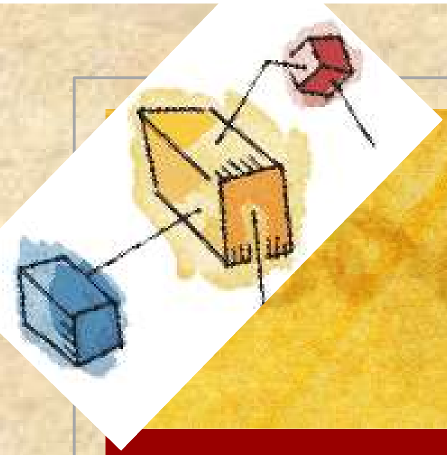
Chapter 1 Operating System Overview

Operating System

- A program that controls the execution of application programs
- An interface between applications and hardware

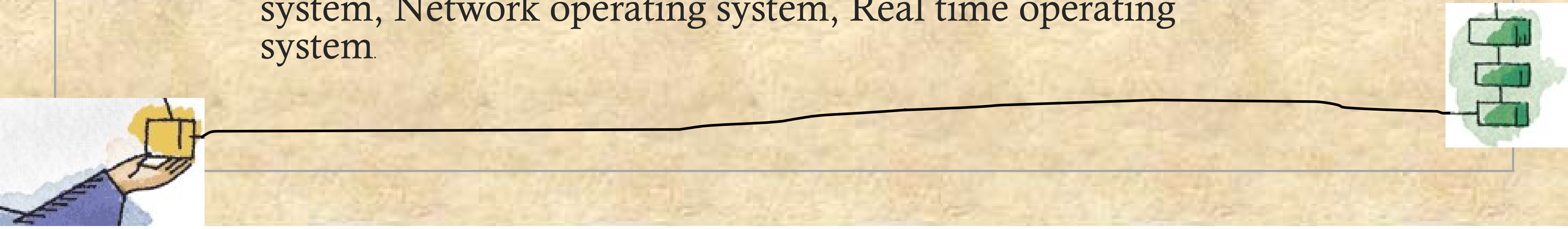
Main objectives of an OS:

- Convenience
- Efficiency
- Ability to evolve



OS objectives

- 1.Convenient: OS is used for user services like Program Execution, I/O operations, File System Manipulation, Communications and Error Detection to execute in a convenient manner.
- 2. Efficient: OS has the capability to provide an environment to run above user services in an efficient manner. Batch processing, Multiprocessing, multiprogramming, etc are some ways by which OS increases efficiency.
- 3. Ability to evolve: OS have the ability to evolve. Evolution has been as following- Batch Operating System, Time-sharing operating system, Distributed operating system, Network operating system, Real time operating system.



The OS as a User/Computer Interface

- Computer Hardware-Software Structure
  - Layered organization
- OS services to users

Computer Hardware and Software Infrastructure

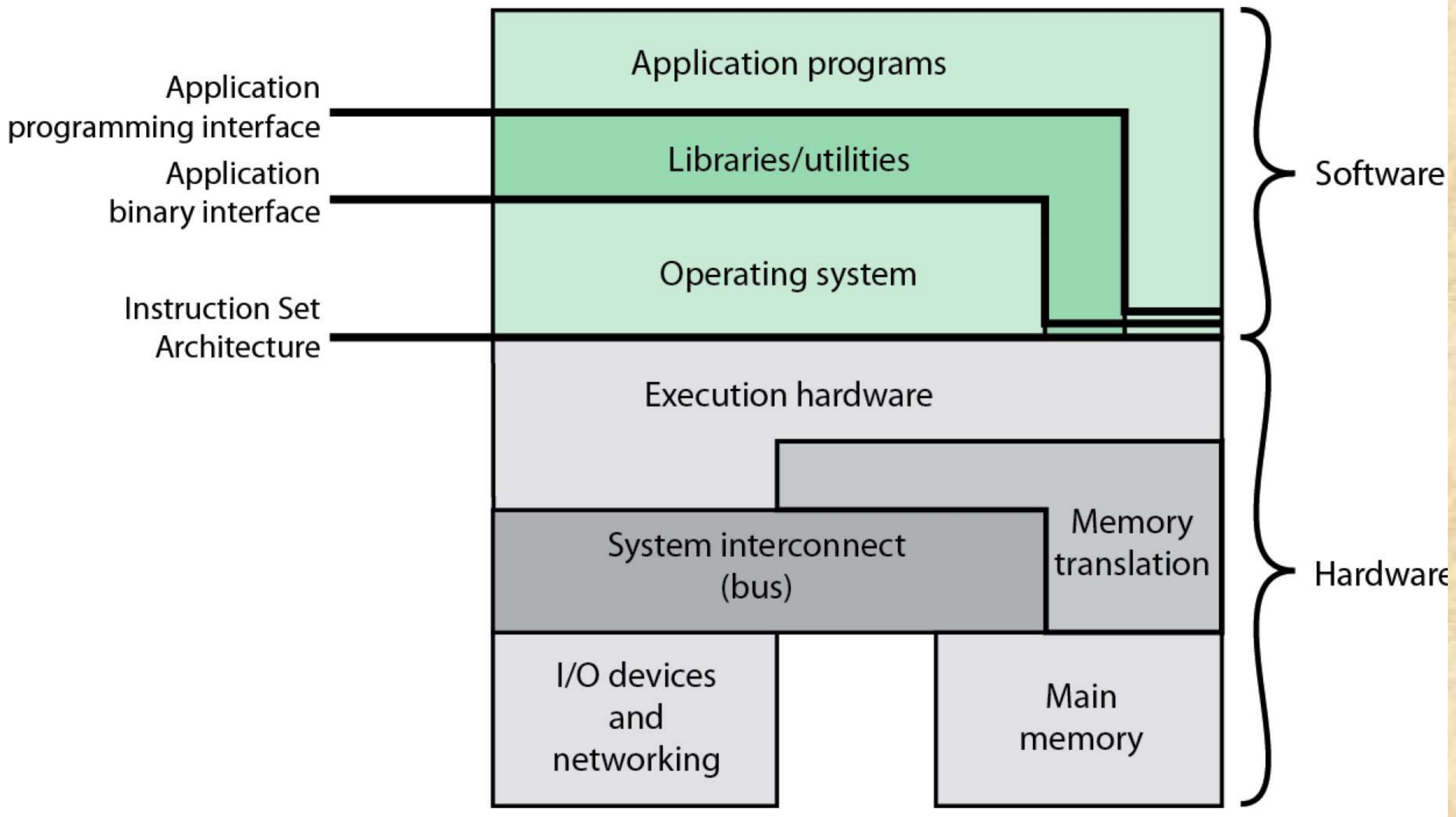

Application programming interface
Application binary interface
Instruction Set Architecture
Application programs
Libraries/utilities
Operating system
Execution hardware
System interconnect (bus)
Memory translation
I/O devices and networking
Main memory
Software
Hardware

Figure 2.1 Computer Hardware and Software Infrastructure

Operating System Services

- One set of operating-system services provides functions that are helpful to the user:
  - User interface - Almost all operating systems have a user interface (UI)
    - Varies between Command-Line (CLI), Graphics User Interface (GUI), Batch
  - Program execution - The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)
  - I/O operations - A running program may require I/O, which may involve a file or an I/O device.
  - File-system manipulation - The file system is of particular interest. Obviously, programs need to read and write files and directories, create and delete them, search them, list file Information, permission management.

