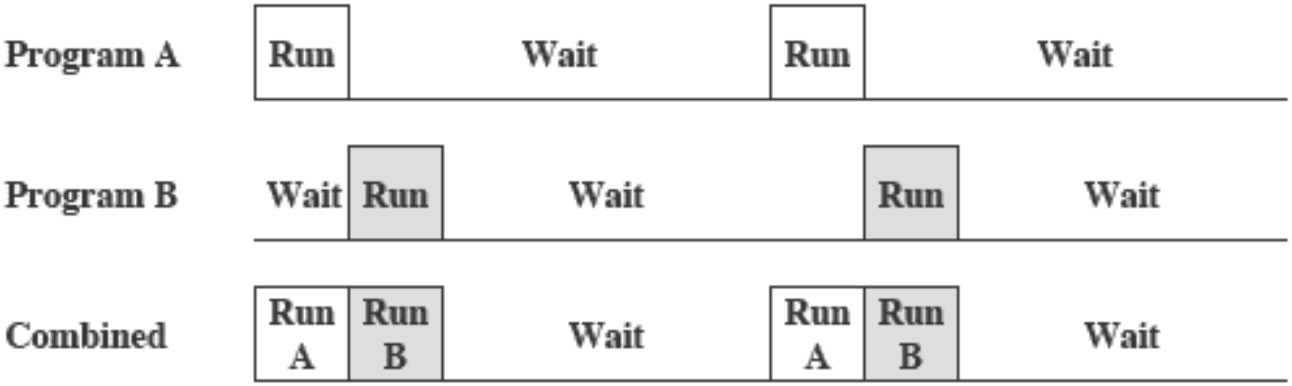


(a) Uniprogramming

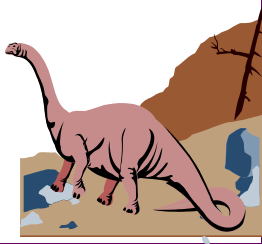


Multiprogramming

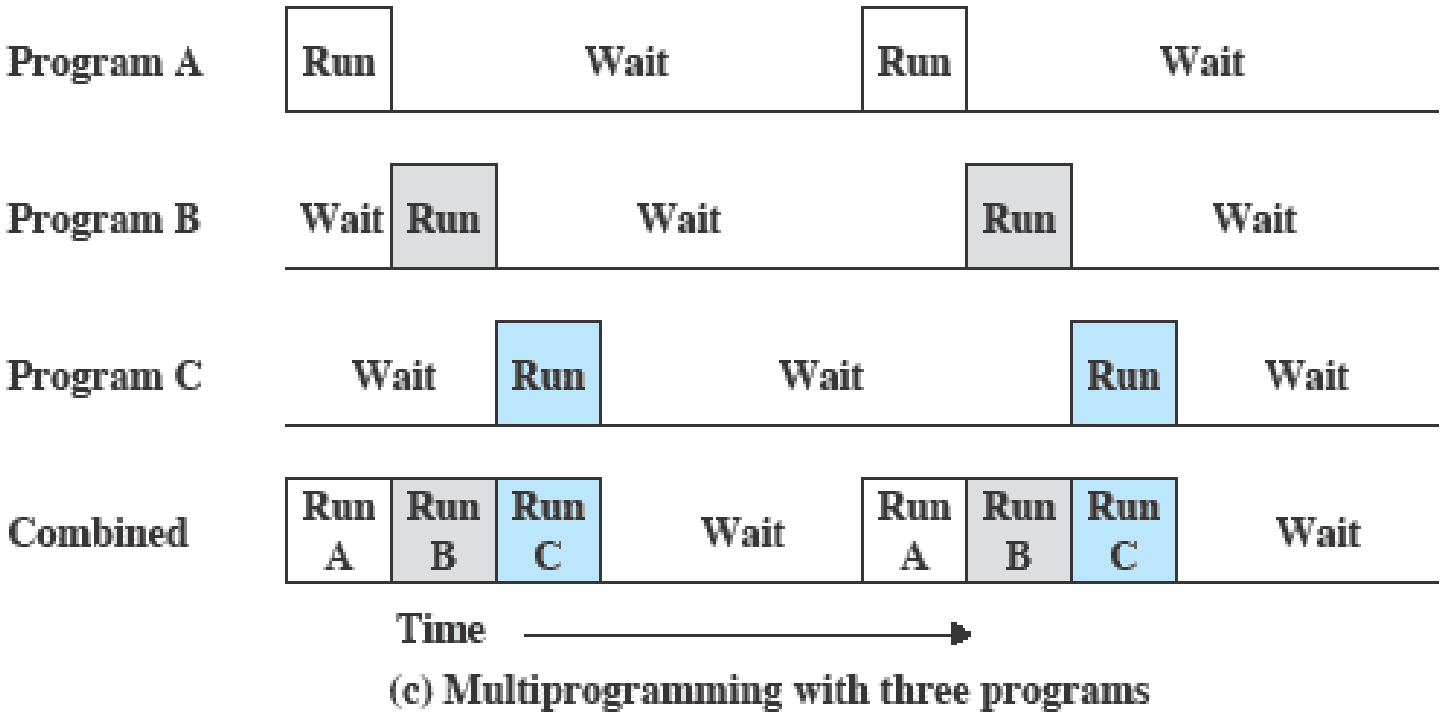
- When one job needs to wait for I/O, the processor can switch to the other job



