Hierarchical architecture views the whole system as a hierarchy structure, in which the software system is decomposed into logical modules or subsystems at different levels in the hierarchy. This approach is typically used in designing system software such as network protocols and operating systems.

In system software hierarchy design, a low-level subsystem gives services to its adjacent upper level subsystems, which invoke the methods in the lower level. The lower layer provides more specific functionality such as I/O services, transaction, scheduling, security services, etc. The middle layer provides more domain dependent functions such as business logic and core processing services. And, the upper layer provides more abstract functionality in the form of user interface such as GUIs, shell programming facilities, etc.

It is also used in organization of the class libraries such as .NET class library in namespace hierarchy. All the design typ

architecture and often combine with other architecture styles.

Hierarchical architectural styles is divided as –

- Main-subroutine
- Master-slave
- Virtual machine

Main-subroutine

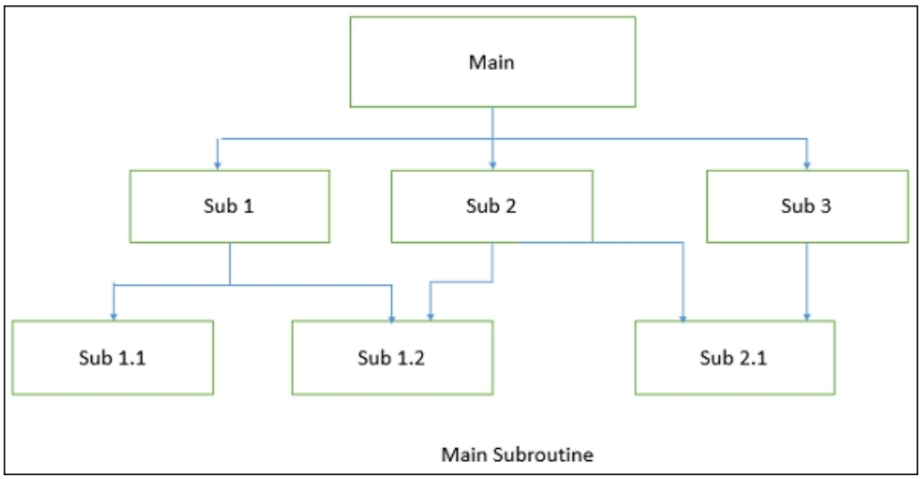
The aim of this style is to reuse the modules and freely develop individual modules or subroutine. In this style, a software system is divided into subroutines by using top-down refinement according to desired functionality of the system.

These refinements lead vertically until the decomposed modules is simple enough to have its exclusive independent responsibility. Functionality may be reused and shared by multiple callers in the upper layers.

There are two ways by which data is passed as parameters to subroutines, namely –

- **Pass by Value** – Subroutines only use

- **Pass by Value** – Subroutines only use the past data, but can't modify it.
- **Pass by Reference** – Subroutines use as well as change the value of the data referenced by the parameter.



Advantages

- Easy to decompose the system based on hierarchy refinement.
- Can be used in a subsystem of object oriented design.

Disadvantages

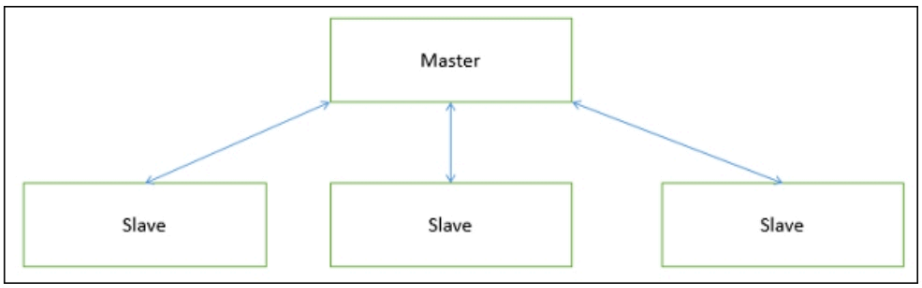
- Vulnerable as it contains globally shared data.
- Tight coupling may cause more ripple effects of changes.