Operating System Services (Cont.)

- One set of operating-system services provides functions that are helpful to the user (Cont):
  - Communications – Processes may exchange information, on the same computer or between computers over a network
    - Communications may be via shared memory or through message passing (packets moved by the OS)
  - Error detection – OS needs to be constantly aware of possible errors
    - May occur in the CPU and memory hardware, in I/O devices, in user program
    - For each type of error, OS should take the appropriate action to ensure correct and consistent computing
    - Debugging facilities can greatly enhance the user's and programmer's abilities to efficiently use the system

Operating System Services (Cont.)

- Another set of OS functions exists for ensuring the efficient operation of the system itself via resource sharing
  - **Resource allocation -** When multiple users or multiple jobs running concurrently, resources must be allocated to each of them
    - Many types of resources - Some (such as CPU cycles,mainmemory, and file storage) may have special allocation code, others (such as I/O devices) may have general request and release code.
  - **Accounting -** To keep track of which users use how much and what kinds of computer resources
  - **Protection and security -** The owners of information stored in a multiuser or networked computer system may want to control use of that information, concurrent processes should not interfere with each other
    - **Protection** involves ensuring that all access to system resources is controlled
    - **Security** of the system from outsiders requires user authentication, extends to defending external I/O devices from invalid access attempts
    - If a system is to be protected and secure, precautions must be instituted throughout it. A chain is only as strong as its weakest link.

The Operating System as a Resource Manager

- A computer is a set of resources for moving, storing, & processing data
- The OS is responsible for managing these resources
- The OS exercises its control through software

Operating System as Resource Manager

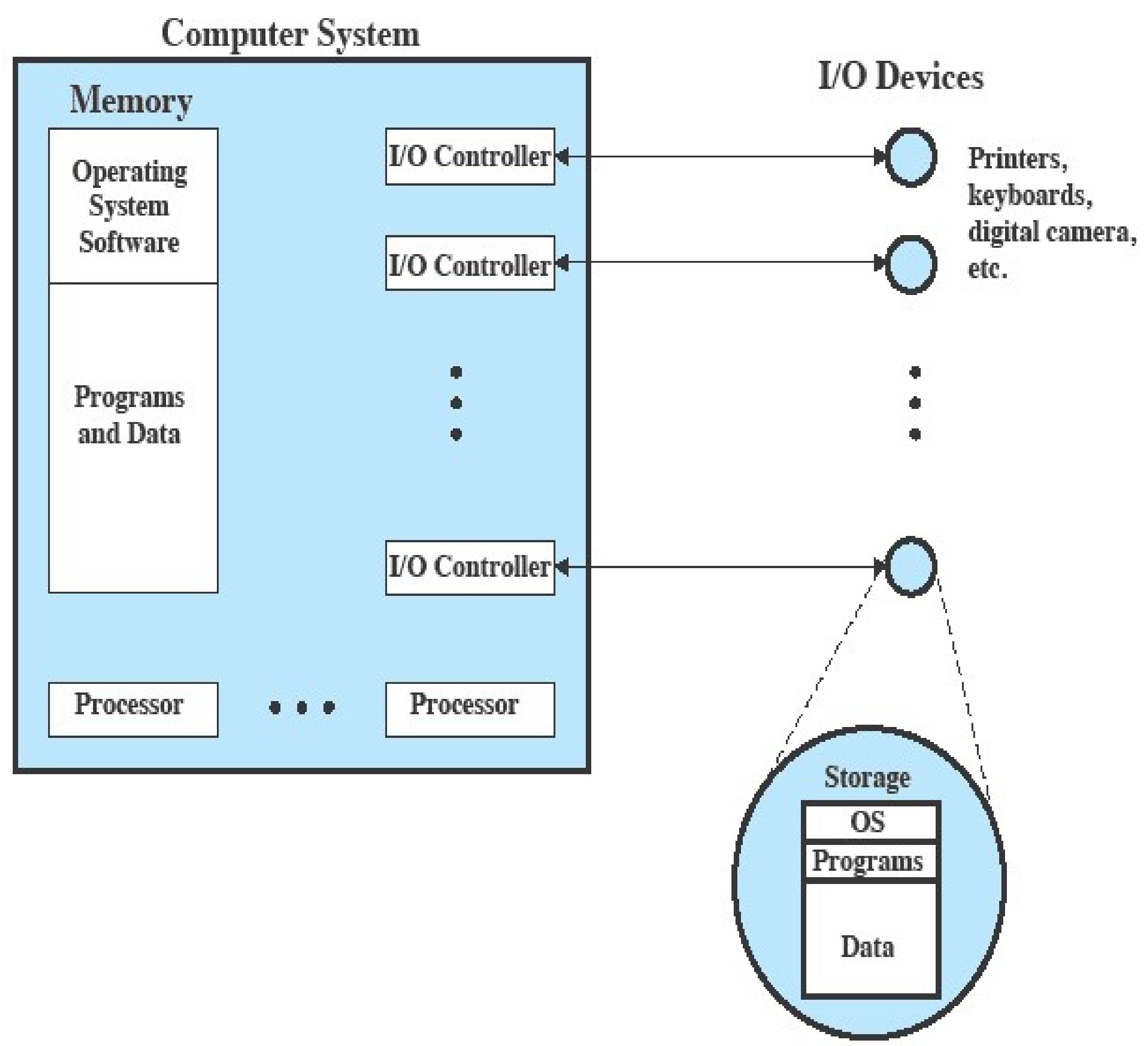


Figure 2.2 The Operating System as Resource Manager

OS as resource manager

- CPU: Perform execution of program – which is in high level language.Compiled or executable program has to be executed by CPUshould be brought into MM from disk.Because CPU can not directly access SS. It can access MM directly.
- MM:Single memory block shared by multiple program of a user/users. DOS/UNIX. When CPU find piece of program or data on SS, it is duty of DD to bring it in MM.
- I/O devices:I/P-keyboard-feed data , Printer/monitor-O/P.
- Secondary storage-special I/O device.-File-R/W.
- So OS responsibility to manage these devices efficiently. So more than 1 user can access those simultaneously.
- Modern OS-Distributed computing- Avoid duplicates copy, efficient use of resources

Evolution of Operating Systems

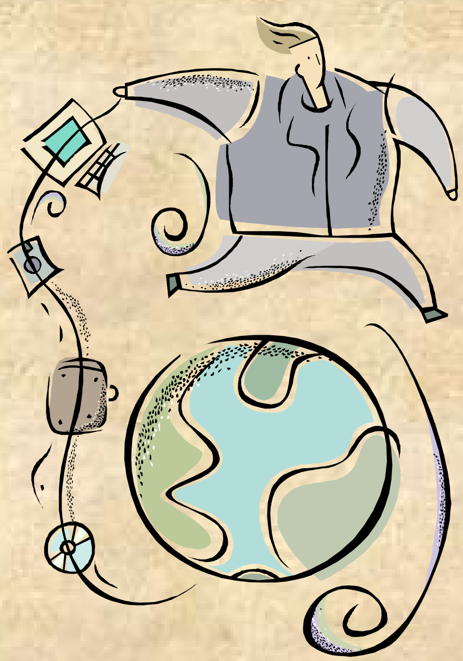
- A major OS will evolve over time for a number of reasons:

Hardware upgrades

New types of hardware

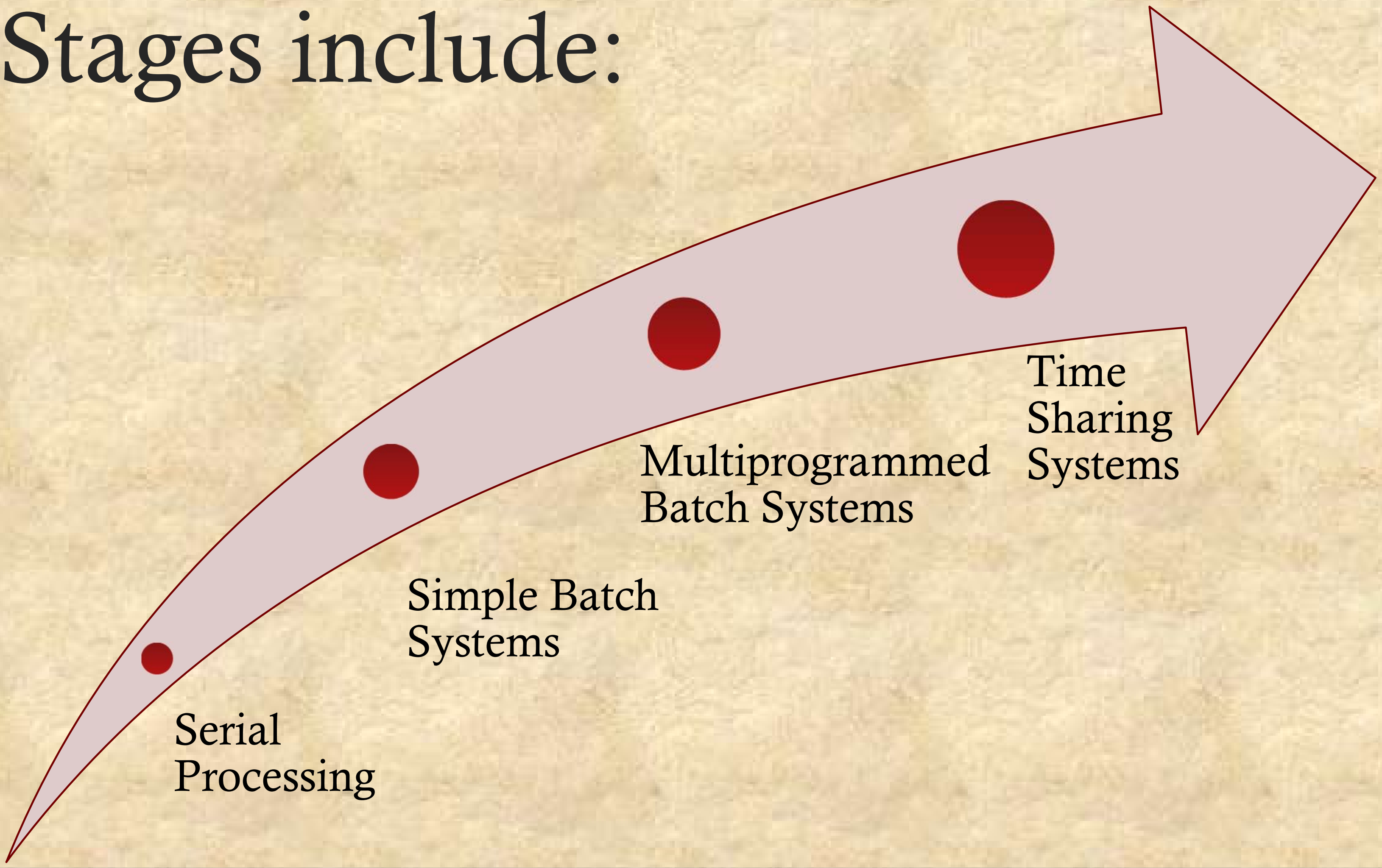
New services

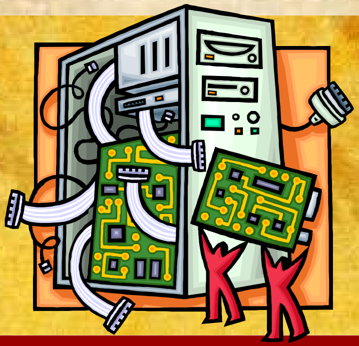
Fixes



Evolution of Operating Systems

- Stages include:





Serial Processing

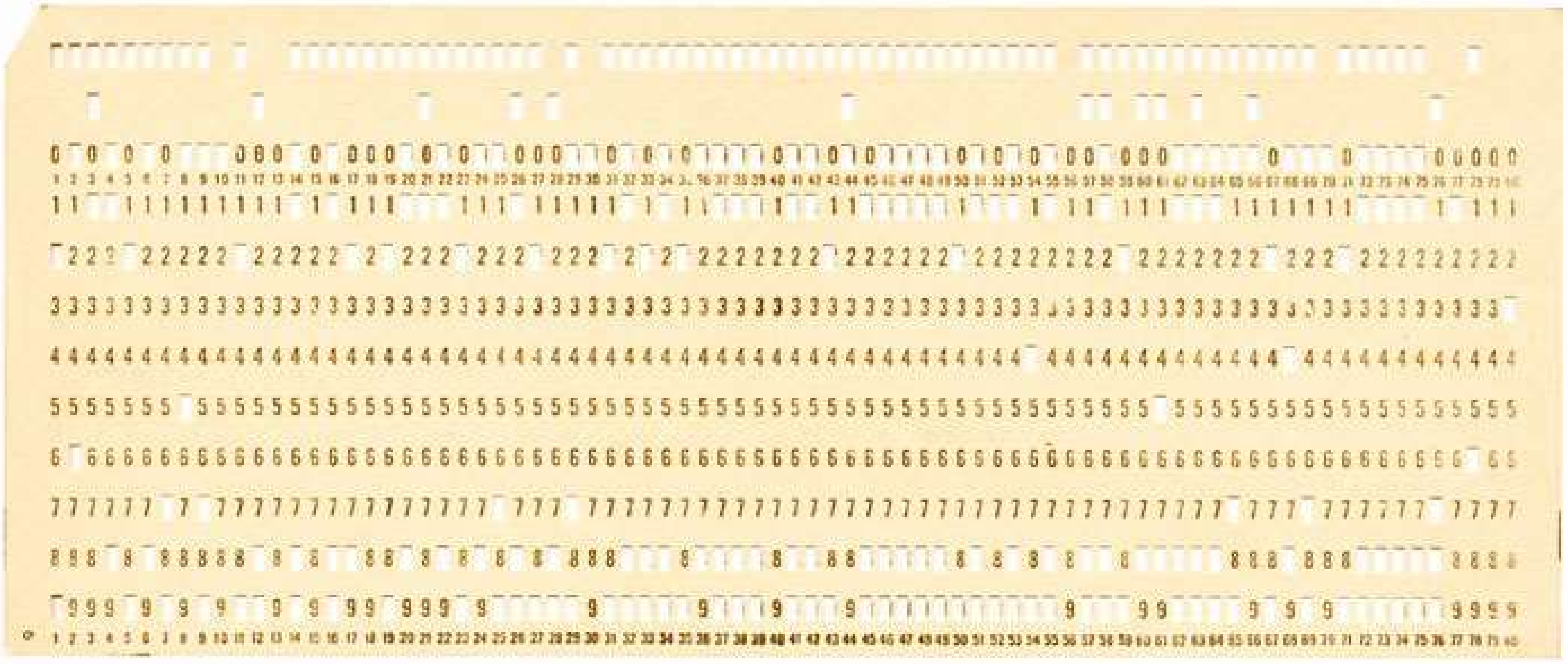
Earliest Computers:

- No operating system
  - programmers interacted directly with the computer hardware
- Computers ran from a console with display lights, toggle switches, some form of input device, and a printer
- Users have access to the computer in "series"

Problems:

- Scheduling:
  - most installations used a hardcopy sign-up sheet to reserve computer time
    - time allocations could run short or long, resulting in wasted computer time
- Setup time
  - a considerable amount of time was spent just on setting up the program to run

Example of a punch card



Simple Batch Systems

- Early computers were very expensive
  - important to maximize processor utilization
- Monitor
  - user no longer has direct access to processor
  - job is submitted to computer operator who batches them together and places them on an input device
  - program branches back to the monitor when finished

Monitor Point of View

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

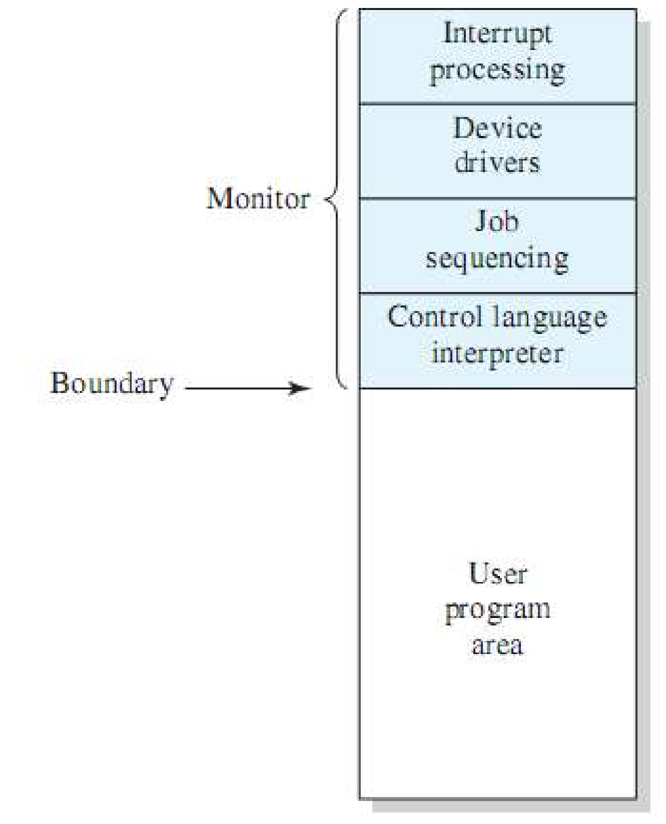


Figure 2.3 Memory Layout for a Resident Monitor

Job Control Language (JCL)

Special type of programming language used to provide instructions to the monitor

↓

what compiler to use

↓

what data to use

Desirable Hardware Features

Memory protection for monitor

- while the user program is executing, it must not alter the memory area containing the monitor

