Operating Systems

Amrita University

Operating System (OS)

- A software that controls the overall operation of a computer
- Acts as
  - Means for the user to store and retrieve files
  - An interface to request the execution of programs
  - An environment necessary to execute the programs requested

Examples of OS

- Windows
  - provided in numerous versions by Microsoft and widely used in the PC arena
- UNIX
  - popular choice for larger computer systems as well as PCs
  - core of two other popular operating systems
    - Mac OS by Apple for its range of Mac machines
    - Solaris, which developed by Sun Microsystems (now owned by Oracle)

More Examples of OS

- Linux
  - originally developed non-commercially by computer enthusiasts
  - now available through many commercial sources, including IBM

Differences in OS

- For casual computer users
  - largely cosmetic
- For computing professionals
  - different operating systems can represent major changes in the tools they work with or
  - the philosophy they follow in disseminating and maintaining their work.
- at their core all mainstream operating systems address the same kinds of problems that computing experts have faced for more than half a century.

History of OS

- Today’s operating systems
  - large, complex software packages
  - Grown from humble beginnings
- Computers in 1940s and 1950s
  - not very flexible or efficient
  - Machines occupied entire rooms
  - Program execution required significant preparation of equipment
    - in terms of mounting magnetic tapes, Placing punched cards in card readers, setting switches etc.

History of OS

- Execution of each program was called a **job**
  - handled as an isolated activity
  - the machine was prepared for executing the program
  - the program was executed
  - all the tapes, punched cards, etc. had to be retrieved before the next program preparation could begin

Time sharing in Earlier OS

- When several users needed to share a machine
  - sign-up sheets were provided (for users to reserve the machine for blocks of time)
  - During the time period allocated to a user, the machine was totally under that user’s control
  - session usually began with program setup, followed by short periods of program execution
  - the next user was impatiently starting to set up

Later OSs

- Began as systems for
  - simplifying program setup and
  - streamlining the transition between jobs
- Early Development
  - separation of users and equipment
  - eliminated the physical transition of people in and out of the computer room
  - A computer operator was hired to operate the machine

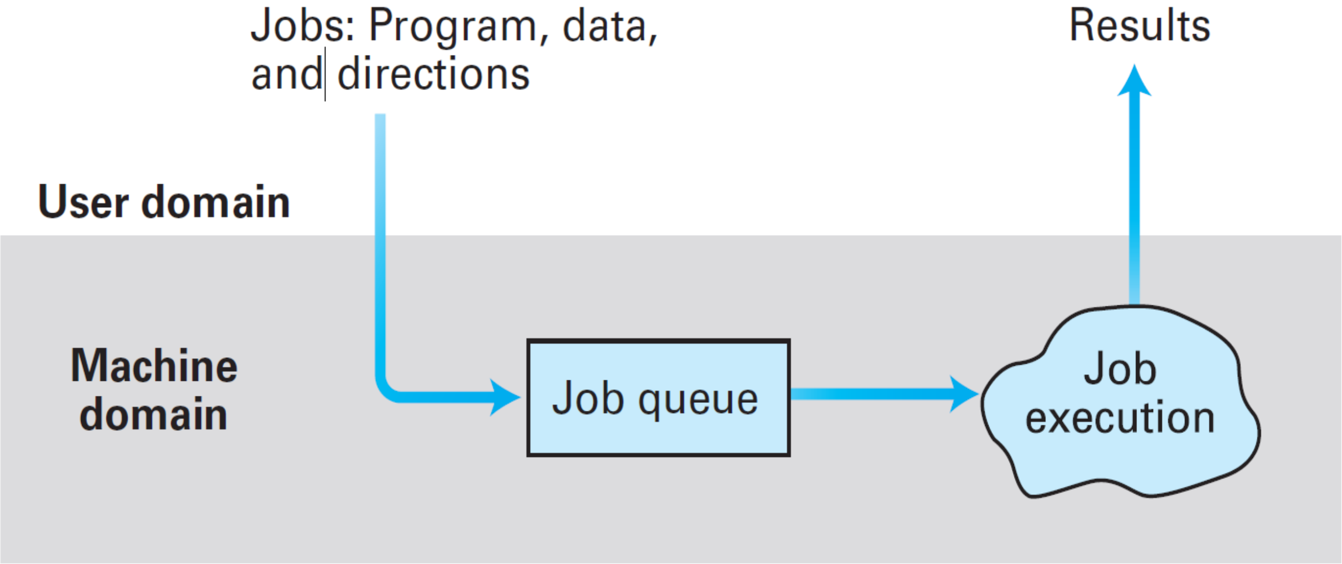
Functions of the programmer

- Anyone wanting to run a program must submit to the operator
  - the program
  - any required data and
  - special directions about the program’s requirements
  - return later for the results

Functions of the operator

- loaded these materials into the machine’s mass storage
- a program called the operating system would read and execute them one at a time
- It was a beginning of **batch processing**
    - the execution of jobs by collecting them in a single batch
    - then executing them without further interaction with the user

Batch processing systems



- Jobs residing in mass storage wait for execution in a **job queue**
- **Queue**
  - is a storage organization
  - objects (in this case, jobs) are ordered in **first-in, first-out** (abbreviated FIFO and pronounced “FI-foe”) fashion.
  - Objects (jobs) are removed from the queue in the order in which they arrived

Batch processing systems (contd.)