Master-Slave

Master-Slave

This approach applies the 'divide and conquer' principle and supports fault computation and computational accuracy. It is a modification of the main-subroutine architecture that provides reliability of system and fault tolerance.

In this architecture, slaves provide duplicate services to the master, and the master chooses a particular result among slaves by a certain selection strategy. The slaves may perform the same functional task by different algorithms and methods or totally different functionality. It includes parallel computing in which all the slaves can be executed in parallel.



The implementation of the Master-Slave pattern follows five steps –

- Specify how the computation of the task can be divided into a set of equal sub-tasks and identify the su

The implementation of the Master-Slave pattern follows five steps −

- Specify how the computation of the task can be divided into a set of equal sub-tasks and identify the sub-services that are needed to process a sub-task.
- Specify how the final result of the whole service can be computed with the help of the results obtained from processing individual sub-tasks.
- Define an interface for the sub-service identified in step 1. It will be implemented by the slave and used by the master to delegate the processing of individual sub-tasks.
- Implement the slave components according to the specifications developed in the previous step.
- Implement the master according to the specifications developed in step 1 to 3.

Applications

- Suitable for applications whe

Applications

- Suitable for applications where reliability of software is critical issue.
- Widely applied in the areas of parallel and distributed computing.

Advantages

- Faster computation and easy scalability.
- Provides robustness as slaves can be duplicated.
- Slave can be implemented differently to minimize semantic errors.

Disadvantages

- Communication overhead.
- Not all problems can be divided.
- Hard to implement and portability issue.

Virtual Machine Architecture

Virtual Machine architecture pretends some functionality, which is not native the hardware and/or software on which it

Virtual Machine Architecture

Virtual Machine architecture pretends some functionality, which is not native to the hardware and/or software on which it is implemented. A virtual machine is built upon an existing system and provides a virtual abstraction, a set of attributes, and operations.

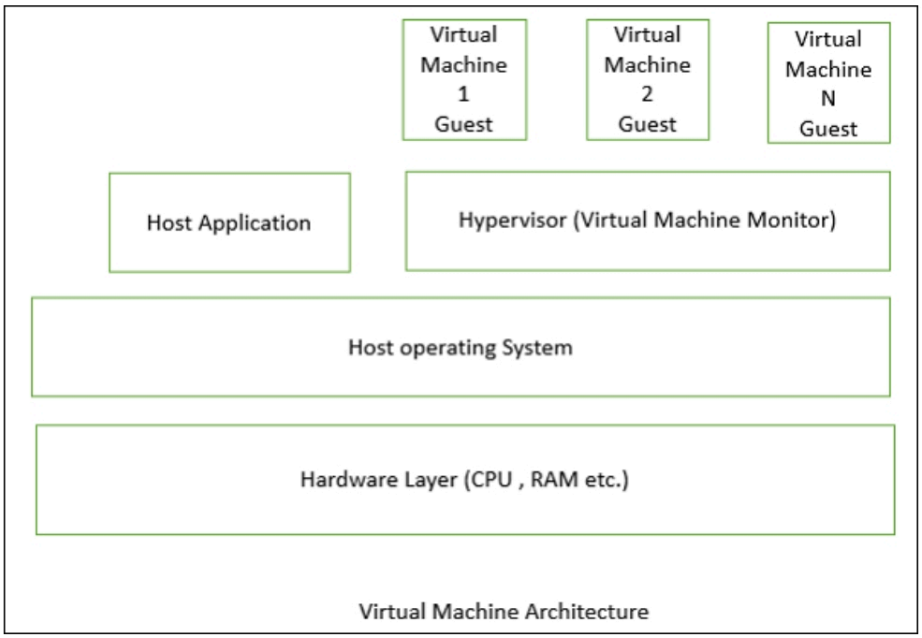
In virtual machine architecture, the master uses the ‘same’ subservice’ from the slave and performs functions such as split work, call slaves, and combine results. It allows developers to simulate and test platforms, which have not yet been built, and simulate "disaster" modes that would be too complex, costly, or dangerous to test with the real system.

In most cases, a virtual machine splits a programming language or application environment from an execution platform. The main objective is to provide **portability**. Interpretation of a particular module via a Virtual Machine may be perceived as −

- The interpretation engine chooses an

- The interpretation engine chooses an instruction from the module being interpreted.
- Based on the instruction, the engine updates the virtual machine's internal state and the above process is repeated.

