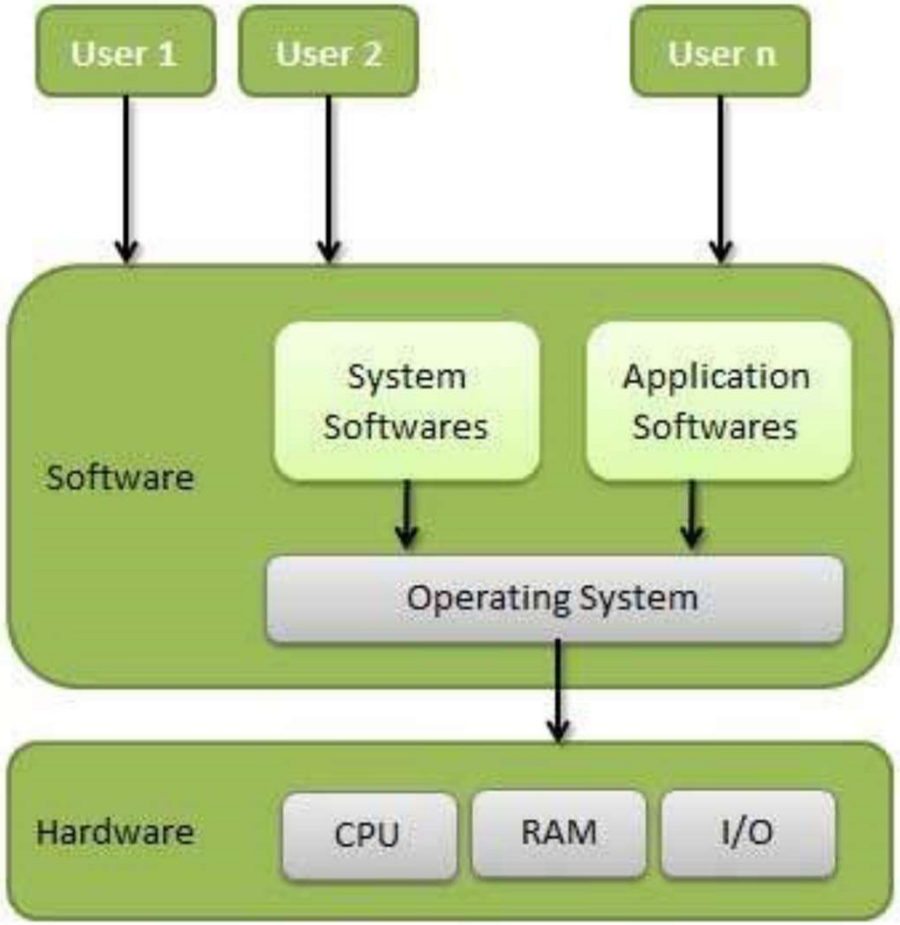
OPERATING SYSTEM

4th Semester CSE (R-23)

OUTLINE

➢An Operating System (OS) is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.



DEFINITION

An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Following are some of important functions of an operating System.

- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users

MEMORY MANAGEMENT

- Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address.
- Main memory provides a fast storage that can be accessed directly by the CPU.
- For a program to be executed, it must in the main memory.

An Operating System does the following activities for memory management :

- ☐ Keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
- ☐ In multiprogramming, the OS decides which process will get memory when and how much.
- ☐ Allocates the memory when a process requests it to do so.
- ☐ De-allocates the memory when a process no longer needs it or has been terminated.

PROCESSOR MANAGEMENT

In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called process scheduling. An Operating System does the following activities for processor management :

- Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.

DEVICE MANAGEMENT

An Operating System manages device communication via their respective drivers. It does the following activities for device management :

- Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

FILE MANAGEMENT

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directories.

An Operating System does the following activities for file management –

- Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.
- Decides who gets the resources.
- Allocates the resources.
- De-allocates the resources.

OTHER IMPORTANT ACTIVITIES

Following are some of the important activities that an Operating System performs :

- Security : By means of password and similar other techniques, it prevents unauthorized access to programs and data.
- Control over system performance : Recording delays between request for a service and response from the system.
- Job accounting : Keeping track of time and resources used by various jobs and users.