- Priority
  - In reality, most job queues do not rigorously follow the FIFO structure
  - most operating systems provide for consideration of job priorities
  - a job waiting in the job queue can be bumped by a higher-priority job

Batch processing systems (contd.)

- Job Control Language (JCL)
  - a set of instructions
  - explaining the steps required
  - to prepare the machine for that particular job
  - JCL stored with the job in the job queue

Batch processing systems (contd.)

- When the job was selected for execution
  - Operating system printed these instructions on a printer
  - These instructions were read and followed by the computer operator
  - OS reported errors such as “network not available” and “printer not responding.”

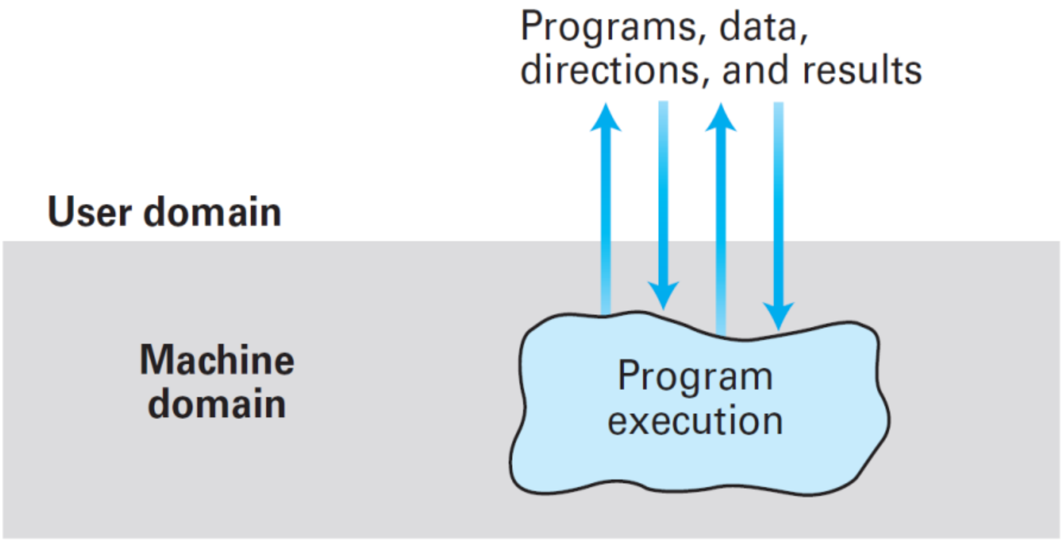
Drawbacks of Batch Processing

- Users have no interaction with their jobs once they are submitted to the operator
- Approach is acceptable for some applications
  - payroll processing where data and all processing decisions are established in advance
- Not acceptable when the user must interact with a program during its execution
  - Reservation systems in which reservations and cancellations must be reported as they occur;
  - word processing systems in which documents are developed in a dynamic write and rewrite manner;
  - computer games in which interaction with the machine is the central feature of the game.

Interactive Processing

- Allowed a program being executed to carry on a dialogue with the user through remote terminals
  - – A terminal was a little more than an electronic typewriter by which
  - – the user could type input and read the computer's response that was printed on paper
  - – Today terminals have evolved into more sophisticated devices called workstations and even into complete PCs that can
  - – function as stand-alone computers when desired

Interactive Processing



- For successful interactive processing,
  - the actions of the computer be sufficiently fast to coordinate with the needs of the user
  - rather than forcing the user to conform to the machine’s timetable

Interactive processing (contd.)

- Task of processing payroll
  - can be scheduled to conform to the amount of time required by the computer
- Using a word processor
  - would be frustrating if the machine did not respond promptly as characters are typed

Real-time processing

- Computer is forced to execute tasks under a deadline
- Actions performed are said to occur in real-time
- The computer performs the task in accordance with deadlines in its (external real-world) environment

Time sharing in Interactive Systems

- To serve only one user at a time, real-time processing would have been no problem
- Computers in the 1960s and 1970s were expensive
- Each machine had to serve more than one user
- Common for several users, working at remote terminals, to seek interactive service from a machine at the same time
- real-time considerations presented obstacles

Time sharing through Multi-programming

- Time is divided into intervals
- Execution of each job is restricted to only one interval at a time
- At the end of each interval, the current job is temporarily set aside
- another is allowed to execute during the next interval
- Rapid shuffling the jobs back and forth creates an illusion of several jobs executing simultaneously

Early time sharing systems

- Depending on the types of jobs being executed,
  - early time-sharing systems were able to provide acceptable real-time processing to as many as 30 users simultaneously

Multi-tasking

- Multiprogramming in single-user systems
  - refers to one user executing numerous tasks simultaneously
  - Whereas time-sharing refers to multiple users sharing access to a common computer