Timer

- prevents a job from monopolizing the system

Privileged instructions

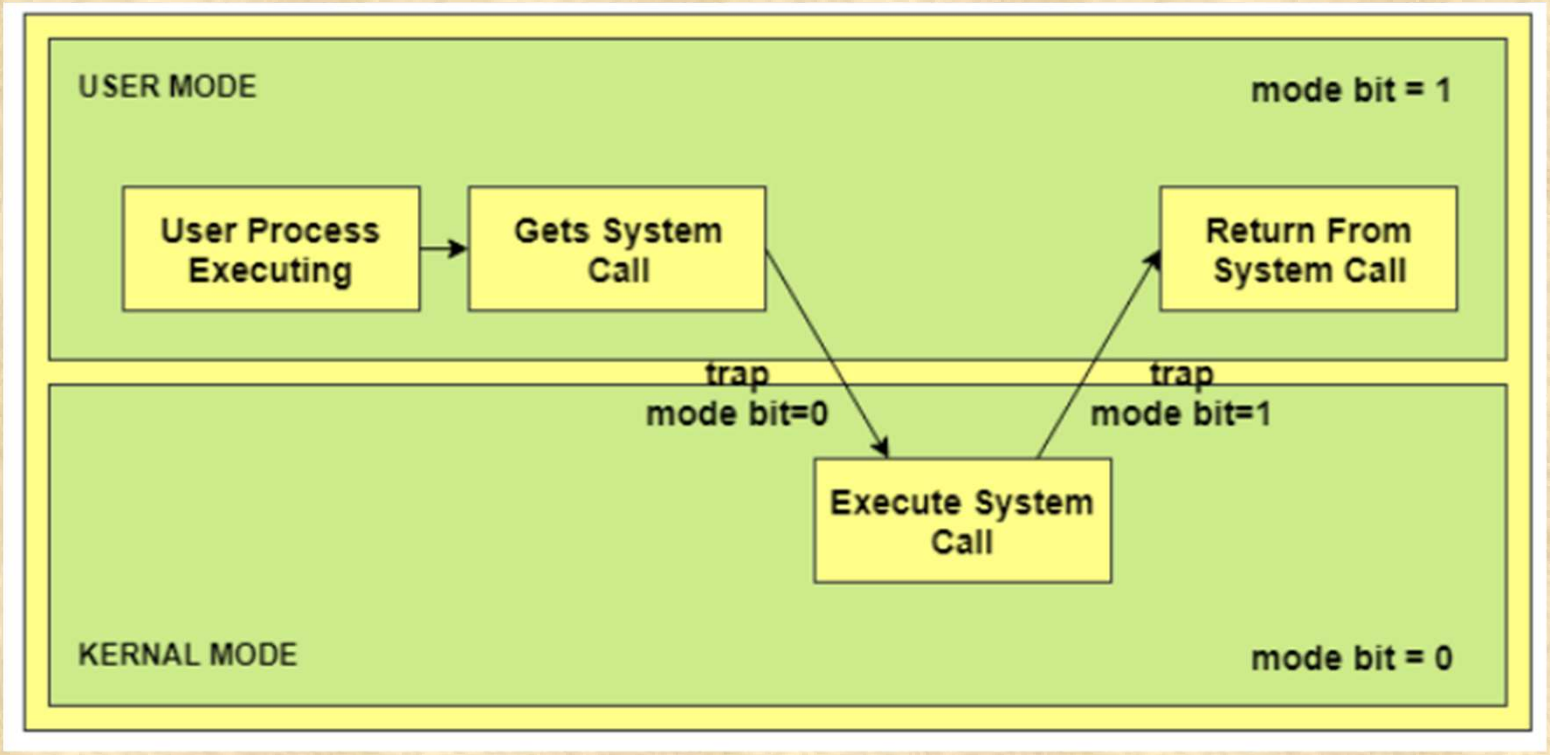
- can only be executed by the monitor

Interrupts

- gives OS more flexibility in controlling user programs

User and kernel mode

- User Mode :The system is in user mode when the operating system is running a user application such as handling a text editor. The transition from user mode to kernel mode occurs when the application requests the help of operating system or an interrupt or a system call occurs.
- The mode bit is set to 1 in the user mode. It is changed from 1 to 0 when switching from user mode to kernel mode.
- Kernel Mode:The system starts in kernel mode when it boots and after the operating system is loaded, it executes applications in user mode. There are some privileged instructions that can only be executed in kernel mode.
- These are interrupt instructions, input output management etc. If the privileged instructions are executed in user mode, it is illegal and a trap is generated.


USER MODE
mode bit = 1
User Process Executing
Gets System Call
Return From System Call
trap
mode bit=0
trap
mode bit=1
Execute System Call
KERNAL MODE
mode bit = 0

Need of 2 mode

- The lack of a dual mode i.e user mode and kernel mode in an operating system can cause serious problems. Some of these are:
- A running user program can accidentaly wipe out the operating system by overwriting it with user data.
- Multiple processes can write in the same system at the same time, with disastrous results.