OTHER IMPORTANT ACTIVITIES

- Error detecting aids : Production of dumps, traces, error messages, and other debugging and error detecting aids.
- Coordination between other software and users : Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

TYPES OF OPERATING SYSTEM

A lot of operating system has been designed as per the necessity of users. We will summarize some of them here.

Batch system:

☐ Early computers were enormous machine run from a console. The users did not interact directly with the computer systems. To speed up processing, operators batched jobs together with similar needs and ran them through the computer as a group. The operator would sort programs into batches with similar requirements and as the system became available would run each batch. The output from each job would be sent back to the programmer.

TYPES OF OPERATING SYSTEM

- In the execution environment the CPU is often idle, as the speed of mechanical IO devices are much slower than those of electronic devices. The difference in speed makes the CPU wait for a long time always.
- Later on the introduction of disk technology allowed the operating system to keep all jobs on a disk and to perform better.

MULTI PROGRAMMED SYSTEMS:

- The most important aspect of job scheduling is the ability to multiprogramming. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute.
- The Operating System keeps several jobs in memory simultaneously. These set of jobs are the part of jobs reside in the job pool of the disk. These jobs can be executed one by one. The Operating System can assign another job to the CPU whenever the running process needs any IO, so that the CPU can be used in a optimal way.

MULTI PROGRAMMED SYSTEMS:

- Multiprogramming is the first instance where the Operating System must make decisions for the users. The jobs reside in the job pool can be chosen by the Operating System through different scheduling criteria and scheduling algorithms.

TIME SHARING SYSTEMS:

- Time sharing (multi tasking) is a logical extension of multiprogramming. The CPU executes multiple jobs by switching among them, but the switches occur so frequently that the user can interact with each program while it is running.
- An interactive computer system provides direct communication between the user and the system. So the response time of it should be short typically with in 1 second. A time shared operating system allows many users to share the computer simultaneously. It uses CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared computer.

TIME SHARING SYSTEMS:

- Here several jobs may be kept simultaneously in memory, so the system may have memory management and protection. To obtain a reasonable response time jobs may have to be swapped in and out of main memory.
- A common method virtual memory is implemented to achieve this goal.

MULTIPROCESSOR SYSTEM:

Multiprocessor system also known as parallel system or tightly coupled system have more than one processor. In close communication, sharing the computer bus, the clock and sometimes memory and peripheral devices.

Advantage:

- Increased throughput: by increasing the number of processors, we hope to get more work done in less time, whereas it also incurs some overhead to maintain the processors.

MULTIPROCESSOR SYSTEM:

- Economy of scale: we can save money by sharing some system resources.
- Increased reliability: if functions can be distributed properly among several processors, then the failure of one processor will not halt the system, only slow it down.
- The ability to continue service providing proportional to the level of surviving hardware is called graceful degradation. Systems designed for graceful degradation are also fault tolerant systems.

MULTIPROCESSOR SYSTEM:

- The most common multiple processor systems now use symmetric multiprocessing (SMP), in which each processor runs an identical copy of the operating system and these copies communicate with one another as needed. Some systems use asymmetric multiprocessing, in which each processor is assigned a specific task. A master processor controls the system, other processors either look to the master for instruction or have predefined task. This scheme defines a master slave relationship.

DISTRIBUTED SYSTEMS:

A network is a communication path between two or more systems. Distributed systems depend on network for their functionality. By being able to communicate, distributed systems are able to share computational tasks and provide a rich set of features to users. As a part of this centralized systems today act as server systems to satisfy requests generated by client systems.

- Server systems can be broadly categorized as compute server and file server

DISTRIBUTED SYSTEMS:

Compute server system: provide an interface to which clients can send requests to perform an action, in response to which they execute the action and send back results to the client.

File server system: provides a file system interface where clients can create, update, delete and read files.

The computer network used in the distributed operating system does not share a memory or a clock instead, each processor has its own local memory. The processors communicate with each other through various communication lines, such as high-speed buses or telephone lines. These systems are usually referred to as loosely coupled system.

REAL TIME SYSTEM:

A real time system is used when rigid time requirements have been placed on the operations of a processor or the flow of data, thus it is often used as a control device in a dedicated application.