Multi-user time-sharing OS

- A large central computer connected to numerous workstations
- Users could communicate directly with the computer from outside the computer room (rather than submitting requests to a computer operator)
- Commonly used programs were stored in the machine’s mass storage devices
- Operating systems were designed to execute these programs as requested from the workstations

Computer Operator Vs System Administrator

- Today role of a computer operator has disappeared
- computer user assumes all of the responsibilities of computer operation
- most large computer installations run essentially unattended
- the job of computer operator has given way to that of a system administrator

Roles and Responsibilities of a System Administrator

- Rather than operating the machines in a hands-on manner, manages the computer system in
  - Obtaining and overseeing the installation of new equipment and software
  - enforcing local regulations such as
    - issuing of new accounts
    - establishing mass storage space limits for the various users
    - coordinating efforts to resolve problems that arise in the system

Problems faced by Oss on multi-processor machines

- **Load balancing**
  - dynamically allocating tasks to the various processors so that all processors are used efficiently
- **Scaling**
  - (breaking tasks into a number of subtasks compatible with the number of processors available
- Manage resources in an Computer Network

Research in OS

- Devices that are dedicated to specific tasks such as
  - medical devices,
  - vehicle electronics,
  - home appliances,
  - cell phones, or
  - other hand-held computers.
- Computer Systems in above devices are known as Embedded Systems

Embedded Systems – Expectations and Examples

- Often expected to conserve battery power
- meet demanding real-time deadlines, or
- operate continuously with little or no human oversight
- Some examples of embedded systems:
  - VxWORKS by Wind River Systems used in the Mars Exploration Rovers named Spirit and Opportunity
  - Windows CE (also known as Pocket PC) developed by Microsoft
  - Palm OS developed by PalmSource, Inc., especially for use in hand-held devices.

Test Yourself

Which of the following activities require real-time processing?