(b) Multiprogramming with two programs



Multiprogramming

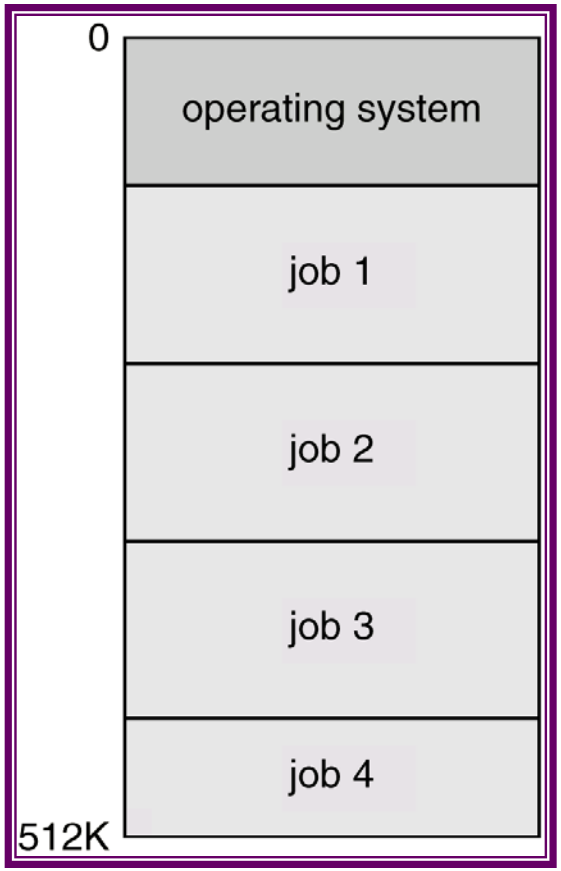


(c) Multiprogramming with three programs

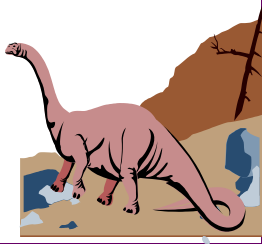
Multiprogrammed Batch Systems

Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.

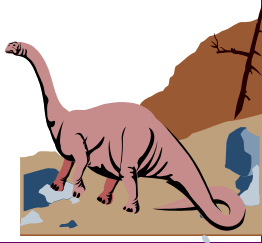
- **Keeps serval jobs in memory simultaneously**
- **Picks and begins to execute one**
- **If that job needs to wait, CPU is switched to another one**

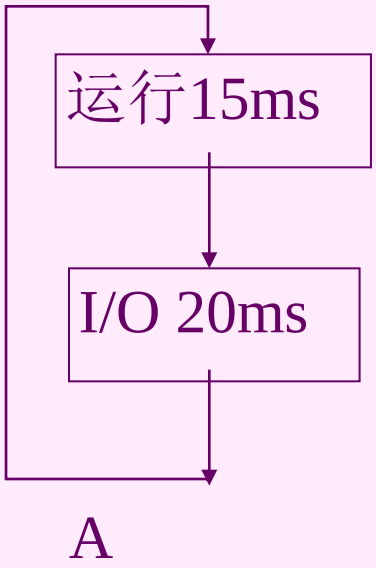


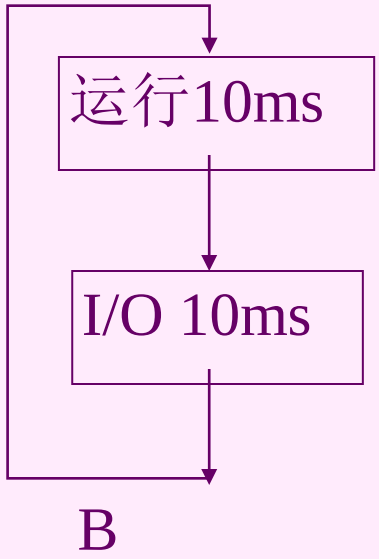
It's the first instance where the operating system must make decisions for the users

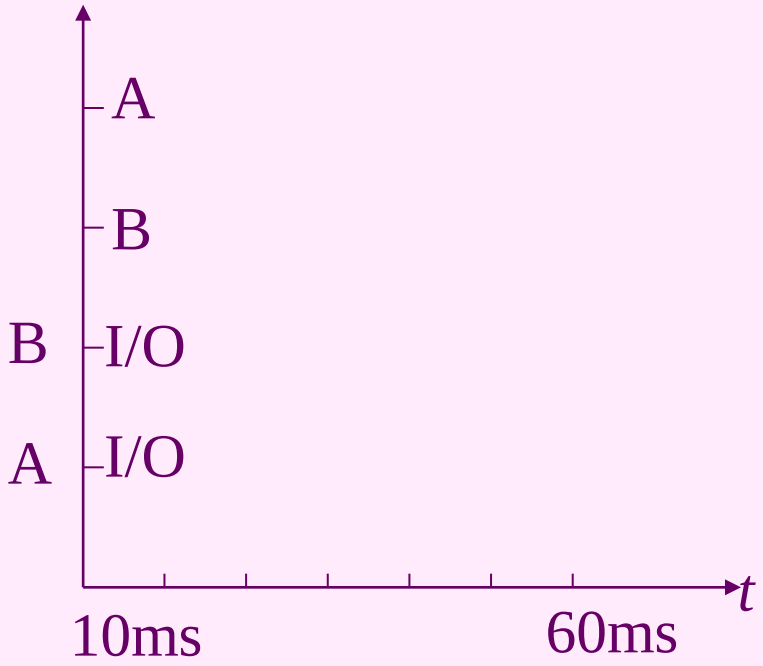


例：有两道程序A、B，按下图以多道程序方式运行，要求在右图画出它们的运行轨迹，并计算在60ms内，CPU的利用率，假设起始时首先运行B，并允许忽略监督程序切换A、B的时间。