The following figure shows the architecture of a standard VM infrastructure on a single physical machine.



The **hypervisor,** also called the **virtual machine monitor**, runs on the host OS and allocates matched resources to each guest OS. When the guest makes a system-call, the hypervisor intercepts a

allocates matched resources to each guest OS. When the guest makes a system-call, the hypervisor intercepts and translates it into the corresponding system-call supported by the host OS. The hypervisor controls each virtual machine access to the CPU, memory, persistent storage, I/O devices, and the network.

Applications

Virtual machine architecture is suitable in the following domains −

- Suitable for solving a problem by simulation or translation if there is no direct solution.
- Sample applications include interpreters of microprogramming, XML processing, script command language execution, rule-based system execution, Smalltalk and Java interpreter typed programming language.
- Common examples of virtual machines are interpreters, rule-based systems, syntactic shells, and command language processors.

- Common examples of virtual machines are interpreters, rule-based systems, syntactic shells, and command language processors.

Advantages

- Portability and machine platform independency.
- Simplicity of software development.
- Provides flexibility through the ability to interrupt and query the program.
- Simulation for disaster working model.
- Introduce modifications at runtime.

Disadvantages

- Slow execution of the interpreter due to the interpreter nature.
- There is a performance cost because of the additional computation involved in execution.

Layered Style

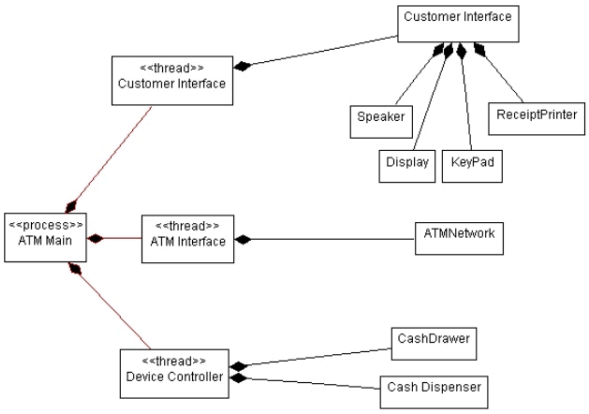
Concepts: Process View

To provide a basis for understanding the process organization of the system, an architectural view called the **process view** is used in the Analysis & Design discipline. There is only one process view of the system, which illustrates the process decomposition of the system, including the mapping of classes and subsystems on to processes and threads. The process view is refined during each iteration. As [BOO98] states: "With UML, the static and dynamic aspects of this view are captured in the same kinds of diagrams as for the design view - i.e. class diagrams, interaction diagrams, activity diagrams and statechart diagrams, but with a focus on the active classes that represent these threads and processes." Of concern when constructing and using the process view are, for example, issues of concurrency, response time, deadlock, throughput, fault tolerance, and scalability.

It is possible to design for concurrency without the use of direct underlying operating system support - for example using a specially written scheduler or other run-time support. In such cases, concurrency is simulated at the application infrastructure level, rather than in the operating system. If necessary, other stereotypes (in addition to the standard threads and processes) may be used to make this distinction (to guide implementation). For example, the Ada programming language contains its own model of concurrency, based on Ada tasks; the Ada run-time has to provide this, whether or not the operating system on which it runs has an appropriate equivalent - threads, say - which could be used to support Ada tasking.

In real-time systems, the Rational Unified Process recommends the use of Capsules to represent active classes in the process view. Capsules have strong semantics to simplify the modeling of concurrency:

- they use asynchronous message-based communication through Ports using well-defined Protocols;
- they use run-to-completion semantics for message processing;
- they encapsulate passive objects (ensuring that thread interference cannot occur).



The process view shows the process organization of the system.