a. Printing mailing labels

b. Playing a computer game

c. Displaying numbers on a smartphone screen as they are dialed

d. Executing a program that predicts the state of next year's economy

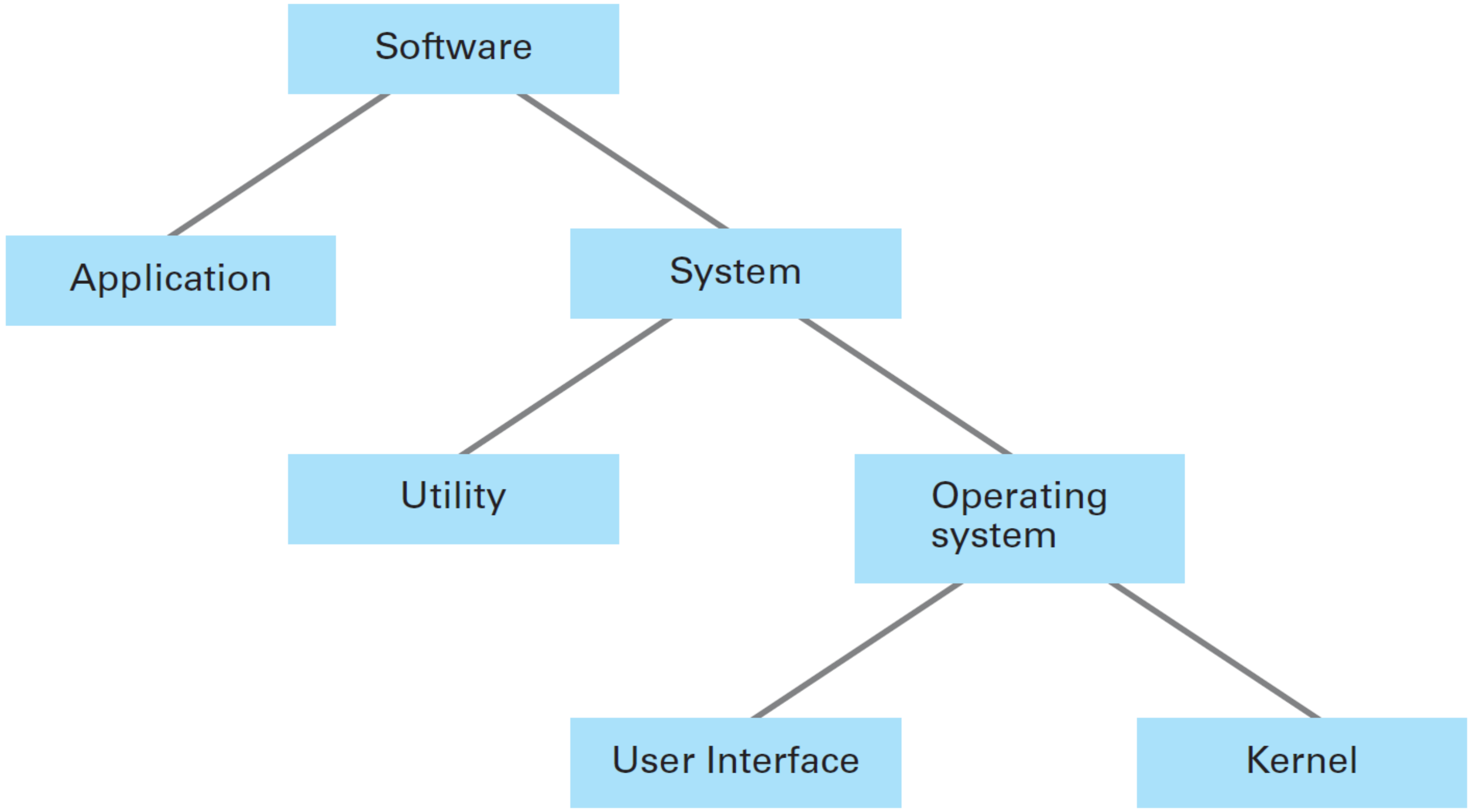
e. Playing an MP3 recording

Answers to Test Yourself

- Options (b), (c), and (e)

Operating System Architecture

- Categories of software



Categories of software

- **Application software**
  - spreadsheets
  - database systems
  - desktop publishing systems
  - accounting systems
  - program development software and
  - games

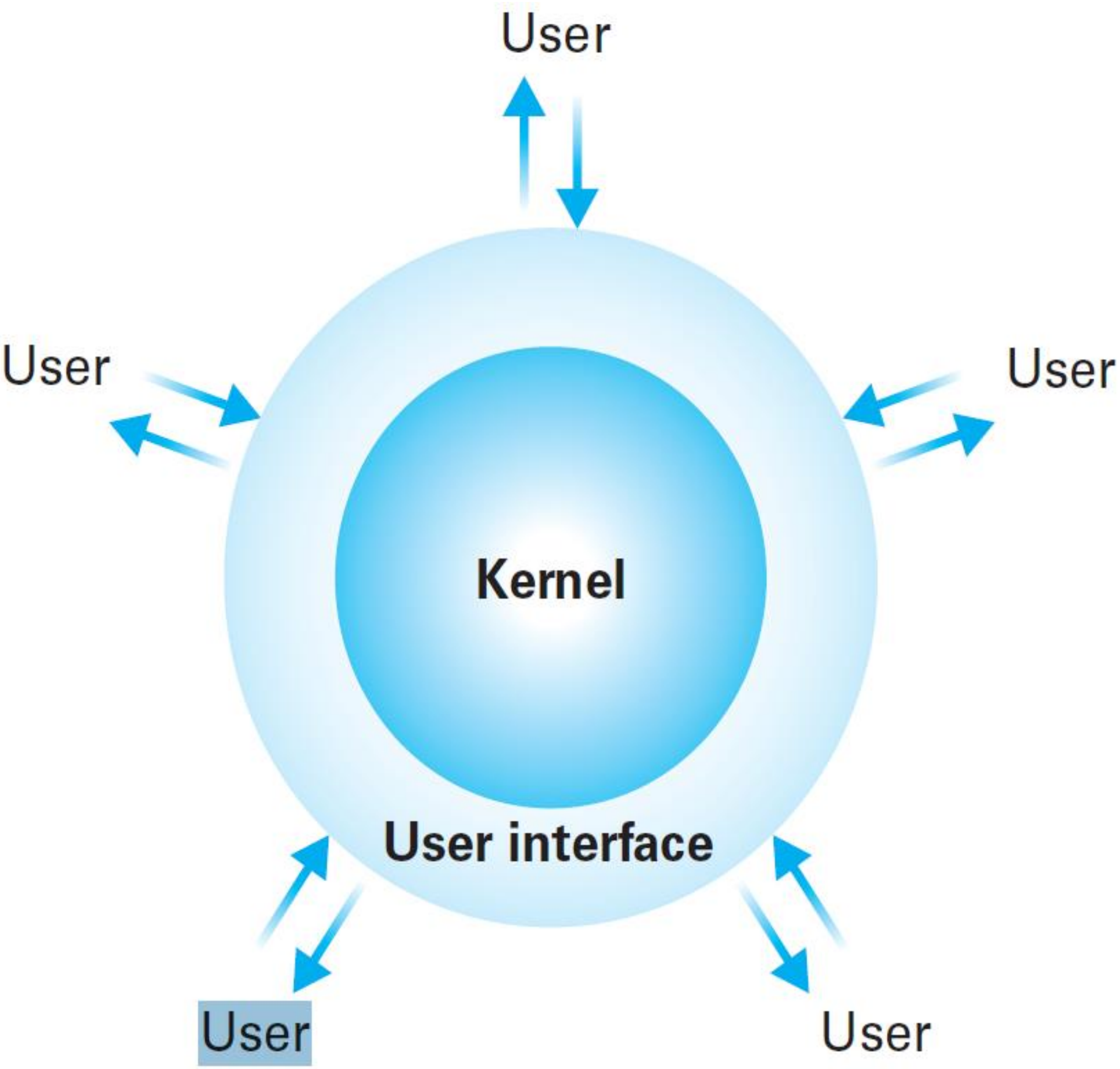
Categories of software

- **System software**
  - **Operating System**
  - **Utility Software**
    - **CD writing software**
    - **Disk Formatting software**
    - **Compressing and decompressing software (winzip)**
    - **Multimedia players**
    - **S/w to handle network communication (Bluetooth, Wifi etc.)**