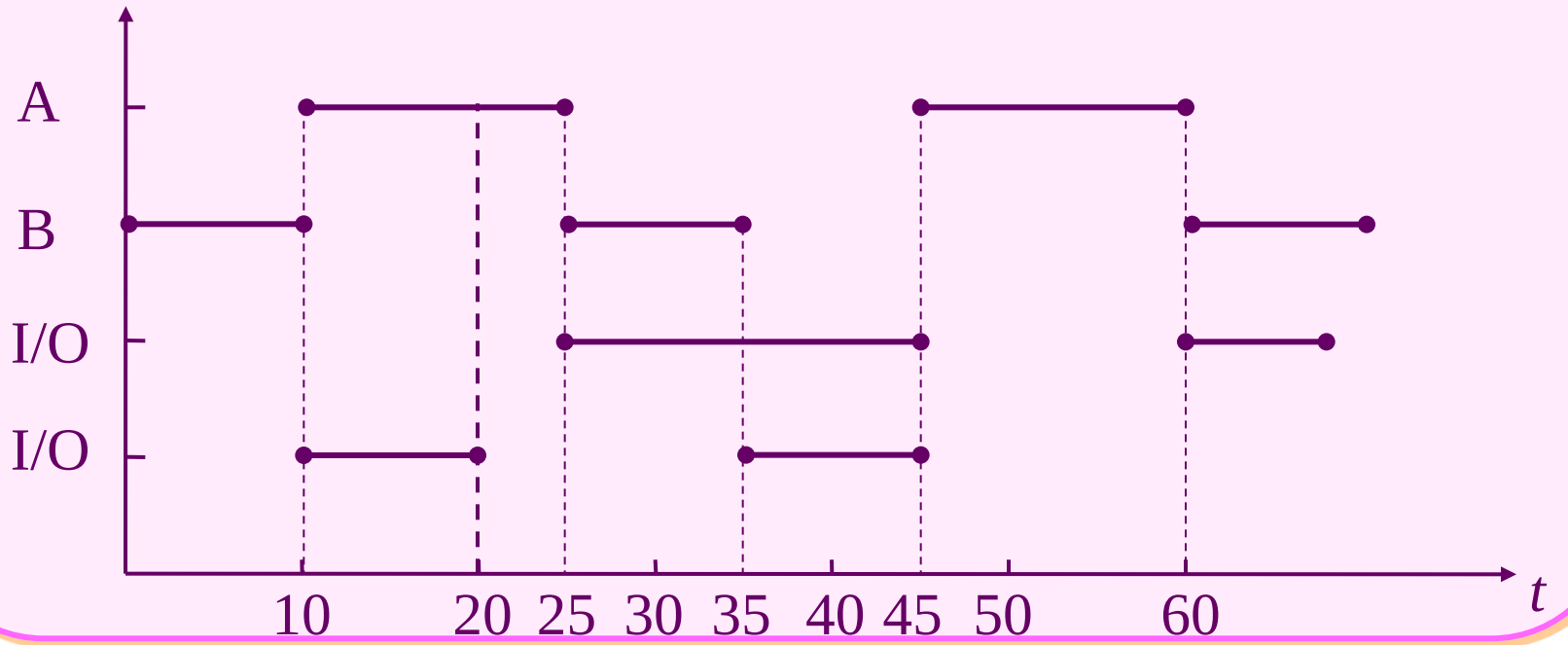



运行15ms
I/O 20ms
A


运行10ms
I/O 10ms
B

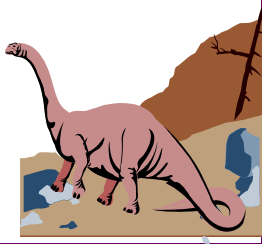

A
B
I/O
I/O
B
A
t
10ms
60ms

解：



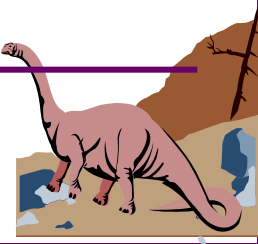
$$P_{\text{CPU}} = \frac{60-(45-35)}{60} \times 100\% = \frac{50}{60} \times 100\% = 83.3\%$$

为了说明多道程序的优点，不妨参考R Turner 提出的例子：某计算机系统，有256KB的主存(不包含操作系统)，一个磁盘，一个终端和一台打印机。同时提交的三个作业分别命名为JOB1、JOB2、JOB3。各作业运行时间分别为5min、15min和10min。它们对资源的使用情况如下表所示：

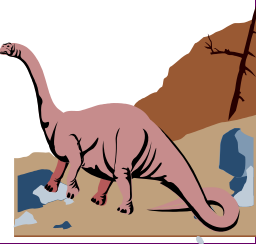


三个作业的执行要求

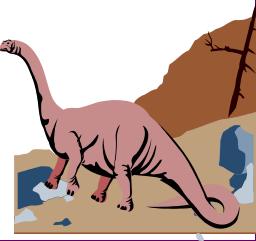
作业名	JOB1	JOB2	JOB3
作业类型	CPU型	I/O型	I/O型
所需主存/KB	50	100	80
所需磁盘	不用	不用	需要
所需终端	不用	需要	不用
所需打印机	不用	不用	需要
运行时间/min	5	15	10



假定JOB1主要使用CPU处理数据，JOB2主要使用终端进行作业的输入，JOB3运行时主要使用磁盘和打印机，后两作业都只需要较少的CPU时间。对于简单批处理情况，这些作业将按顺序执行。JOB1运行5min完成，JOB2在等待5min后，运行15min完成，JOB3在等待20min后开始执行。三个作业全部完成需要30min(这三个作业是一批)。