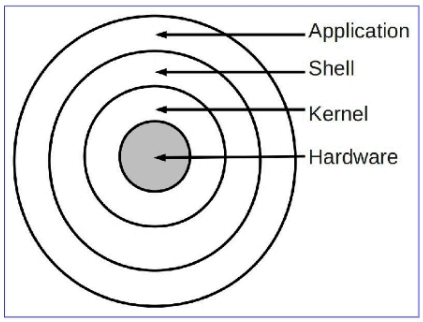
There are four additional views, the **Use-Case View** (handled in the Requirements discipline), and the **Logical View**, **Deployment View**, and **Implementation View**; these views are handled in the Analysis & Design and Implementation disciplines.

The architectural views are documented in a **Software Architecture Document**. You may add different views, such as a security view, to convey other specific aspects of the software architecture.

So in essence, architectural views can be seen as abstractions or simplifications of the models built, in which you make important characteristics more visible by leaving the details aside. The architecture is an important means for increasing the quality of any model built during system development.

General Architecture of a Operating System

Learning objective: Explain the general architecture of a operating system



Click on image to enlarge.

An operating system is a program that acts as an interface between a user of a computer and the computer resources. The purpose of an operating system is to provide an environment in which a user may execute programs.

Hardware

The hardware consists of the memory, CPU, arithmetic-logic unit, various bulk storage devices, I/O, peripheral devices and other physical devices.

Kernel

In computing, the kernel is the central component of most computer operating systems; it is a bridge between applications and the actual data processing done at the hardware level. The kernel's responsibilities include managing the system's resources (the communication between hardware and software components). Usually as a basic component of an operating system, a kernel can provide the lowest-level

and other physical devices.

Kernel

In computing, the kernel is the central component of most computer operating systems; it is a bridge between applications and the actual data processing done at the hardware level. The kernel's responsibilities include managing the system's resources (the communication between hardware and software components). Usually as a basic component of an operating system, a kernel can provide the lowest-level abstraction layer for the resources (especially processors and I/O devices) that application software must control to perform its function. It typically makes these facilities available to application processes through inter-process communication mechanisms and system calls.

Shell

A shell is a piece of software that provides an interface for users to an operating system which provides access to the services of a kernel. The name shell originates from shells being an outer layer of interface between the user and the innards of the operating system (the kernel). [Wikipedia]

Operating system shells generally fall into one of two categories: command-line and graphical. Command-line shells provide a command-line interface (CLI) to the operating system, while graphical shells provide a graphical user interface (GUI). In either category the primary purpose of the

interrupt signal.

Types of interrupts

Interrupts are classified into two types:

I. Hardware interrupt

A hardware interrupt is an electronic signal from an external hardware device that indicates it needs attention from the OS. One example of this is moving a mouse or pressing a keyboard key. In these examples of interrupts, the processor must stop to read the mouse position or keystroke at that instant.

In this type of interrupt, all devices are connected to the Interrupt Request Line (IRL). Typically, a hardware IRQ has a value that associates it with a particular device. This makes it possible for the processor to determine which device is requesting service by raising the IRQ, and then provide service accordingly.

There are three types of hardware interrupts:

Maskable interrupts

In a processor, an internal interrupt mask register selectively enables and disables hardware requests. When the mask bit is set, the interrupt is enabled

There are three types of hardware interrupts:

Maskable interrupts

In a processor, an internal interrupt mask register selectively enables and disables hardware requests. When the mask bit is set, the interrupt is enabled. When it is clear, the interrupt is disabled. Signals that are affected by the mask are maskable interrupts.

Non-maskable interrupts

In some cases, the interrupt mask cannot be disabled so it does not affect some interrupt signals. These are non-maskable interrupts and are usually high-priority events that cannot be ignored.

Spurious interrupts

Also known as a *phantom interrupt* or *ghost interrupt*, a spurious interrupt is a type of hardware interrupt for which no source can be found. These interrupts are difficult to identify if a system misbehaves. If the ISR does not account for the possibility of such interrupts, it may result in a system deadlock.

II. Software interrupts

II. Software interrupts

A software interrupt occurs when an application program terminates or requests certain services from the OS. Usually, the processor requests a software interrupt when certain conditions are met by executing a special instruction. This instruction invokes the interrupt and functions like a subroutine call. Software interrupts are commonly used when the system interacts with device drivers or when a program requests OS services.

In some cases, software interrupts may be triggered unexpectedly by program execution errors rather than by design. These interrupts are known as *exceptions* or *traps*.