- Sensors being data to the computer. The computer must analyze the data and possibly adjust controls to modify the sensor inputs. For e.g. scientific experiments, medical imaging systems etc.
- A real time system has well defined fixed time constraints, hence processing must be done within the defined constraints or the system fails.

REAL TIME SYSTEM:

Real time system comes in two flavors :

- Hard real time system guarantees that the critical task be completed on time. This goal requires that all delays in the system be bounded from the retrieval of stored data to the time that it takes the operating system to finish any request made of it.
- Soft real time system is less restrictive compared to hard real time system, where a critical real time task gets priority over other tasks and retain that priority until it completes.

REAL TIME SYSTEM:

- The soft real time systems are useful however in several areas including multimedia, virtual reality and advanced scientific projects.
- Virtual memory almost never found in real time systems.

THE COMPUTER SYSTEM STRUCTURE RELATED TO OPERATING SYSTEM

- A modern general-purpose system consists of a CPU and a number of device controllers that are connected through a common bus, which provides access to share memory. Each device control is in charge of a specific type of device.
- To start a program or a computer system we need a bootstrap program, which is stored in read only memory (ROM) or in EEPROM as firmware. It initializes all aspects of the system from CPU registers to device controller to the memory contents. After execution of the bootstrap program the OS starts executing the first process "init" and wait for some event to occur.

THE COMPUTER SYSTEM STRUCTURE RELATED TO OPERATING SYSTEM

- The occurrence of an event is usually signaled by an interrupt from either the hardware or software.
- Software may trigger an interrupt by executing a special operation called a system call or monitor call. When the CPU is interrupted it stops what it is doing and immediately transfers execution to a fixed location, which actually contains the starting address where the service routine for the interrupt is located.
- The information about the interrupts may be stored in a table of pointer which is generally stored in low memory (generally the first hundred location).

THE COMPUTER SYSTEM STRUCTURE RELATED TO OPERATING SYSTEM

- These locations hold the address of the interrupt service routines for the various devices. This array or interrupt vector of addresses is then indexed by a unique device number given with the interrupt request, to provide the address of the interrupt service routine for the interrupting device.
- A system call is invoked in a variety of ways, depending on the functionality provided by the processor. In all forms it is the method used by a process to request action by the operating system.

THE COMPUTER SYSTEM STRUCTURE RELATED TO OPERATING SYSTEM

- IO structure: A device controller is in charge of a specific type of device or may be attached to more than one device. The device controller is responsible for moving the data between the peripheral devices that it controls and its local storage buffer.
- IO interrupts: To start an IO operation, the CPU loads the appropriate register with in the device controller, in turn the device controller examines the registers to find the action, that is READ or WRITE.
- For a read operation it starts transfer of data from the device to its local buffer and after completion it informs the CPU. The information passing can be done by triggering an interrupt.

OPERATING SYSTEM - SERVICES

An Operating System provides services to both the users and to the programs.

- It provides programs an environment to execute.
- It provides users the services to execute the programs in a convenient manner.

Following are a few common services provided by an operating system:

- ☐ Program execution
- ☐ File System manipulation
- ☐ Error Detection
- ☐ Protection
- ☐ I/O operations
- ☐ Communication
- ☐ Resource Allocation

OPERATING SYSTEM - SERVICES

- Program execution : Operating systems handle many kinds of activities from user programs to system programs. Following are the major activities of an operating system with respect to program management:

- ☐ Loads a program into memory.
- ☐ Executes the program.
- ☐ Handles program's execution.
- ☐ Provides a mechanism for process synchronization.
- ☐ Provides a mechanism for process communication.
- ☐ Provides a mechanism for deadlock handling.

OPERATING SYSTEM - SERVICES

I/O Operation: An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users.

- An Operating System manages the communication between user and device drivers.
- I/O operation means read or write operation with any file or any specific I/O device.
- Operating system provides the access to the required I/O device when required.

OPERATING SYSTEM - SERVICES

File system manipulation : A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long term storage purpose.

- A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directories.
- Following are the major activities of an operating system with respect to file management:

OPERATING SYSTEM - SERVICES

- Program needs to read a file or write a file.
- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.
- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file system.