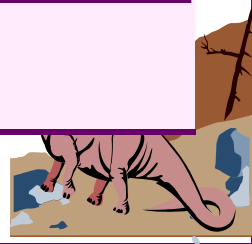


采用多道程序设计技术，可让这三个作业并行运行。由于它们运行中几乎不同时使用同一资源，所以三个作业可同时运行。JOB1在进行数据处理的同时，JOB2在终端上进行作业输入，JOB3在使用磁盘和打印机。因此，JOB1只需5min完成，JOB2需15min完成，JOB3需10min完成。这样三个作业全部完成的时间只需15min，显然系统处理效率明显提高。



多道程序与单道程序的平均资源利用率

	单　道	多道 (三道作业)
CPU利用率		
主存利用率		
磁盘利用率		
打印机利用率		
全部作业完成时间/min		
吞吐量/(作业 $\cdot h^{-1}$)		
平均周转时间/min		



Utilization Histograms

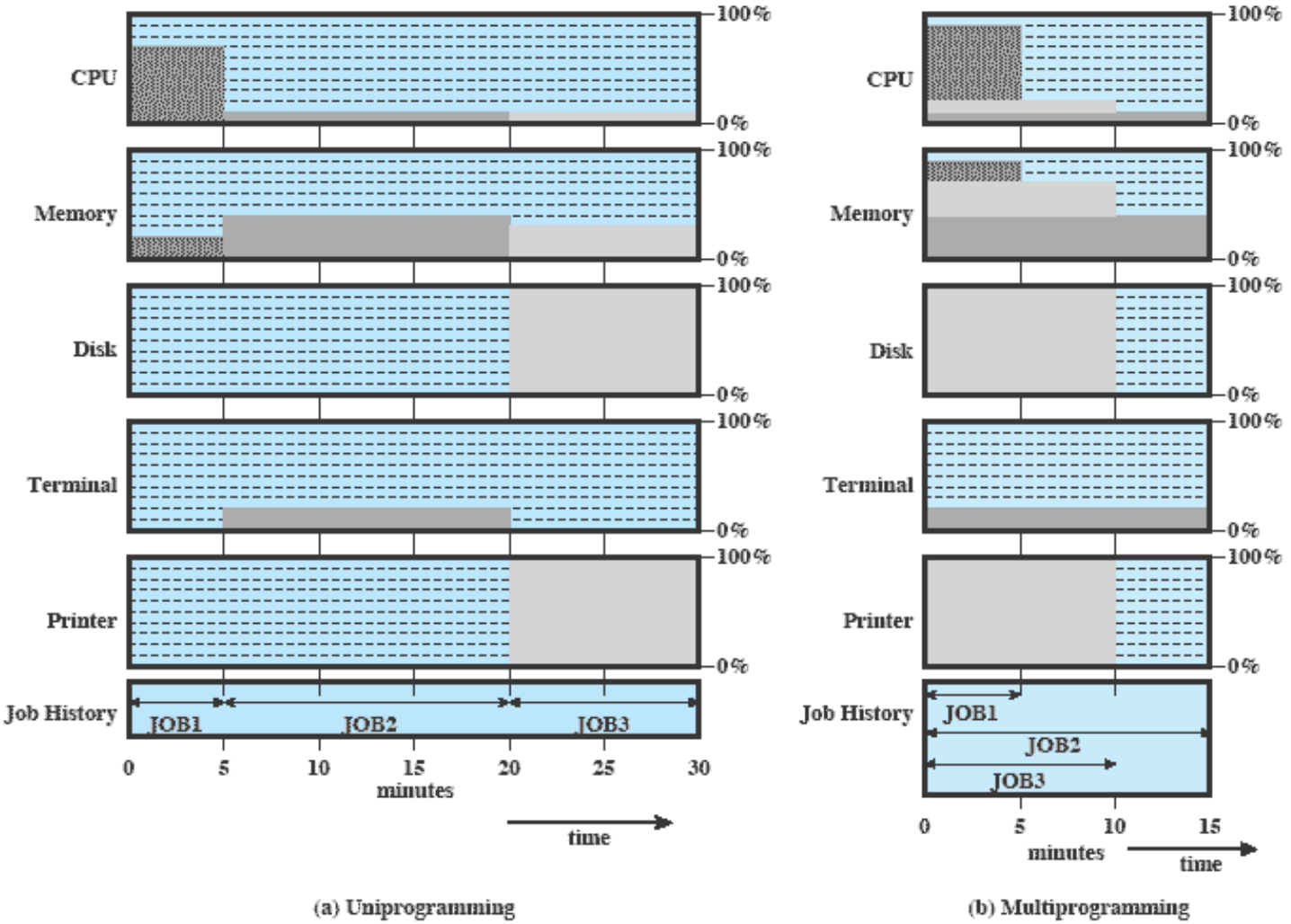


Figure 2.6 Utilization Histograms

Advantages and disadvantages of multiprogramming:

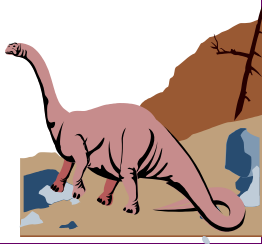
- CPU 利用率大大提高

用户无控制权，无交互性，延迟大

引入多道程序设计技术的根本目的：

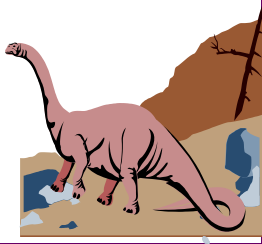
提高CPU的利用率，充分发挥并行性，

这包括：程序之间； 设备之间；设备与CPU之间均并行工作。



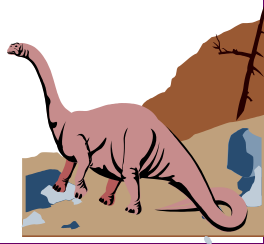
OS Features Needed for Multiprogramming

- **Memory management** – the system must allocate the memory to several jobs.
- **CPU scheduling** – the system must choose among several jobs ready to run.
- **Job scheduling** – the system must choose among jobs ready to be brought into memory
- I/O routine supplied by the system.
- Allocation of devices.