Categories of software

- Need for isolating utility software from OS
  - To allow customization
  - Needs of a particular installation
- Distinction between
  - Application and utility software and
  - Utility software and OS is vague

Components of OS



Components of OS

- Communication with users
  - Through user interface
  - Shells – (old textual (cmd) interfaces)
    - Communication via keyboard and monitor
  - GUI – (modern, graphical interfaces)
    - Communication via mouse, stylus (pointing device), touch screens
- Research in user interfaces
  - 3D projection systems
  - Tactile sensory devices
  - Surround sound audio reproduction systems

Types of shells

- Unix
  - Bourne shell
  - C shell
  - Korn shell
  - X11 (GUI)
- Windows
  - cmd.exe
- Apple OS X
  - terminal utility shell

Window Manager

- An important component within today’s GUI shells
  - allocates blocks of space on the screen called windows
  - keeps track of which application is associated with each window
  - responsible for “style” of a GUI
  - most managers offer a range of configurable choices.
  - Example KDE and Gnome window managers in Linux

Kernel

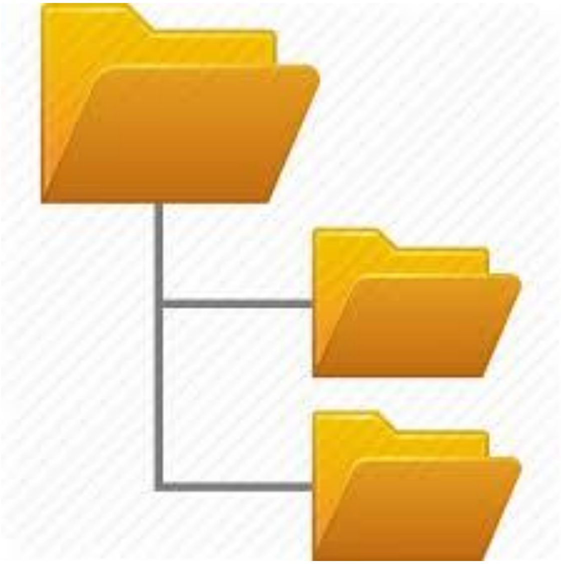
- Internal part of an operating system
  - Contains software components that perform the very basic functions required by the computer installation
  - Example: File Manager
    - Coordinates the use of the machine's mass storage facilities
    - Maintains records of all the files stored in mass storage
    - Maintains Location of each file
    - Contains access information of each file
    - Portions of mass storage are available for new files
    - Extensions to existing files

File manager

- File information records are kept on the individual storage medium containing the related files
- Each time the medium is placed online, the file manager can retrieve the records shows what is stored on that particular medium.

File organization

- Related files are placed in a directory / folder
- Hierarchical organization using sub-directories
- Example utility for viewing – Windows explorer



Directory path

- Example: animals/prehistoric/dinosaurs

- For Windows users the slashes in such a path expression are reversed as in animals\prehistoric\dinosaurs

File access

- access to a file by other software units
  - obtained at the discretion of the file manager
  - Software requests access by a procedure called opening the file
  - If access approved by file manager,
    - Provides the information needed to locate and manipulate the file

Device Driver Collection

- Component of the kernel
- Software units that communicate with the controllers or directly with peripheral devices
- Carry out operations on the peripheral devices attached to the machine
- uniquely designed for its particular type of device

  Example: such as a printer, disk drive, or monitor
- translates generic requests into the more technical steps required by the device assigned to that driver

Device Driver - Example

- Printer’s device driver contains the software for reading and decoding
  - that particular printer’s status word as well as
  - all the other handshaking details
- Other software components do not have to deal with those technicalities to print a file
- Other components can merely rely on the device driver software to print the file and take care of the details

Advantages of Device Drivers

- Software units can be independent of the unique characteristics of particular devices

- Results in a generic operating system that can be customized for particular peripheral devices by merely installing the appropriate device drivers.

Memory manager

- Coordinates the machine's use of main memory
- Duties are minimal in single-tasking environment
- Current task is placed in a predetermined location in the memory, executed and replaced by the program for next task

Memory Manager

- Multi-user or multi-tasking environment
  - Duties are extensive
  - Computer has to address many needs at the same time
  - Many programs and data must reside simultaneously in memory
  - Memory manager must find and assign memory space for these needs
  - Ensure that the actions of each program are restricted to the program’s allotted space
  - As the needs of different activities come and go, the memory manager must keep track of those memory areas no longer occupied.