Interrupts vs. polling

Polling is a state of continuous monitoring where a microcontroller in a computing system continuously checks the status of all devices. The first device encountered with the IRQ bit is serviced first and the appropriate ISR is called to service that device. The system is easy to implement and ensures that the component that needs service gets it.

However, because the microcontroller

In some cases, software interrupts may be triggered unexpectedly by program execution errors rather than by design. These interrupts are known as *exceptions* or *traps*.

Interrupts vs. polling

Polling is a state of continuous monitoring where a microcontroller in a computing system continuously checks the status of all devices. The first device encountered with the IRQ bit is serviced first and the appropriate ISR is called to service that device. The system is easy to implement and ensures that the component that needs service gets it.

However, because the microcontroller uses all its processing time for polling, it cannot carry out simultaneous operation. Also, a lot of time is wasted by interrogating the IRQ bit of all devices. Interrupts provide a solution to this problem. With interrupts, the controller does not need to regularly monitor the status of devices. Instead, it responds only when an interrupt occurs. So, when there is an interrupt, the controller is notified that it needs service.

What is an ISR?

More Detail

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

Program execution

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the comple

- Resource Allocation
- Protection

Program execution

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use). 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.

I/O Operation

An I/O subsystem comprises of I/O devices and their corresponding driv
software. Drivers hide the peculiarities of

handling.

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.

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. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.

media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.

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

- 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

Communication

In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.

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

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

Considering a computer system havi

- CPU scheduling algorithms are used for better utilization of CPU.

Protection

Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities.

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.

Previous Page Next Page

Virtual Memory is a storage allocation scheme in which secondary memory can be addressed as though it were part of the main memory. The addresses a program may use to reference memory are distinguished from the addresses the memory system uses to identify physical storage sites, and program-generated addresses are translated automatically to the corresponding machine addresses.

The size of virtual storage is limited by the addressing scheme of the computer system and the amount of secondary memory is available not by the actual number of the main storage locations.

It is a technique that is implemented using both hardware and software. It maps memory addresses used by a program, called virtual addresses, into physical addresses in computer memory.

1. All memory references within a process are logical addresses that are dynamically translated into physical addresses at run time. This means that a process can be swapped in and out of the main memory such that it occupies different places in the main memory at different times during the course of execution.
2. A process may be broken into a number of pieces and these pieces need not be continuously located in the main memory during execution. The combination of dynamic run-

the course of execution.

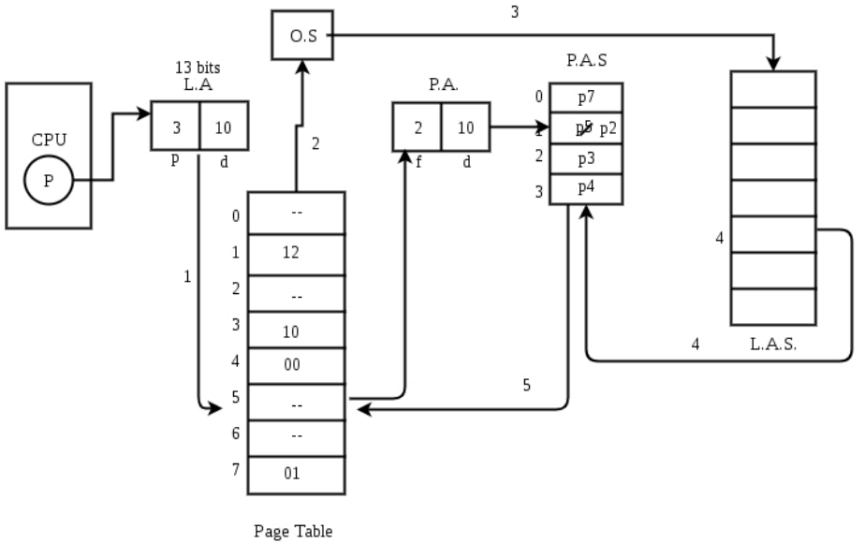
2. A process may be broken into a number of pieces and these pieces need not be continuously located in the main memory during execution. The combination of dynamic run-time address translation and use of page or segment table permits this.