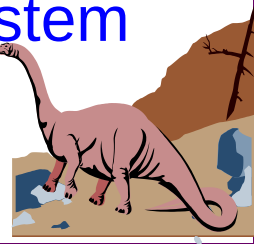
Time Sharing Systems

- Using multiprogramming to handle multiple interactive jobs
- Processor’s time is shared among multiple users
- Multiple users simultaneously access the system through terminals



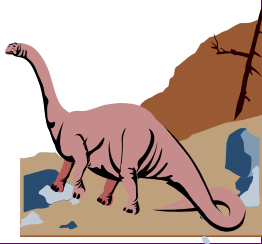
Time-Sharing Systems–Interactive Computing(1/2)

- Interactive computer system provides direct communication between user and the system
  - The user gives instructions to the system directly, waits for immediately results. Response time should be short.
  - when the operating system finishes the execution of one command, it seeks the next “control statement” from the user’s keyboard.
- Time-sharing operating system allows many users to share the computer simulataneously
  - Switches rapidly from one user to the next
  - Gives users the impression: the entire computer system is dedicated for my use.



Time-Sharing Systems–Interactive Computing(2/2)

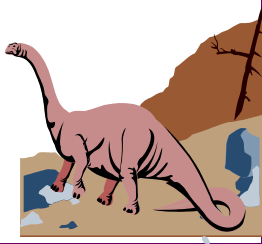
- The CPU is multiplexed among several jobs that are kept in memory and on disk
- To obtain a reasonable response time, a job needs to be swapped in and out of memory to the disk (virtual memory).
- Disk management must be provided
- Job synchronization mechanisms are needed
- It may ensure that jobs do not get stuck in a deadlock



Batch Multiprogramming vs. Time Sharing

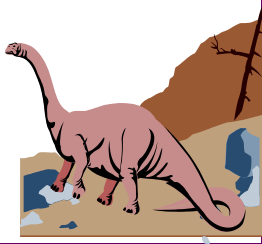
Table 2.3 Batch Multiprogramming versus Time Sharing

	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



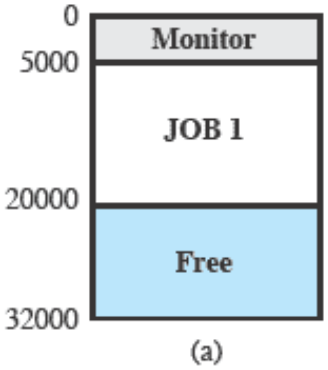
Early Example: CTSS

- Compatible Time-Sharing System (CTSS)
  - Developed at MIT as project MAC
- Time Slicing:
  - When control was passed to a user
  - User program and data loaded
  - Clock generates interrupts about every 0.2 sec
  - At each interrupt OS gained control and could assign processor to another user

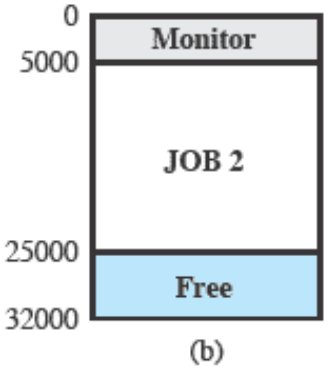


CTSS Operation

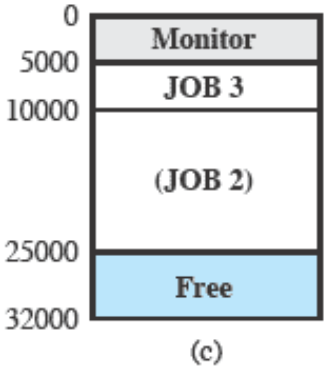
- Job is always loaded at address 5000 to simplify monitor and memory management
- To reduce disk io, job memory is only written back to disk when memory remained is not enough for incoming job



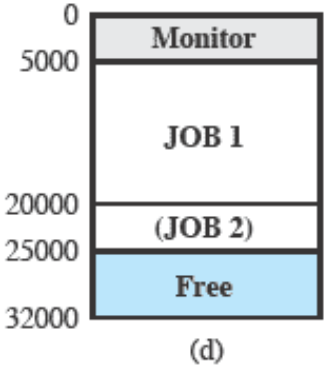
(a)



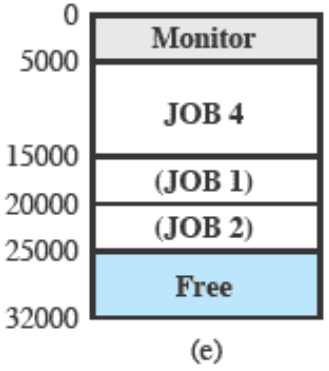
(b)



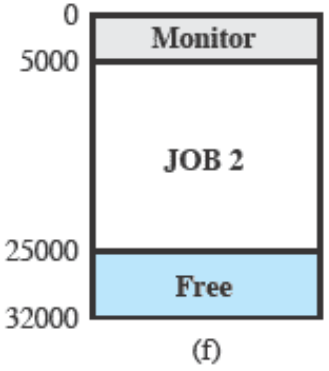
(c)