COMMUNICATION

The OS handles routing and connection strategies, and the problems of contention and security. Following are the major activities of an operating system with respect to communication:

- Two processes often require data to be transferred between them
- Both the processes can be on one computer or on different computers, but are connected through a computer network.
- Communication may be implemented by two methods, either by Shared Memory or by Message Passing.

ERROR HANDLING

Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling:

- ☐ The OS constantly checks for possible errors.
- ☐ The OS takes an appropriate action to ensure correct and consistent computing.

RESOURCE MANAGEMENT

In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management:

- The OS manages all kinds of resources using schedulers.
- CPU scheduling algorithms are used for better utilization of CPU.

PROTECTION

Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system. Following are the major activities of an operating system with respect to protection:

- ☐ The OS ensures that all access to system resources is controlled.
- ☐ The OS ensures that external I/O devices are protected from invalid access attempts.
- ☐ The OS provides authentication features for each user by means of passwords

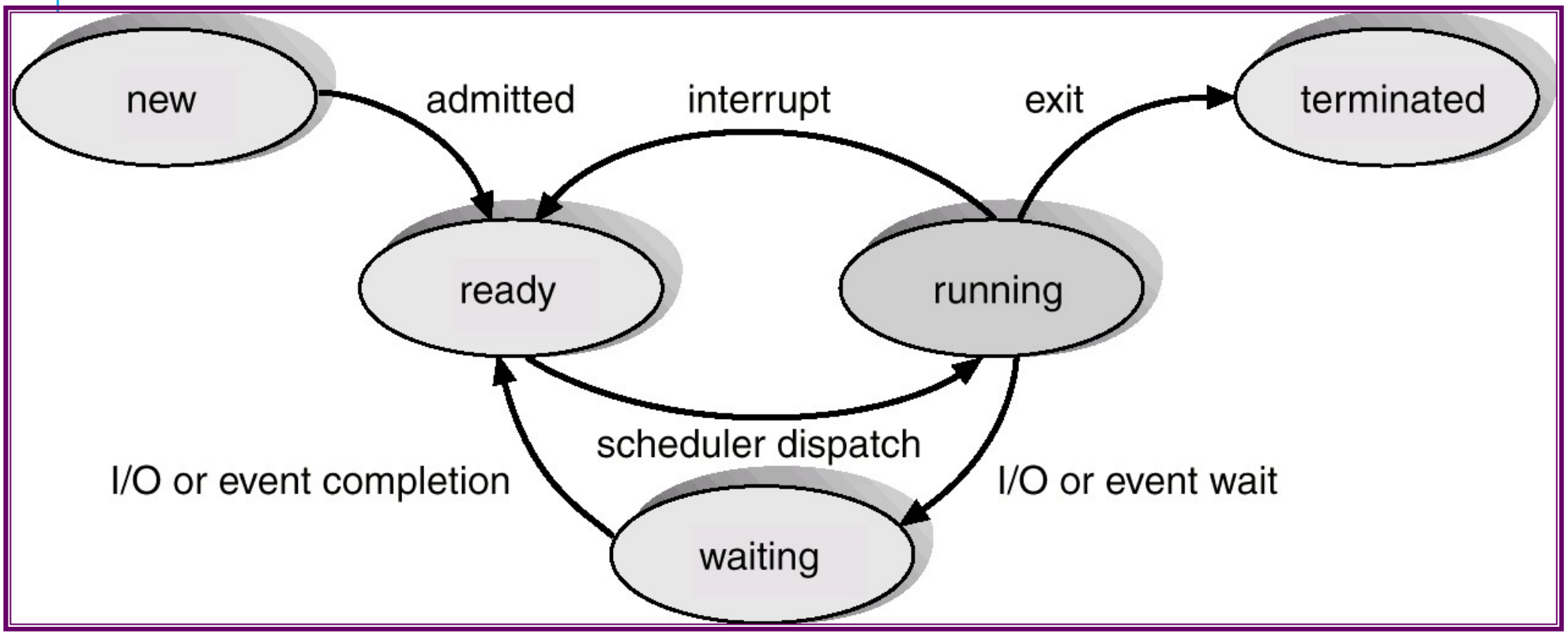
PROCESS

- A process can be thought of as a program in execution. It needs some resources to continue its work such as CPU time, memory, files and I/O devices.
- We can say that the program itself is not a process. A program is a passive entity such as the contents of a file stored on disk, where as a process is an active entity with a program counter specifying the next instruction to execute and a set of associated resources attached to it.