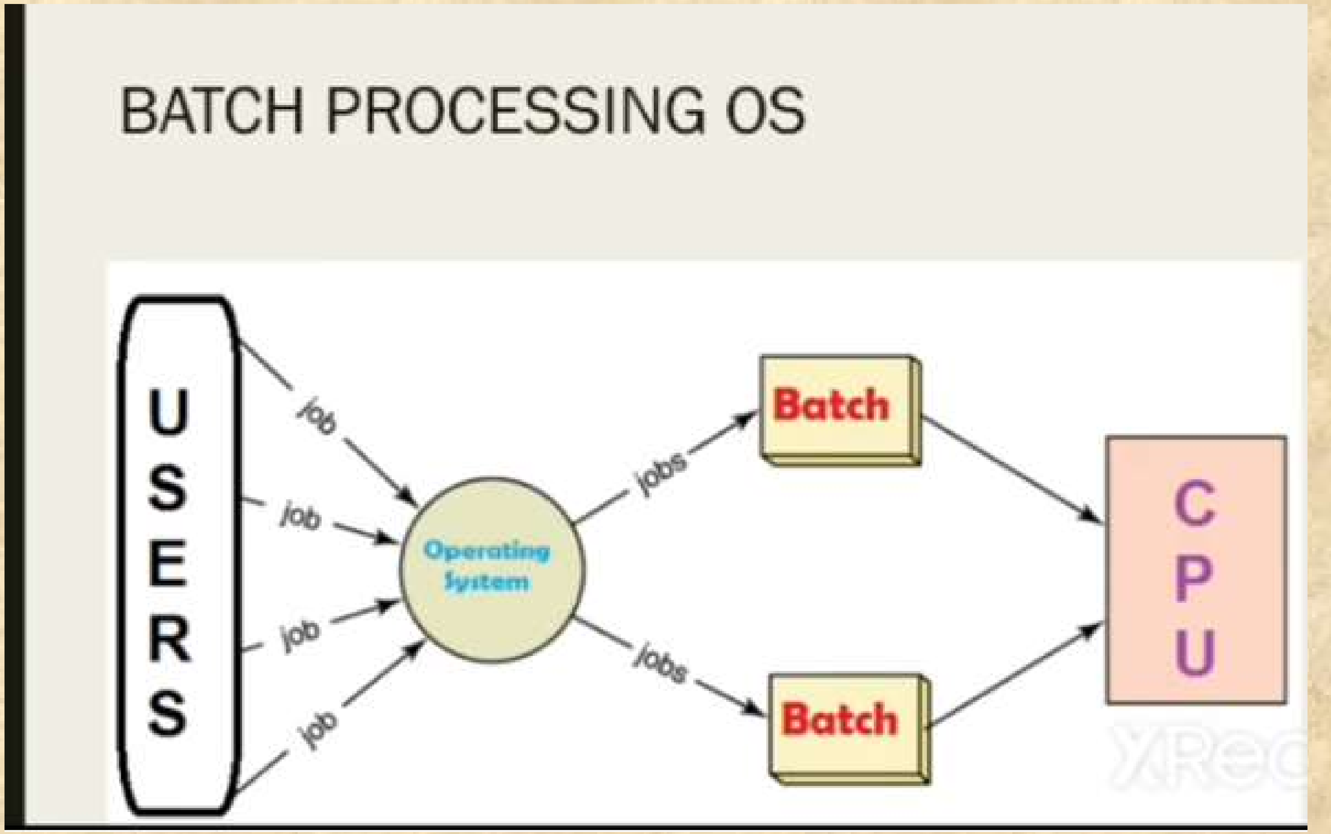
Modes of Operation

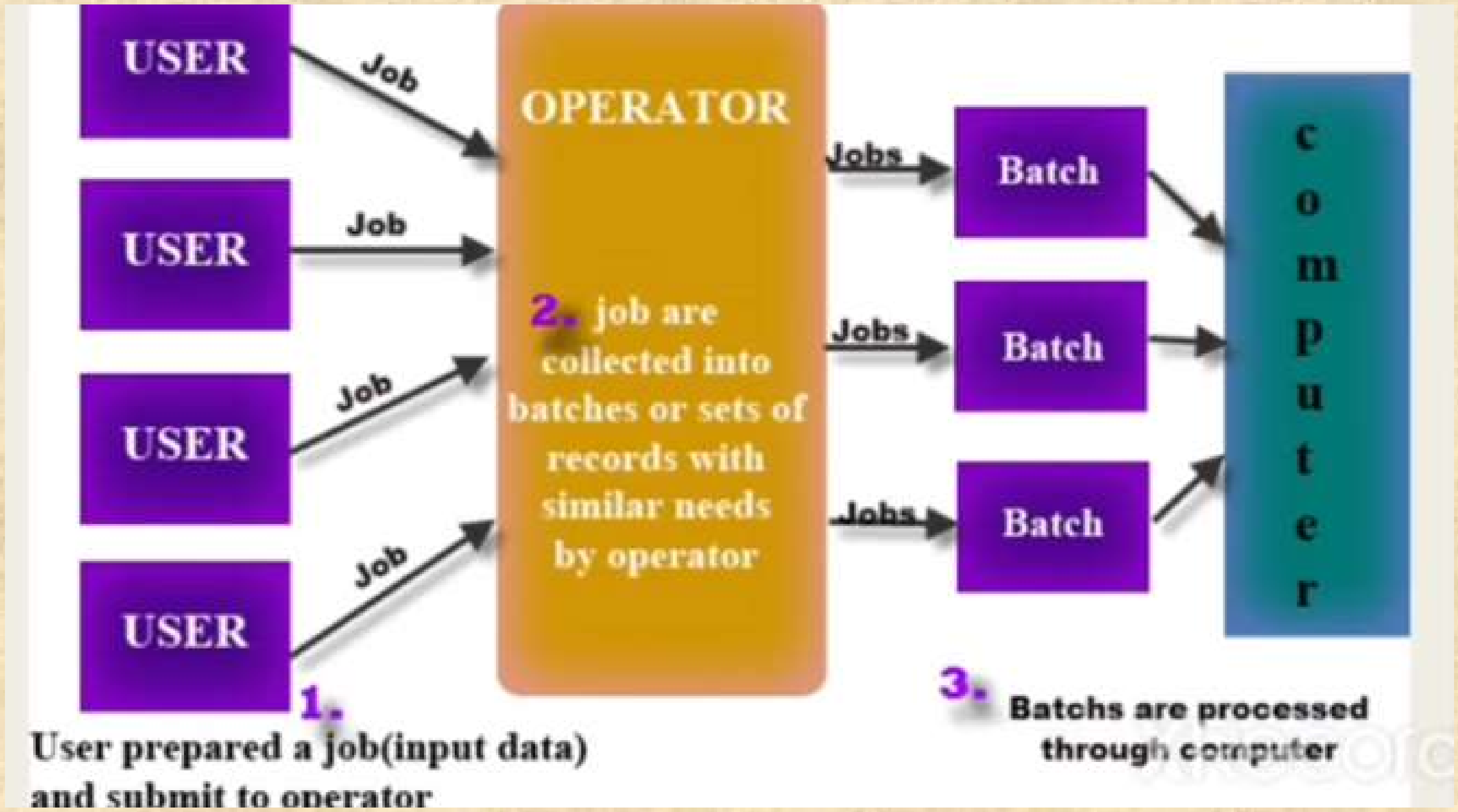
User Mode

- user program executes in user mode
- certain areas of memory are protected from user access
- certain instructions may not be executed

Kernel Mode

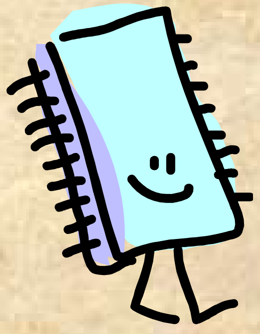
- monitor executes in kernel mode
- privileged instructions may be executed
- protected areas of memory may be accessed


BATCH PROCESSING OS


USER
Job
USER
Job
USER
Job
USER
Job
OPERATOR
2. job are collected into batches or sets of records with similar needs by operator
Jobs
Batch
Jobs
Batch
Jobs
Batch
c o m p u t e r
1.
User prepared a job(input data) and submit to operator
3.
Batchs are processed through computer

Simple Batch System Overhead

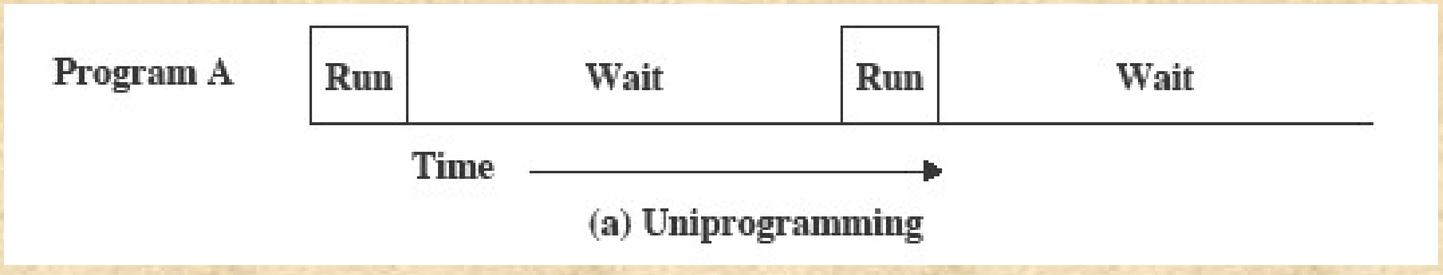
- Processor time alternates between execution of user programs and execution of the monitor
- Sacrifices:
  - some main memory is now given over to the monitor
  - some processor time is consumed by the monitor
- Despite overhead, the simple batch system improves utilization of the computer. (How?)



Multiprogrammed Batch Systems

- Processor is often idle
  - even with automatic job sequencing
  - I/O devices are slow compared to processor

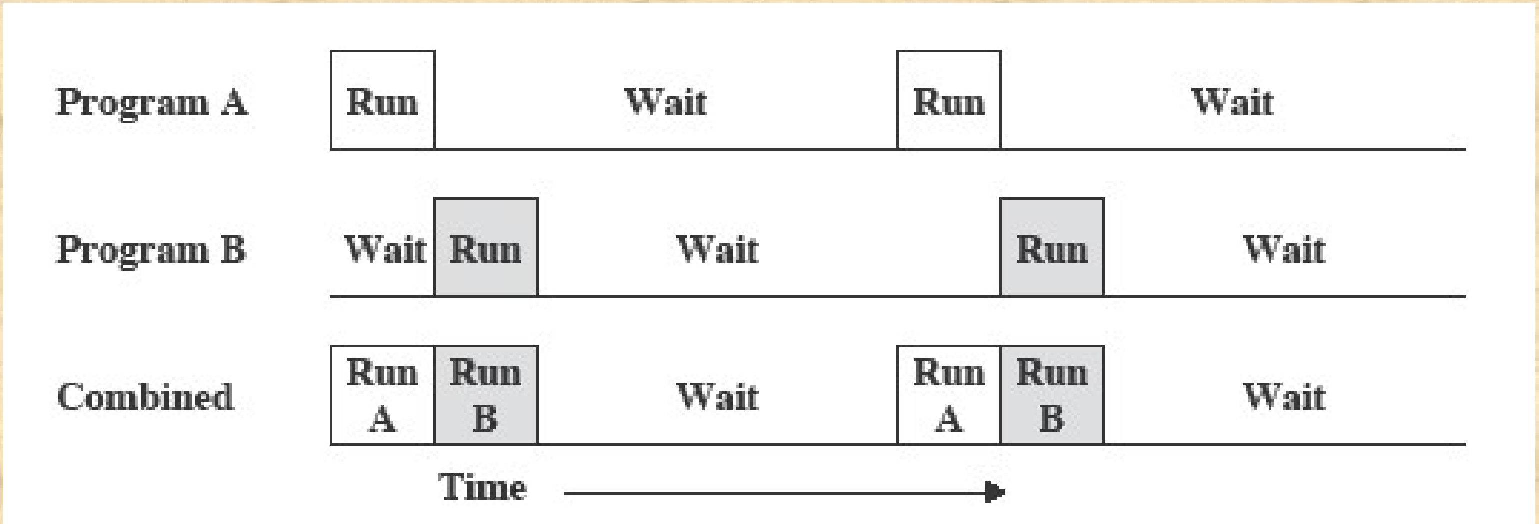
Uniprogramming



(a) Uniprogramming

- The processor spends a certain amount of time executing, until it reaches an I/O instruction; it must then wait until that I/O instruction concludes before proceeding