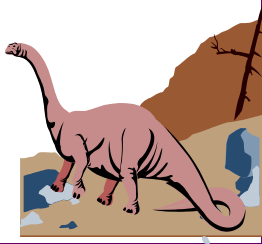
(d)



(e)

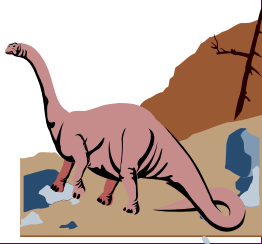


(f)

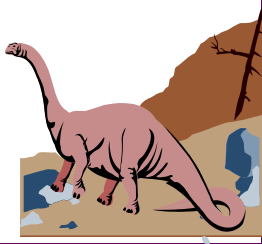


Problems and Issues

- Multiple jobs in memory must be protected from each other's data
- File system must be protected so that only authorised users can access
- Contention for resources must be handled
  - Printers, storage etc

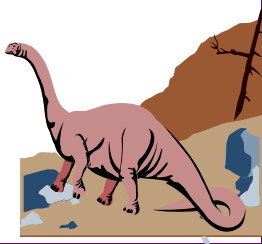


作业：从技术发展的角度，简要介绍批处理系统、多道程序系统和分时系统各自的技术特性。（在该技术出现以前系统存在哪些问题，该技术是如何解决这些问题的）



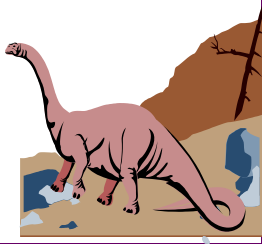
Desktop Systems(1/2)

- *Personal computers* – computer system dedicated to a single user.
- I/O devices – keyboards, mouse, display screens, small printers.
- User convenience and responsiveness.



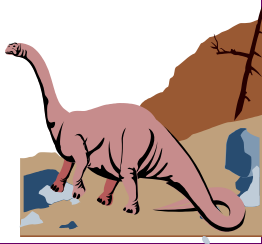
Desktop Systems(2/2)

- Can adopt technology developed for larger operating system
- individuals have sole use of computer and do not need advanced CPU utilization of protection features.
- May run several different types of operating systems (Windows, MacOS, UNIX, Linux)



Parallel Systems

- Multiprocessor systems with more than one CPU in close communication.
- *Tightly coupled system* – processors share memory and a clock; communication usually takes place through the shared memory.

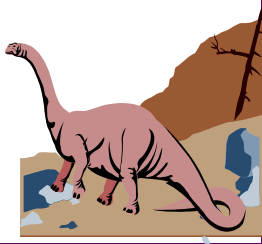


Parallel Systems (Cont.)

- Advantages of parallel system:
  - Increased throughput: Gets more jobs done
  - Economy of scale: Because of resource

sharing, multiprocessor systems is cheaper than multiple single processor systems

  - Increased reliability: the failure of one processor will not halt the whole system
    - ✓ graceful degradation
    - ✓ Fault-tolerant systems

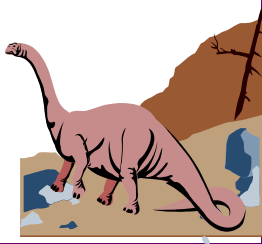


Parallel Systems (Cont.)

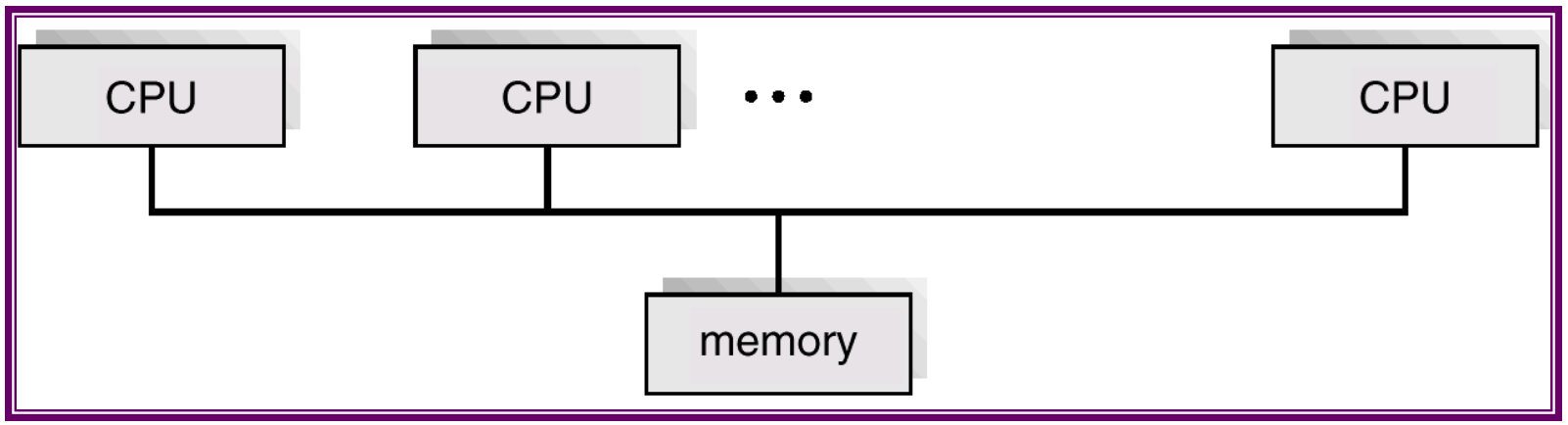
- *Symmetric multiprocessing (SMP)*
  - Each processor runs an identical copy of the operating system.
  - Many processes can run at once without performance deterioration.
  - Must carefully control I/O to ensure that the data reach the appropriate processor
  - May result in inefficiencies
  - Most modern operating systems support SMP

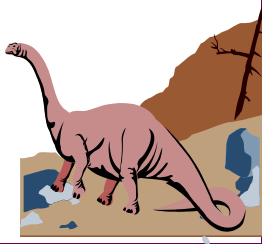
We Have Example !