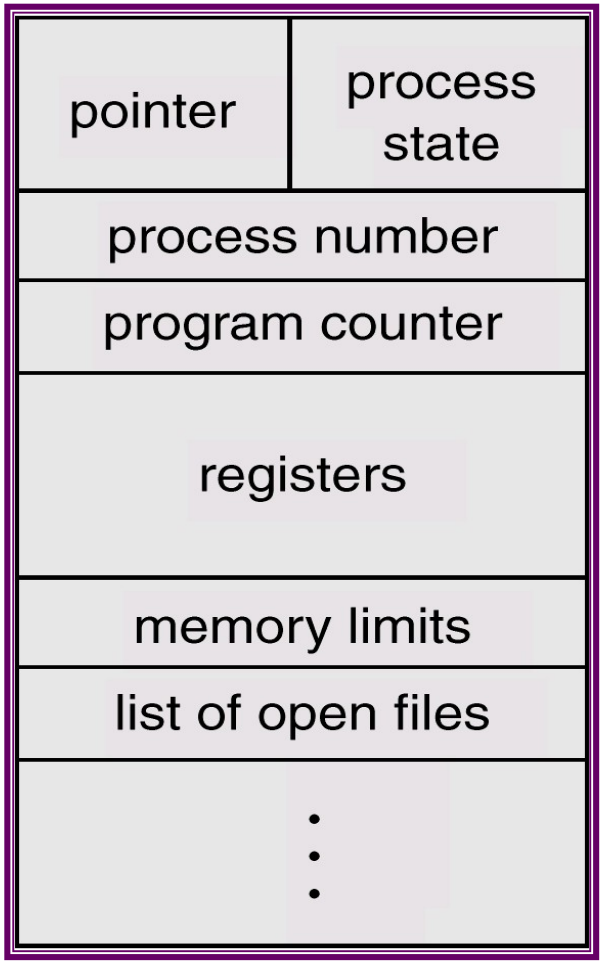
PROCESS STATE:

- The state of a process is defined in parts by the current activity of the process, such as
  - New: The process is being created.
  - Running: instructions are being executed.
  - Waiting: the process is waiting for some event to occur.
  - Ready: waiting to be assigned to some process.
  - Terminated: the process has finished execution
- Only one process can be running on any processor at any instance, although many processes may be ready and waiting.

PROCESS STATE:



PROCESS CONTROL BLOCK:



☐ Each process is represented in the OS by a process control block. The PCB contains all information of a specific process associated with it such as:

- Process state: the state may be new, ready, running, waiting, or halted.
- Program counter: the counter indicates the address of the next instruction to be executed.

PROCESS CONTROL BLOCK:

- CPU registers: they are many types such as: index registers, stack pointers, general purpose registers etc..
- CPU scheduling information: process priority number, pointers to scheduling queues etc.
- Memory management information: such as base and limit registers, page table and segment table information etc.
- Accounting information: includes the amount of CPU and real time used, time limits, job or process nos. etc.
- I/O status information: the information includes the list of I/O devices allocated to the process, a list of open files etc.

SCHEDULERS:

A process migrates between various scheduling queues throughout its life time. The selection of process for the scheduling purpose is done by the scheduler.

In a batch system often more processes are submitted then can be executed immediately. These processes are spooled to a mass storage for later execution.

The scheduler may be categorized into three categories:

- long term scheduler
- short term scheduler
- medium term scheduler

SCHEDULERS:

long term scheduler : The long-term scheduler selects process from the spooled processes kept in mass storage for later execution and loads them into memory for execution.

It controls the degree of multiprogramming by looking the rate of creation and termination rates of processes thus, it may be invoked only when a process is leaving the system after its completion.

As it has lot of time to select a process it should careful selection according to its nature, as the process may be I/O bound (spends more time doing I/O) or CPU bound (spends more time for computation) so a long-term scheduler should select a good process mix of i/o bound and CPU bound to balance the load of CPU.