Paging

- Task of memory manager is complicated
  - When total main memory space required exceeds the space actually available in the computer

- Paging
  - The memory manager may create the illusion of additional memory space by rotating programs and data back and forth between main memory and mass storage

Paging - Example

- Suppose a main memory of 8GB is required but the computer only has 4GB
- To create the illusion of the larger memory space
  - the memory manager reserves 4GB of storage space on a magnetic disk
  - records the bit patterns that would be stored in main memory if main memory had an actual capacity of 8GB
  - This data is divided into uniform sized units called **pages** (typically a few kb in size)

Virtual Memory

- Memory manager
  - shuffles these pages back and forth between main memory and mass storage so that the pages that are needed at any given time are actually present in the 4GB of main memory
  - the computer is able to function as though it actually had 8GB of main memory
  - This large “fictional” memory space created by paging is called **virtual memory**

Scheduler and Dispatcher

- Two additional components within the kernel of an operating system
- Scheduler
  - Determines which activities are to be considered for execution
- Dispatcher
  - Controls the allocation of time to these activities

Boot Strapping (a.k.a booting)

- Procedure by which OS gets started
- Performed by computer when turned on
- OS transferred from mass storage to memory
- Bootstrap process
  - PC starts with a specific address when CPU turned on
  - CPU expects the beginning of the program to be executed
  - A small portion of a computer's main memory where the CPU expects to find its initial program is constructed from special non-volatile memory cells
- Such memory is known as **read-only memory (ROM)**
  - Contents can be read but not edited
  - Most ROM today are constructed as flash drives

Boot loader

- Permanently stored on machine’s ROM and

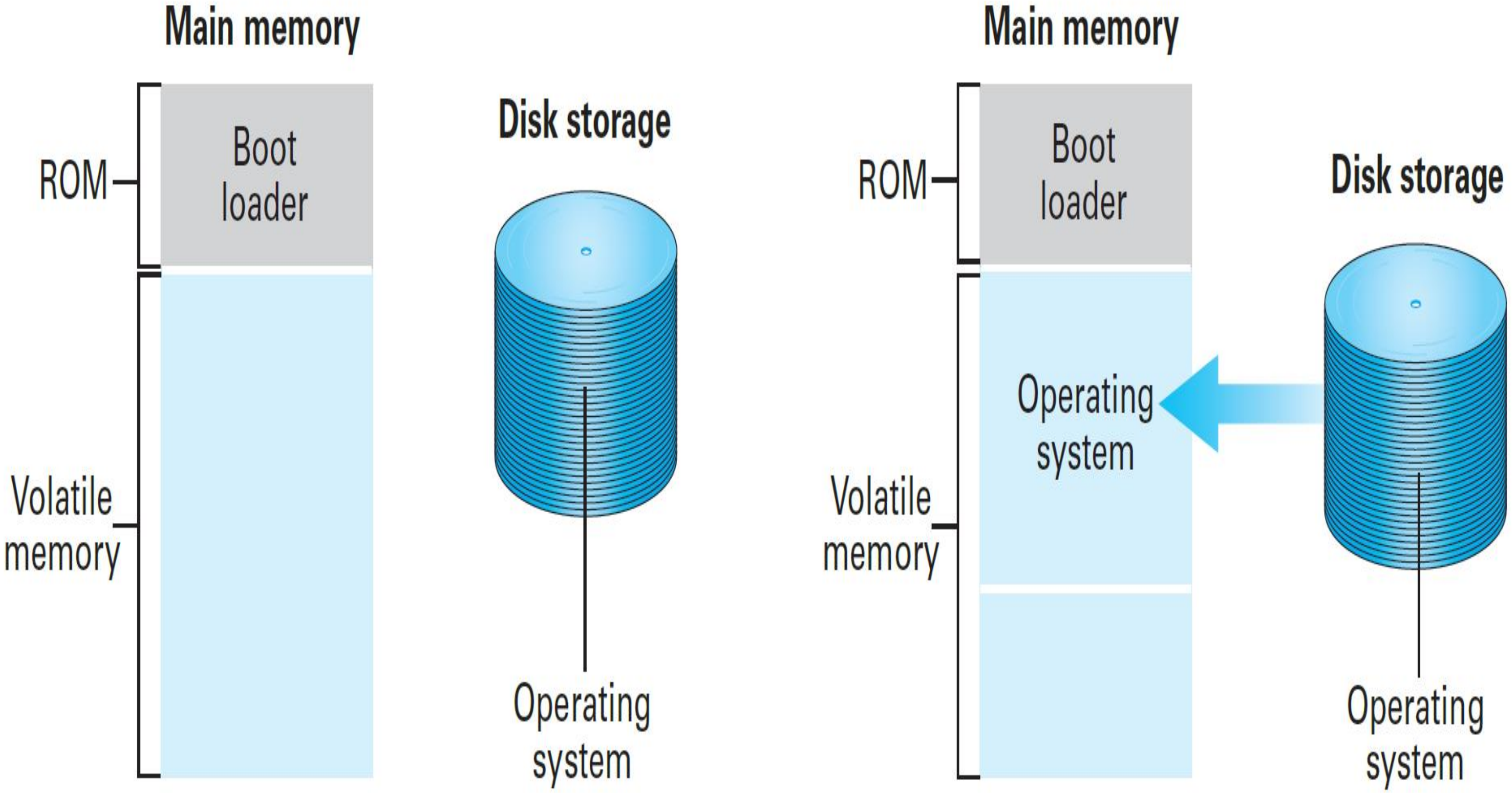
- Initially executed when machine is turned on