On SEU HPC Cluster: cat /proc/cpuinfo



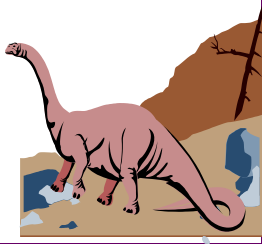
Symmetric Multiprocessing Architecture





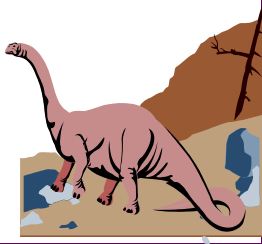
Parallel Systems (Cont.)

- *Asymmetric multiprocessing*
  - Each processor is assigned a specific task.
  - A master processor controls the system. The other processors either look to the master for instructions or have predefined tasks.
  - Thus, this defines a master-slave relationship
  - master processor schedules and allocated work to slave processors.
  - More common in extremely large systems



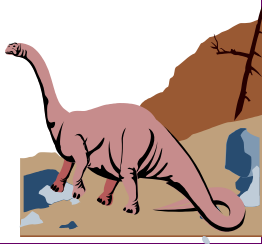
Distributed Systems

- Distribute the computation among several physical processors.
- *Loosely coupled system* – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or telephone lines.



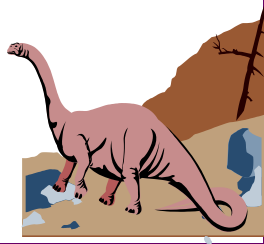
Distributed Systems (cont)

- A distributed system is a collection of physically *separate*, possible *heterogeneous* computer systems that are *networked* to provide the users with access to the various resources that the system maintains
- Advantages of distributed systems.
  - Resources Sharing
  - Computation speed up – load sharing
  - Reliability
  - Communications

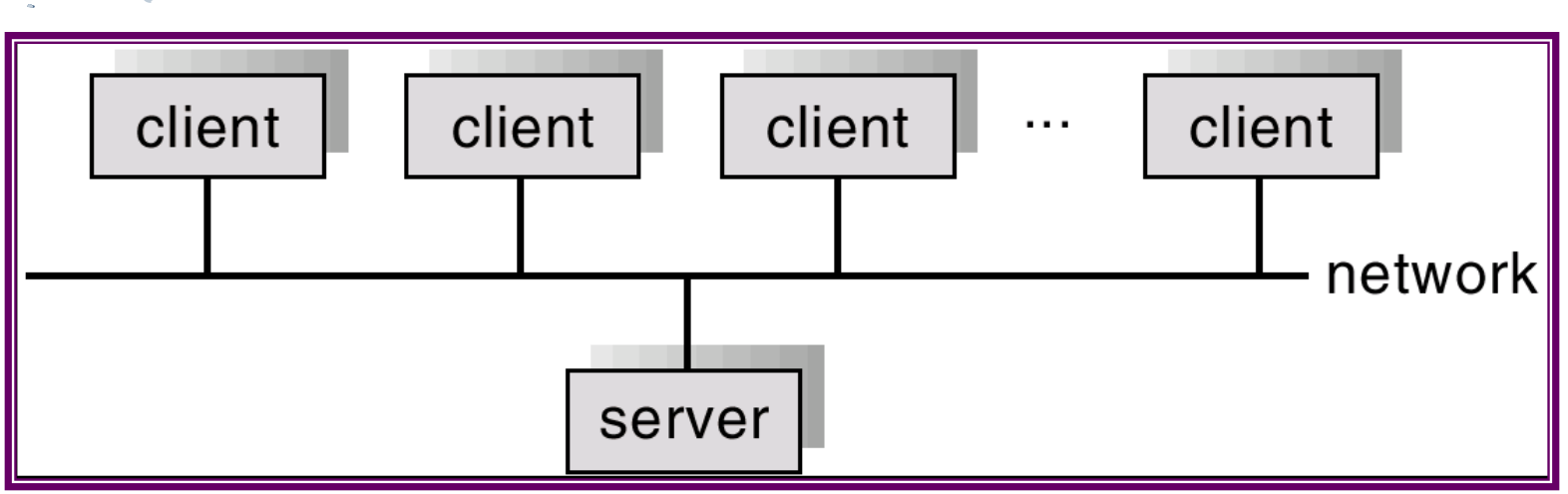


Distributed Systems (cont)

- Requires networking infrastructure.
- Local area networks (LAN) or Wide area networks (WAN)
- May be either client-server or peer-to-peer systems.



General Structure of Client-Server



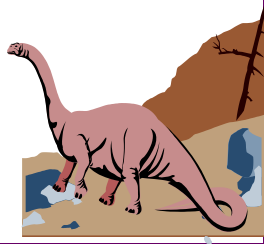
Clustered Systems

- A clustered system has two or more individual systems shared storage and closely linked via LAN networking
- The key of clustered system is high availability.
- Asymmetric Clustering:
  - One machine is in hot-standby mode while others are running applications.
  - The hot-standby machine (*i.e.*, does nothing but) monitors other machines and becomes active if one server fails.
- Symmetric Clustering:
  - Two or more hosts are running applications and monitor each other.
  - This is more efficient as it uses all available hardware

Clustered Systems

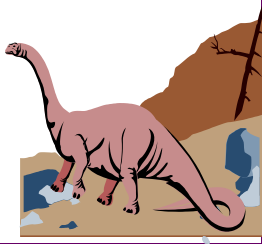
We Have Example !