SCHEDULERS:

Short Term Schedulers:

- After submission of the process in memory the short term scheduler will take over the responsibility.
- By the help of dispatcher, the short-term scheduler choose a process from the ready queue to give the control of CPU by using some scheduling algorithm.

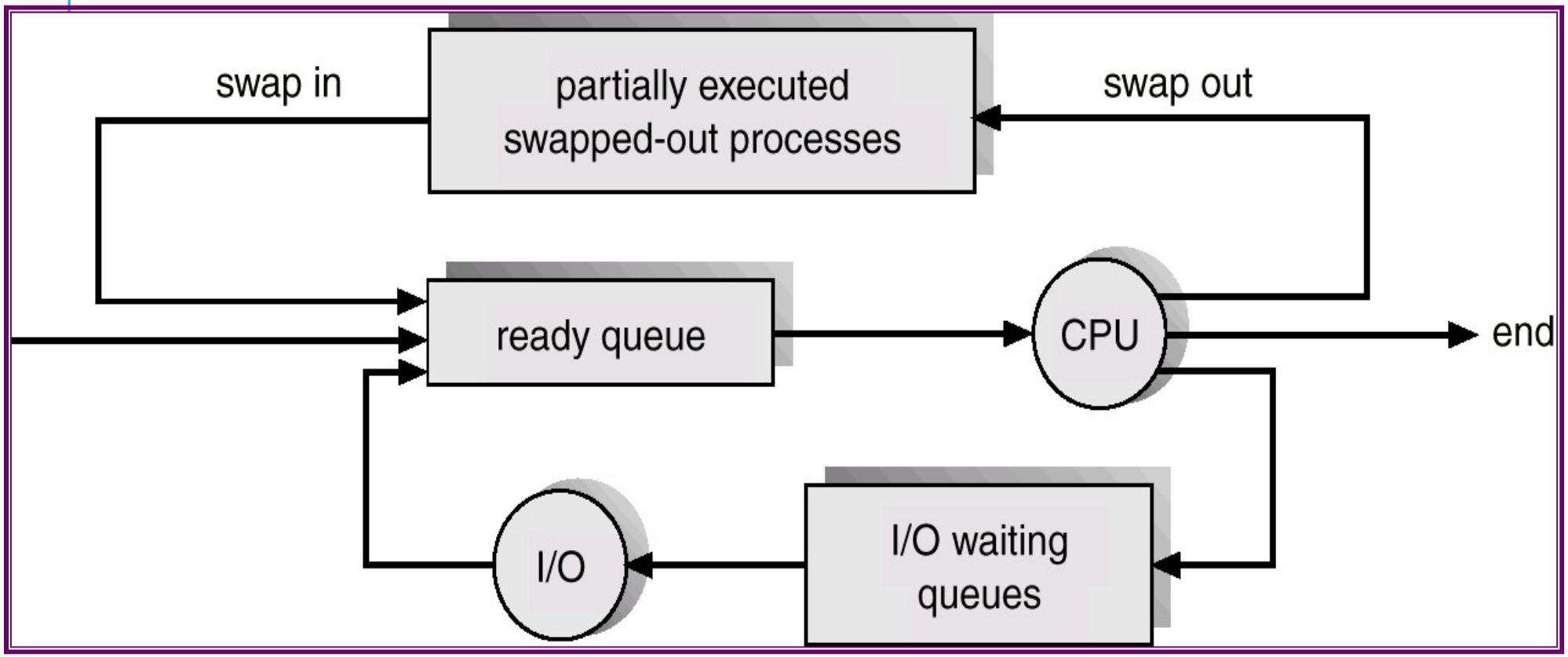
SCHEDULERS:

Medium term scheduler :

On some systems such as a time-shared system may introduce an intermediary level of scheduler called medium-term scheduler.

When there is lack of main memory at the time of execution due to the virtual memory implementation that is when the required memory for execution is larger than the available memory then some process lies in the waiting queue may be swapped out to make room for a currently executing process. This job is done by the medium-term scheduler.