- The instructions in the boot loader direct the CPU to transfer the operating system from a predetermined location into the volatile area of main memory

Boot loader

- Modern boot loaders copy OS from variety of locations
  - – Embedded systems – special flash memory
  - – Small workstations at companies or universities – Distant machine over a network
- Once OS placed in main memory, bootloader directs CPU to execute a jump instruction to that area of memory
- OS takes over and controls the machine

Booting Process



Step 1: Machine starts by executing the boot loader program already in memory. Operating system is stored in mass storage.

Step 2: Boot loader program directs the transfer of the operating system into main memory and then transfers control to it.

Entire OS inside ROM

- Booting from mass storage would not be necessary
- Feasible for embedded systems with small operating systems
- Devoting large blocks of main memory in general-purpose computers to non-volatile storage is not efficient with today's technology

Entire OS inside ROM

- Computer operating systems undergo frequent updates in order to
  - maintain security and
  - keep abreast of new and improved device drivers for the latest hardware
- Firmware Update
  - Updating operating systems and boot loaders stored in ROM
  - Possible but technological limits make mass storage the common choice for traditional computing systems

Understanding the operation of general-purpose computers

- Requires
  - Understanding the boot process
  - Distinctions between an operating system,
  - Utility software and
  - Application software
- The above understanding allows one to comprehend the overall methodology

Overall process

- When a machine is first turned on
  - the boot loader loads and activates the operating system
  - The user then makes requests to the operating system regarding the utility or application programs to be executed
  - As each utility or application is terminated, the user is put back in touch with the operating system
  - In the above time the user can make additional requests

Learning to use a computing system

- Two-layered process
  - Learning the details of the specific utility or application desired
  - Also learn enough about the machine’s operating system to navigate among the applications

Coordination of Machine’s Activities

- How does an operating system coordinate the execution of
  - Application software,
  - Utility software, and
  - Units within the operating system itself

The Concept of a Process

- Program
  - A static set of directions
- Activity of executing a program
  - A dynamic activity whose properties change as time progresses
- This distinction is one of the most fundamental concepts of modern operating systems

Process and Process State

- Process
  - The activity of executing a program under the control of the operating system
- Process state
  - The current status of the activity (Associated with a process)
  - Includes the current position in the program being executed (the value of the program counter)
  - The values in the other CPU registers and
  - The associated memory cells

Process State

- A snapshot of the machine at a particular time
- At different times during the execution of a program (at different times in a process) different snapshots (different process states) will be observed.

Managing the Processes

- Typical time-sharing/multitasking computers run many processes
- All processes compete for machine's resources
- Task of the operating system
  - To manage these processes such that
    - Each process has the resources (peripheral devices, space in main memory, access to files, and access to a CPU) that it needs,
    - Independent processes do not interfere with one another, and
    - The processes which need to exchange information are able to do so.

Process Administration

- Coordination tasks of process execution handled by
  - Scheduler and
  - Dispatcher
- Both are Present within the operating system’s kernel

Functions of the Scheduler

- When a user requests the execution of an application, the scheduler adds the execution of that application to the pool of current processes and
  - Maintains a record of the processes present in the computer system
  - Introduces new processes to this pool, and
  - Removes completed processes from the pool.

Process Table

- A block of information in main memory
- Maintained by the scheduler
- To keep track of all the processes
- New process entry created by the scheduler in the process table for every execution of a program