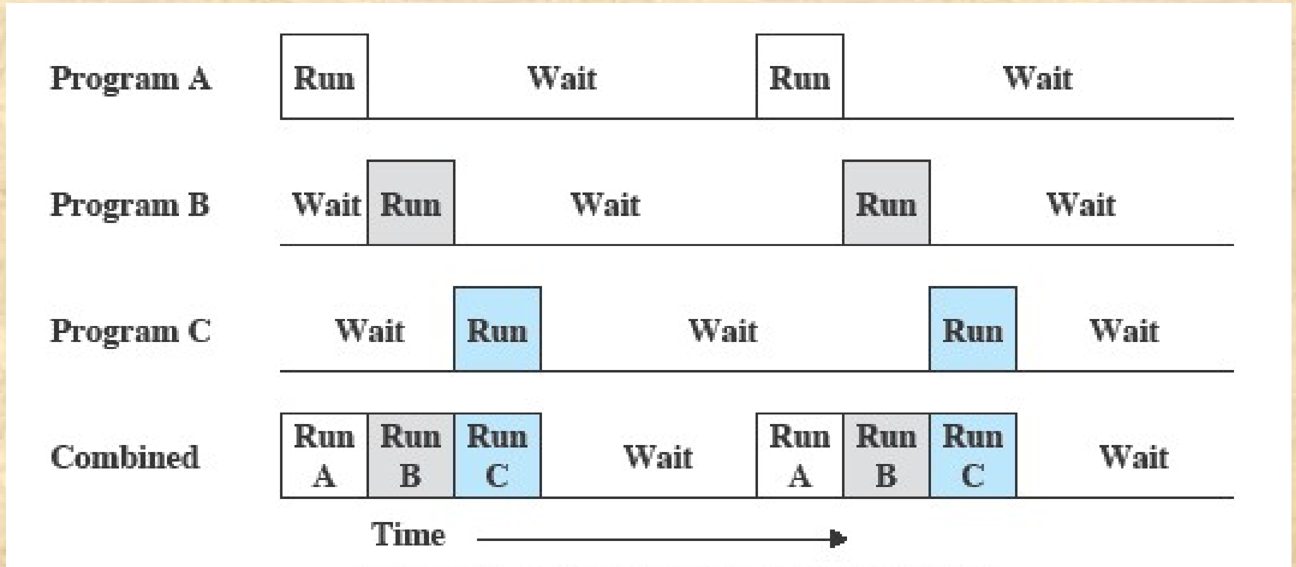
Multiprogramming



(b) Multiprogramming with two programs

- There must be enough memory to hold the OS (resident monitor) and one user program
- When one job needs to wait for I/O, the processor can switch to the other job, which is likely not waiting for I/O

Multiprogramming



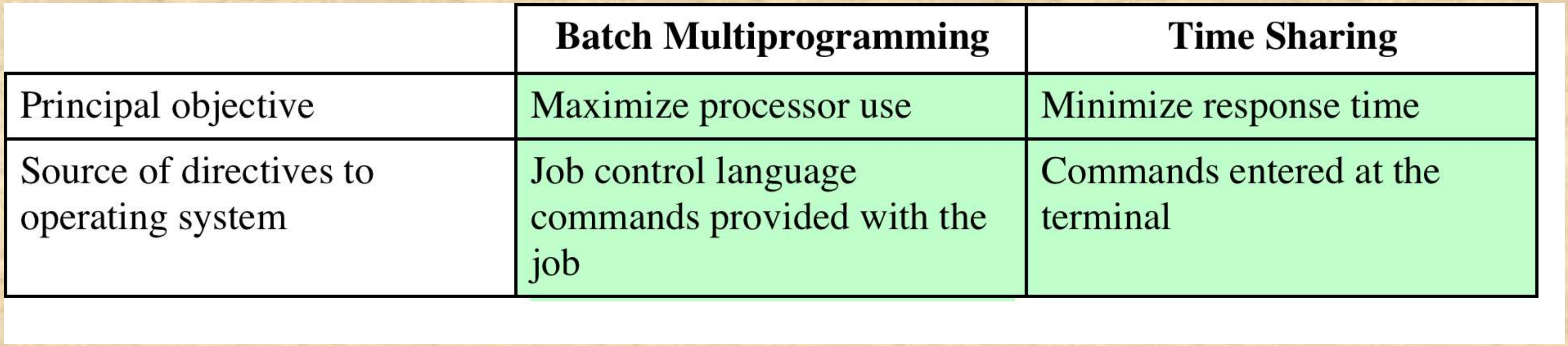
(c) Multiprogramming with three programs

- Multiprogramming
  - also known as multitasking
  - memory is expanded to hold three, four, or more programs and switch among all of them

Time-Sharing Systems

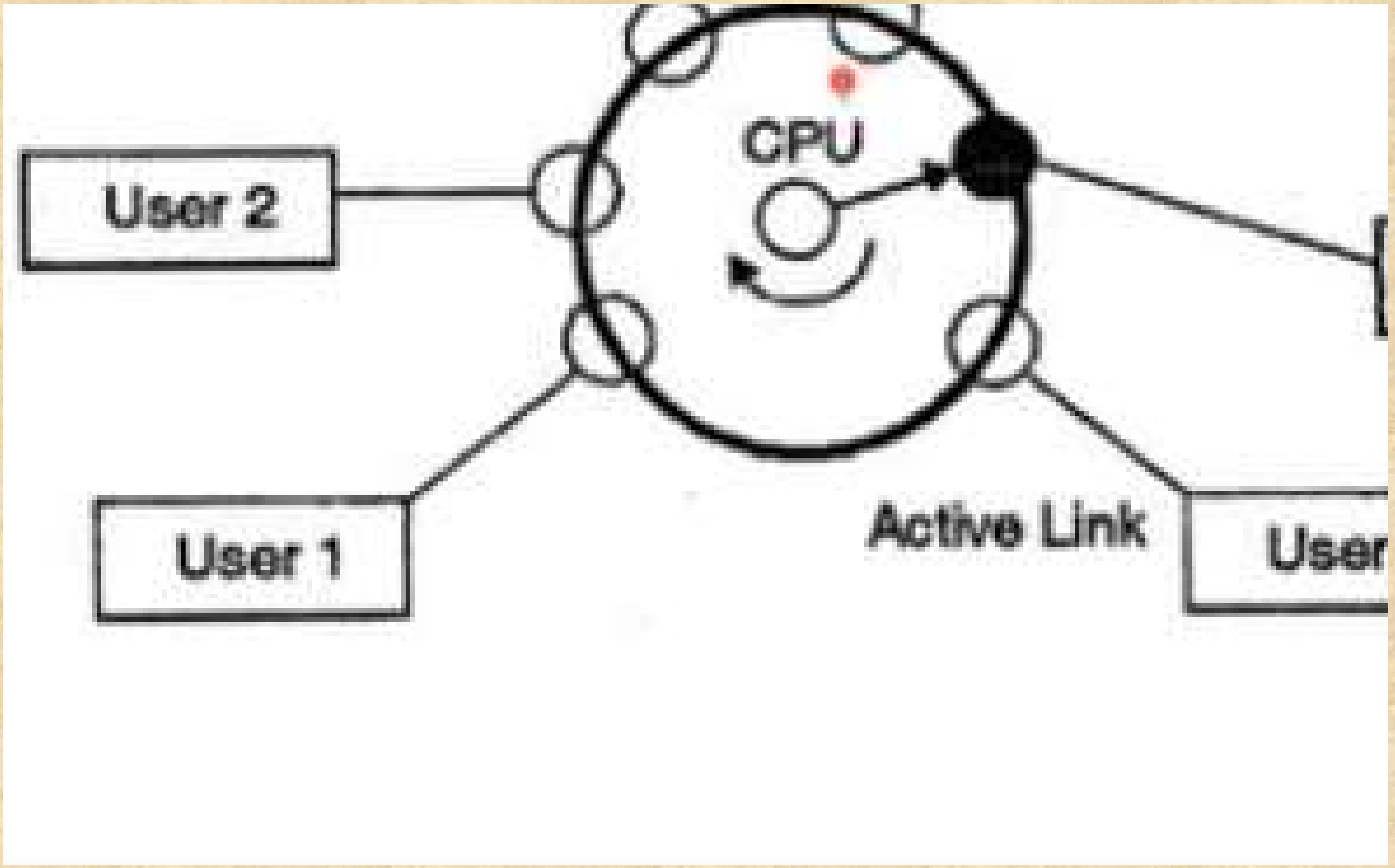
- Can be used to handle multiple interactive jobs
- Processor time is shared among multiple users
- Multiple users simultaneously access the system through terminals, with the OS interleaving the execution of each user program in a short burst or quantum of computation