If these characteristics are present then, it is not necessary that all the pages or segments are present in the main memory during execution. This means that the required pages need to be loaded into memory whenever required. Virtual memory is implemented using Demand Paging or Demand Segmentation.

Demand Paging :

The process of loading the page into memory on demand (whenever page fault occurs) is known as demand paging.

The process includes the following steps :



1. If the CPU tries to refer to a page that is currently not available in the

1. If the CPU tries to refer to a page that is currently not available in the main memory, it generates an interrupt indicating a memory access fault.
2. The OS puts the interrupted process in a blocking state. For the execution to proceed the OS must bring the required page into the memory.
3. The OS will search for the required page in the logical address space.
4. The required page will be brought from logical address space to physical address space. The page replacement algorithms are used for the decision-making of replacing the page in physical address space.
5. The page table will be updated accordingly.

6. The signal will be sent to the CPU to continue the program execution and it will place the process back into the ready state.

Hence whenever a page fault occurs these steps are followed by the operating system and the required page is brought into memory.

Advantages :

- More processes may be maintained in the main memory: Because we are going to load only some of the pages of any particular process, there is room for more processes. This leads to more efficient utilization of the processor because it is more likely that at least one of the more numerous processes will be in the ready state at any particular time.
- A process may be larger than all of the main memory: One of the most

particular time.

- A process may be larger than all of the main memory: One of the most fundamental restrictions in programming is lifted. A process larger than the main memory can be executed because of demand paging. The OS itself loads pages of a process in the main memory as required.
- It allows greater multiprogramming levels by using less of the available (primary) memory for each process.

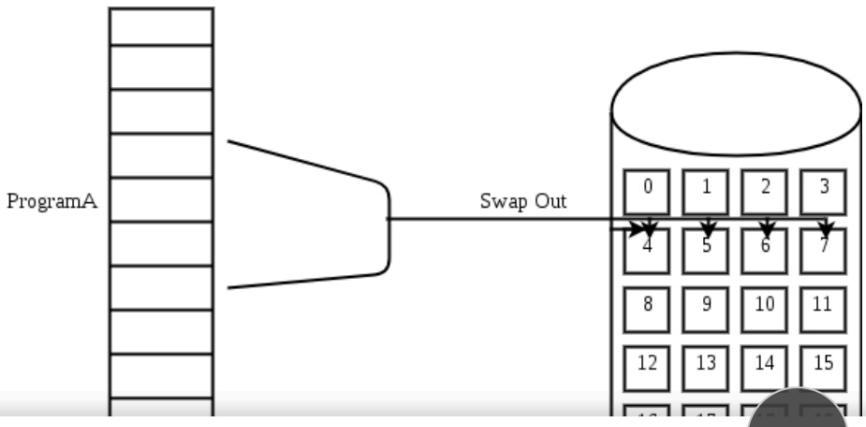
Page Fault Service Time :

The time taken to service the page fault is called page fault service time. The page fault service time includes the time taken to perform all the above six steps.

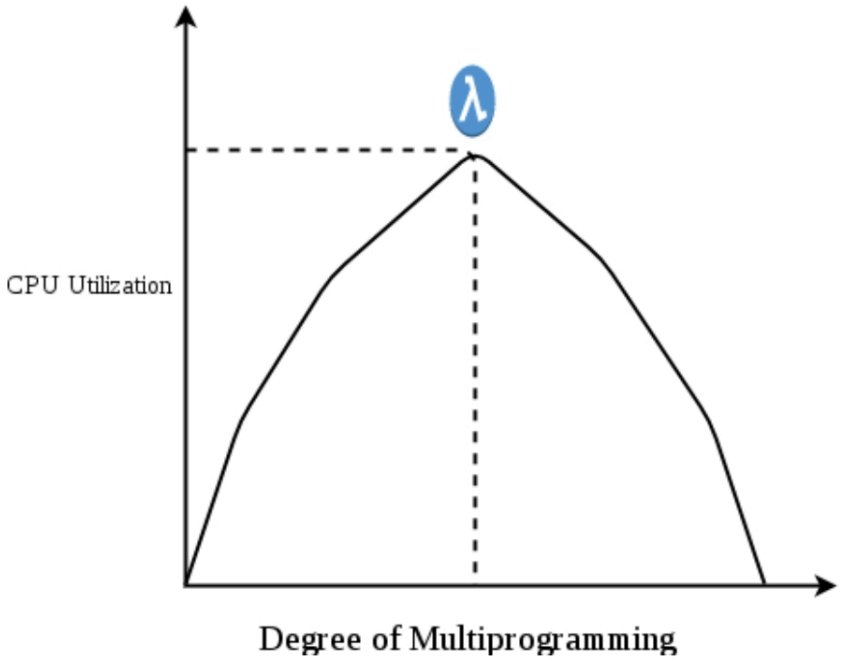
```
Let Main memory access time is: m
Page fault service time is: s
Page fault rate is : p
```