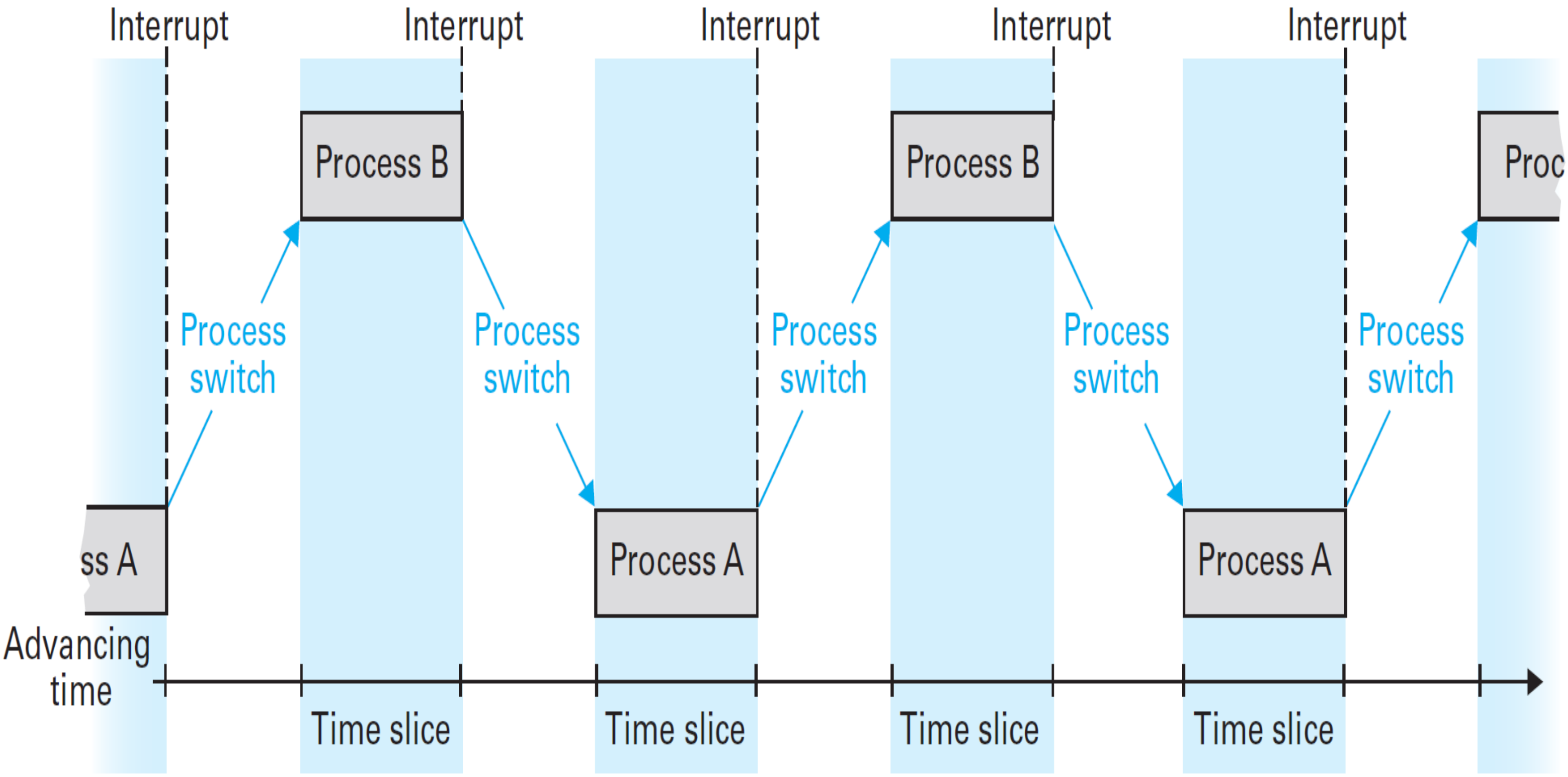
Process table entry

- Contains information such as
  - Memory area assigned to the process (obtained from memory manager)
  - Priority of the process
  - Whether the process is ready or waiting
  - Ready
    - Progress can continue
  - Waiting
    - Progress delayed due to some external operation
      - Mass storage operation, keypress, arrival of message from another process

Dispatcher

- Component of kernel
- Oversees execution of scheduled processes
  - By multiprogramming
    - Dividing time into short segments (time slice)
    - Measured in milliseconds or microseconds
  - Switches the CPU attention among processes for every time slice (process switch / context switch)

Multiprogramming between Process A and Process B



Interrupt

- Signal generated by a timer circuit indicating the end of a time slice
- Initiated by the dispatcher when a time slice is allotted to a process
- CPU reacts to an interrupt signal by
  - Completing the current machine cycle
  - Saves its position in current process
  - Begins executing an interrupt handler program

Interrupt Handler

- Is a part of dispatcher
- Describes how the dispatcher should respond to the interrupt signal
- Stored at a predetermined location in main memory

Effect of interrupt signal

- Pre-empt the current process
- Transfer control back to the dispatcher
- Dispatcher selects the process from the process table that has the highest priority among the ready processes (as determined by the scheduler)
- Restarts the timer circuit
- Allows the selected process to begin its time slice

Success of multiprogramming System

- Ability to stop, and later restart, a process
- Process environment (process's state) has to be recreated
  - Includes the value of the program counter
  - The contents of the registers and
  - Pertinent memory cells
- Multi-programming CPUs are equipped with this capability
  - Machine-language instructions for reloading a previously saved state (dispatcher's job is easier)

Salient features of multiprogramming

- Shuffling of processes required by multiprogramming introduces an overhead
- But use of multiprogramming is found to increase the overall efficiency of a machine
- Without multiprogramming
  - Each process runs to completion before the next process begins
  - The time that a process is waiting for peripheral devices to complete tasks or for a user to make the next request is wasted
  - This lost time is given to another process by multiprogramming

An example

- If a process executes an I/O request, say a request to retrieve data from a magnetic disk,
  - The scheduler updates the process table to reflect that the process is waiting for an external event.
  - The dispatcher will cease to award time slices to that process.
  - When the I/O request has been completed, the scheduler will update the process table to show that the process is ready, and that process again competes for time slices
  - In short, progress is made on other tasks while the I/O request is being performed,
  - Hence the entire collection of tasks will be completed in less time than if executed in a sequential manner.