On SEU HPC Cluster: bsub –q seu_cse –I hostname



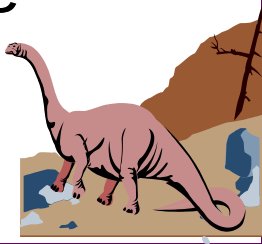
Real-Time Systems

- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.



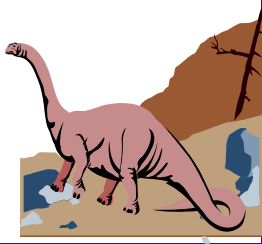
Real-Time Systems (Cont.)

- A real-time operating system has well-defined, fixed time constraints.
- Processing *must* be done within the defined constraints, or the system will fail.
- Real-Time systems may be either *hard* or *soft* real-time.
  - *Hard Real-Time Systems* guarantee that critical tasks be completed on time.
  - *Soft Real-Time Systems* prioritize critical tasks. That is, a critical task get priority over other tasks, and retains that priority until it completes.
- Embedded systems almost always run real-time operating systems

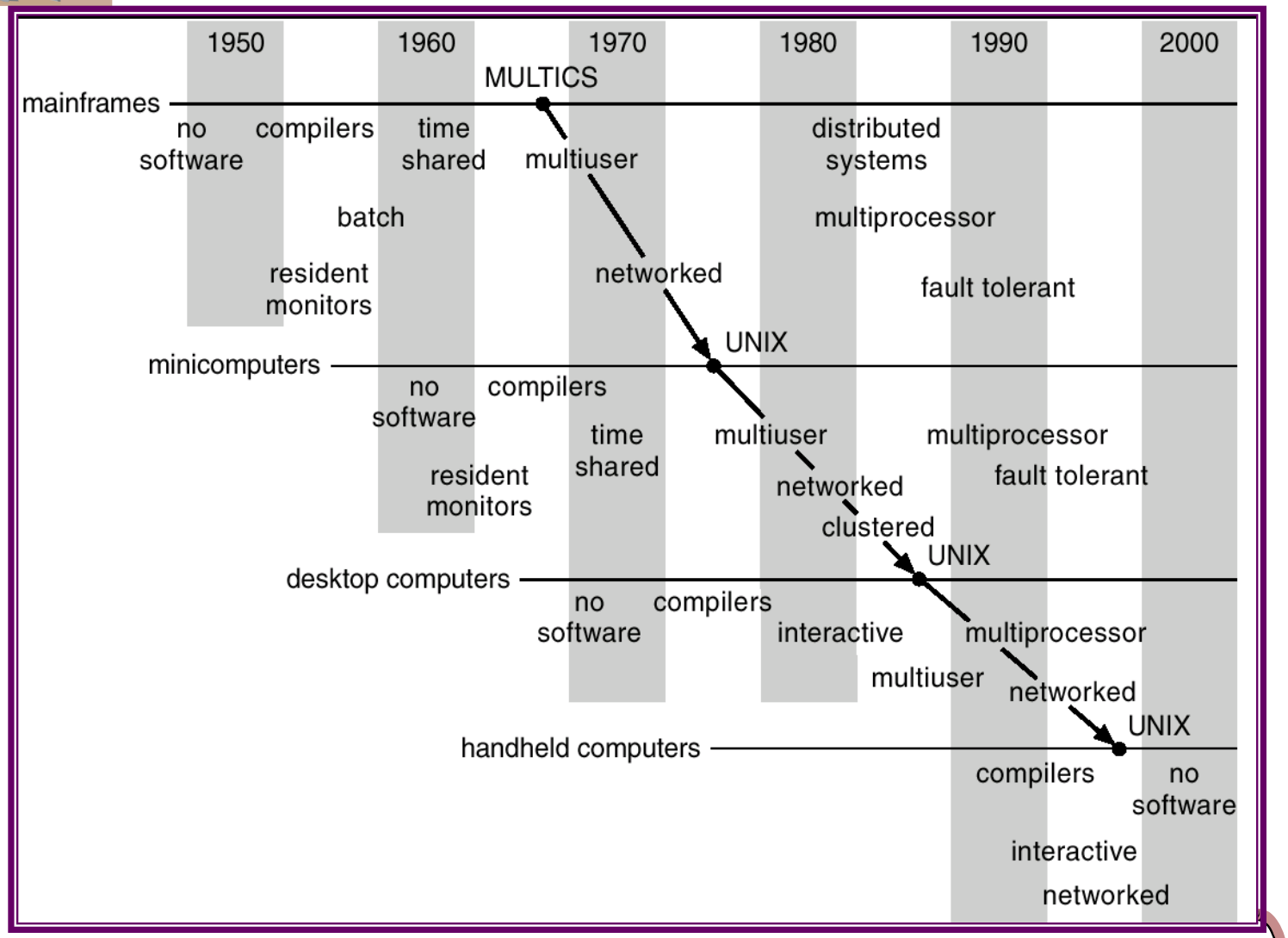


Handheld Systems

- Personal Digital Assistants (PDAs)
- Cellular telephones
- Issues:
  - Limited memory
  - Slow processors
  - Small display screens.



Migration of Operating-System Concepts and Features



Computing Environments

- Traditional computing
- Web-Based Computing
- Embedded Computing
- Pervasive Computing
- Cloud Computing
- ……