Swapping:

Swapping a process out means removing all of its pages from memory, or marking them so that they will be removed by the normal page replacement process. Suspending a process ensures that it is not runnable while it is swapped out. At some later time, the system swaps back the process from the secondary storage to the main memory. When a process is busy swapping pages in and out then this situation is called thrashing.



Thrashing :



At any given time, only a few pages of any process are in the main memory and therefore more processes can be maintained in memory. Furthermore, time is saved because unused pages

At any given time, only a few pages of any process are in the main memory and therefore more processes can be maintained in memory. Furthermore, time is saved because unused pages are not swapped in and out of memory. However, the OS must be clever about how it manages this scheme. In the steady-state practically, all of the main memory will be occupied with process pages, so that the processor and OS have direct access to as many processes as possible. Thus when the OS brings one page in, it must throw another out. If it throws out a page just before it is used, then it will just have to get that page again almost immediately. Too much of this leads to a condition called Thrashing. The system spends most of its time swapping pages

Too much of this leads to a condition called Thrashing. The system spends most of its time swapping pages rather than executing instructions. So a good page replacement algorithm is required.

In the given diagram, the initial degree of multiprogramming up to some extent of point(lambda), the CPU utilization is very high and the system resources are utilized 100%. But if we further increase the degree of multiprogramming the CPU utilization will drastically fall down and the system will spend more time only on the page replacement and the time is taken to complete the execution of the process will increase. This situation in the system is called thrashing.

Causes of Thrashing :

1. **High degree of multiprogramming** : If the number of processes keeps on increasing in the memory then the number of frames allocated to each process will be decreased. So, fewer frames will be available for each process. Due to this, a page fault will occur more frequently and more CPU time will be wasted in just swapping in and out of pages and the utilization will keep on decreasing.
   For example:
   Let free frames = 400
   Case 1: Number of process = 100
   Then, each process will get 4 frames.

   Case 2: Number of processes = 400
   Each process will get 1 frame.
   Case 2 is a condition of thrashing

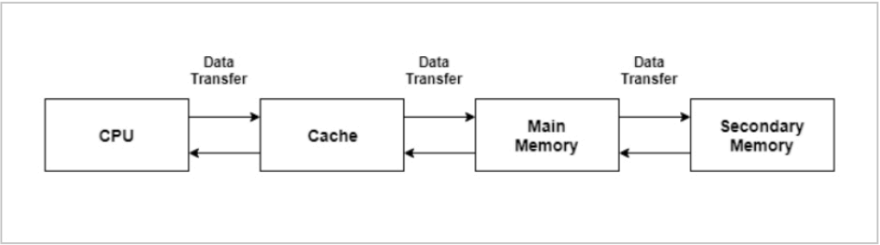
2. **Lacks of Frames**: If a process has fewer frames then fewer pages of that process will be able to reside in memory and hence more frequent swapping in and out will be required. This may lead to thrashing. Hence sufficient amount of frames must be allocated to each process in order to prevent thrashing.

Recovery of Thrashing :

- Do not allow the system to go into thrashing by instructing the long-term scheduler not to bring the processes into memory after the threshold.
- If the system is already thrashing then instruct the mid-term scheduler to suspend some of the processes so that we can recover the system from thrashing.

Cache is a type of memory that is used to increase the speed of data access. Normally, the data required for any process resides in the main memory. However, it is transferred to the cache memory temporarily if it is used frequently enough.