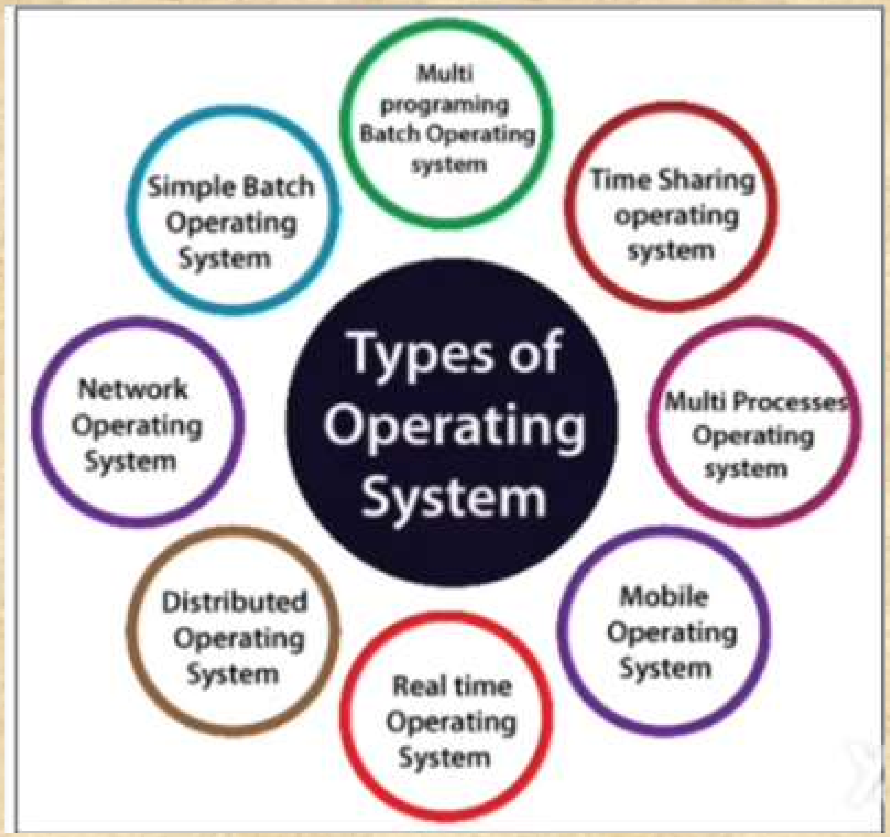
Batch Multiprogramming vs. Time Sharing



	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

Table 2.3 Batch Multiprogramming versus Time Sharing

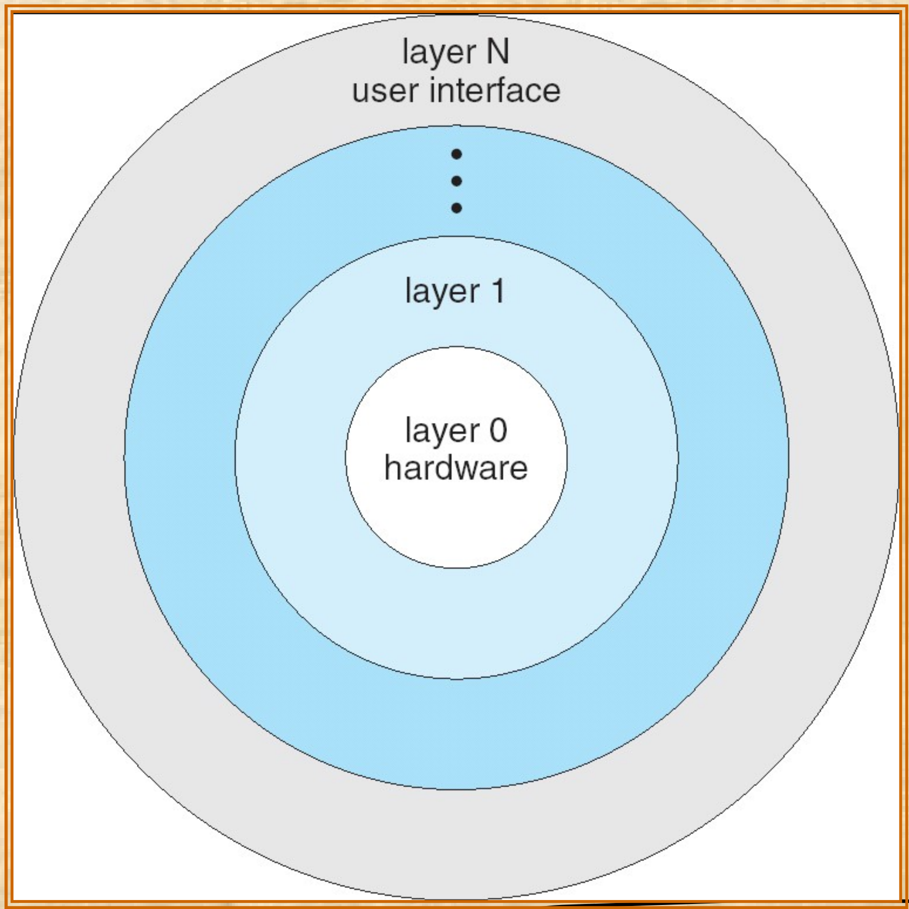

CPU
User 2
User 1
Active Link
User


Types of Operating System
Multi programing Batch Operating system
Time Sharing operating system
Multi Processes Operating system
Mobile Operating System
Real time Operating System
Distributed Operating System
Network Operating System
Simple Batch Operating System

Layered Approach

- The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers

Layered Operating System


layer N
user interface
layer 1
layer 0
hardware