SCHEDULERS:


swap in
partially executed swapped-out processes
swap out
ready queue
CPU
end
I/O
I/O waiting queues

CONTEXT SWITCH:

Switching the CPU to another process requires saving the state of the old process and loading the saved state for the new process. This task is known as context switch.

- The information about the context is stored inside the PCB of a process.
- Context switch is pure overhead because the system does no useful work while switching.
- Context switch times are highly dependent on hardware support.

COMMUNICATION BETWEEN PROCESSES (INTER PROCESS COMMUNICATION):

A concurrent process can be of two types:

- Independent: a process is independent if it cannot affect or be affected by other processes executing in the system.
- Co-operative: a process is said to be co-operating if it can affect or be affected by other processes.

The inter process communication (IPC) in Co-operating processes can be done through two models:

- ☐ Shared memory
- ☐ Message passing.

COMMUNICATION BETWEEN PROCESSES (INTER PROCESS COMMUNICATION):

Shared Memory: Inter process communication using shared memory requires communicating processes to establish a shared memory region.

- A shared memory region resides in the address space of the process creating the shared memory segment. Other processes that wish to communicate using this shared memory segment must attach themselves to the address space of the creator's address space.
- They can then exchange information by reading and writing data in the shared area. The form and the locations determined by these processes are not under the control of the operating system. The processes are also responsible for ensuring that they are not writing to the same location simultaneously.

COMMUNICATION BETWEEN PROCESSES (INTER PROCESS COMMUNICATION):

Message Passing:

Message passing provides a mechanism to allow processes to communicate and to synchronize their actions without sharing the same address space and is useful in distributed environment, where the communication processes may reside on different computers connected by a network.

A message passing facility provides at least two operations

- Send
- Receive

COMMUNICATION BETWEEN PROCESSES (INTER PROCESS COMMUNICATION):

Message sent by processes may either fixed or variable size. If the message is of fixed size the system level implementation becomes easy, this restriction however makes the task of programming more difficult. Whereas for a variable size message system level implementation is difficult with programming task simpler.

In order to send and receive message from each other, the process must have a communication link existing between them. The link can be implemented in many ways. Logically a link can be implemented using the following methods:

- ☐ Direct or indirect communication
- ☐ Synchronous or asynchronous communication
- ☐ Automatic or explicit buffering.

DIRECT COMMUNICATION:

In this method each process which wants to communicate must name explicitly the sender or receiver of the communication, and the send () and receive () primitives are defined as follows

- Send (P, message): send a message to process P
- Receive (Q, message): receive a message from process Q

A communication link in this method will have the following properties:

- Automatic link establishment take place between every pair of processes which wants to communicate and the processes are required to know only each other's identity for communication.

DIRECT COMMUNICATION:

- Only two processes are associated with a link
- There will be only one link between each pair of processes.

The discussed mechanism of direct communication exhibits symmetry in addressing, that is both the sender and the receiver process must name other to communicate whereas a variant of this scheme employs asymmetry in addressing where the send and receive primitives may be defined as:

- Send (P, message): send a message to process P
- Receive (ID, message): receive a message from any process; the variable id is set to the name of the process with which communication has taken place.

DIRECT COMMUNICATION:

Disadvantage of symmetric and asymmetric schemes are the limited modularity of the resulted process definition, that is a process name changing may require examining all other process definition and all the references to the old name must be found, so that they can be modified to the new name.

INDIRECT COMMUNICATION:

- In this scheme the sending and receiving of message is done using a mail box (also called as ports). A mailbox abstractly can be viewed as an object, where the processes place their message and removes the messages from it.
- Each mailbox will have a unique identification and a process can communicate with some other processes through number of different mailboxes.
- Communication between two processes is possible only if they have a shared mailbox. In this scheme the send and receive primitives are defined as:
  - Send (B, message): send a message to mailbox B
  - Receive (B, message): receive a message from mailbox B

INDIRECT COMMUNICATION:

The communication link in this scheme has the following properties:

- A link can be established between a pair of processes only if they have a shared mail box.
- More than two processes can be associated with a link.
- There can be a number of different links between each pair of communicating processes and each link corresponds to one mailbox.
- A mailbox can be owned by a process or by the operating system.

INDIRECT COMMUNICATION:

- If it is owned by a process then the process that creates the mail box can receive the message from that mail box and others can send messages to this mail box. A mail box disappears when the process that created the mailbox terminates and the other processes those tries to send message to the mailbox must be notified about the absence of the mailbox.
- If a mailbox is owned by the operating system, it has its own existence. It is independent and is not attached to any process.

INDIRECT COMMUNICATION:

Operating system provides a mechanism using which a process.

- Can create a new mailbox
- Can send and receive messages through the mailbox.
- Can destroy a mailbox

The process who creates the mailbox is the owner of the mailbox by default. The ownership and receive privilege can be passed to other processes through some system calls.

SYNCHRONIZATION:

Message passing may be either blocking or nonblocking also known as synchronous or asynchronous

- Blocking send: the sending process is blocked until the message is received by the receiving process or by the mailbox.
- Non blocking send: the sending process sends the message and resumes its operation.
- Blocking receive: the receiver blocks until a message is available.
- Non blocking receive: the receiver retrieves either a valid message or a null

BUFFERING:

Whether communication is direct or indirect, messages resides in a temporary queue called buffer. Basically, such queues are implemented in three ways:

- Zero capacity: here the link cannot have any message waiting in it, that is the sender must block until the receiver receives the message.
- Bounded Buffer: the queue has finite length n, thus can keep up to n messages, hence if the link is full the sender must block until space is available in the queue.
- Unbounded buffer: the queues length is potentially infinite; thus, any number of messages can wait in it, hence is known as automatic buffering.

PROCESS SCHEDULING

CPU scheduling is the basis of multi-programmed operating system. The computer can be made productive by switching the CPU among processes. The objective of multiprogramming is to have some process running at all times.

- On a system when a process is executing it's I/O the CPU sits idle so if we use this time productively then we can increase the CPU utilization.
- The solution is to keep several processes in memory at one time. When one process is busy with the I/O the control of the CPU is given to another process in the memory that is ready to execute.

PROCESS SCHEDULING

- A process execution generally alternates between the CPU execution and I/O wait cycle.
- Process execution always starts with a CPU burst and is followed by an I/O burst and so on and always completes with a CPU burst with a system request to terminate execution at the completion of a process execution.

CPU scheduler: Whenever the CPU becomes idle, the operating system must select one of the processes in the ready queue to be executed. The selection process is carried out by the short time scheduler by the help of some scheduling algorithm.

SCHEDULING:

Generally, the scheduling can be done in two ways.

- Non-Preemptive: Here once the CPU has been allocated to a process, the process keeps the CPU until it release the CPU by terminating itself after completion of execution or by switching to the waiting state.
- Preemptive: here the control of CPU can be taken any time due to some reason as a process may be preempted if a higher priority process arrives or due to a system call interruption etc.

SCHEDULING:

- In a time-sharing system a time slice is used, so after expiration of the time slot the process may be preempted. Here all processes are got the same time slot for its execution.
- The preemptive scheduling has greater performance on non-preemptive, but in some cases an inconsistent result may occur due to the preemptive scheduling, hence the non-preemptive scheduling may be used for better result.

SCHEDULING:

- CPU-scheduling decisions may take place under the following four circumstances.
  - When a process switches from the running state to waiting state (as a result of an i/o request or invocation of wait for the termination of one of the child process)
  - When a process switches from the running state to ready state (when an interrupt occurs)

SCHEDULING:

- When a process switches from the waiting state to ready state (at completion of i/o)
- When a process terminates.

When scheduling takes place only under circumstances 1 and 4, we say the scheduling scheme is non-preemptive as it does not have any choice, however for situations 2 and 3 we have some choice, hence can be preemptive.

DISPATCHER:

The dispatcher is the module that gives control of the CPU to the process selected by the short-term scheduler which includes certain functions as:

- Switching context
- Switching to user mode
- Jumping to the proper location in the user program.

The time taken by the dispatcher to stop one process and start another is known as dispatch latency.

DISPATCHER:

The dispatcher is the module that gives control of the CPU to the process selected by the short-term scheduler which includes certain functions as:

- Switching context
- Switching to user mode
- Jumping to the proper location in the user program.

The time taken by the dispatcher to stop one process and start another is known as dispatch latency.