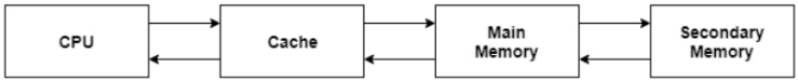
A diagram to better understand the data transfer in cache management is as follows –



Cache Performance

The cache performance can be explained using the following steps –

- If a process needs some data, it first searches in the cache memory. If the data is available in the cache, this is termed as a cache hit and the data is accessed as required.
- If the data is not in the cache then it is termed as a cache miss. Then the data



Cache Performance

The cache performance can be explained using the following steps −

- If a process needs some data, it first searches in the cache memory. If the data is available in the cache, this is termed as a cache hit and the data is accessed as required.
- If the data is not in the cache then it is termed as a cache miss. Then the data is obtained from the main memory. After that the data is transferred to the cache memory under the assumption that it will be needed again.
- The performance of the cache is measured using the hit ratio. It is the number of cache hits divided by the total cache accesses. The formula for this is:

$$\text{Hit Ratio} = \frac{\text{Number of Cache Hits}}{\text{Number of Cache Hits} + \text{Number of Cache Misses}}$$

Types of Cache Memory

There are mainly two types of cache memory i.e. primary cache and secondary cache. These are explained in

Types of Cache Memory

There are mainly two types of cache memory i.e. primary cache and secondary cache. These are explained in detail as follows −

Primary Cache

Primary cache is very fast and its access time is similar to the processor registers. This is because it is built onto the processor chip. However because of this reason, its size is quite small. It is also known as a level 1 cache and is build using static RAM (SRAM).

Secondary Cache

The secondary cache or external cache is cache memory that is external to the primary cache. It is located between the primary cache and the main memory. It is also known as a level 2 cache and is often housed on the processor chip as well.

Advantages of Cache Memory

Some of the advantages of cache memory are as follows −

Advantages of Cache Memory

Some of the advantages of cache memory are as follows −

- Cache memory is faster than main memory as it is located on the processor chip itself. Its speed is comparable to the processor registers and so frequently required data is stored in the cache memory.
- The memory access time is considerably less for cache memory as it is quite fast. This leads to faster execution of any process.
- The cache memory can store data temporarily as long as it is frequently required. After the use of any data has ended, it can be removed from the cache and replaced by new data from the main memory.

Disadvantages of Cache Memory

Some of the disadvantages of cache memory are as follows −

- Since the cache memory is quite fast, it is extremely useful in any computer system. However, it is also quite expensive and so is used judiciously.
- The cache is memory expensive as observed from the previous point. Also

Que 1.2. **Describe the classification of operating system.**

OR

Write down the different types of operating system.

AKTU 2016-17, Marks 10

Answer

Various types of operating system are as follows :

1. **Batch system :**
   a. This type of operating system was used in the earlier age.
   b. To speedup processing, jobs with similar needs were batched together and were run through the computer as a group.
   c. The definitive feature of a batch system is the lack of interaction between the user and the job while that job is executing.
   d. In this execution environment, the CPU is often idle.
2. **Multi-programming system :**
   a. In this type of operating system, more than one program will reside into main memory.
   b. The operating system picks and begins to execute one of the jobs in the memory.

   c. Eventually, the job may have to wait for some task, the operating system simply switches to another job and executes it.
   d. When the first job finishes, waiting job gets the CPU back.
   e. As long as there is always some job to execute, the CPU will never be idle.
3. **Time-sharing system :**
   a. A time-shared operating system allows the many users to share the computer simultaneously.
   b. A time-shared operating system uses CPU scheduling and multi-programming to provide each user with a small portion of a time-shared computer.
4. **Real time operating system :**
   a. Real time operating system is a special purpose operating system, used when there are rigid time requirements on the operation of a processor or the flow of data.

Que 1.3. **What is real time operating system ? What is the difference between hard real time and soft real time operating system ?**

AKTU 2013-14, Marks 05

Answer

1. Real time operating system is a special purpose operating system, used when there are rigid time requirements on the operation of a processer or the flow of data.
2. Real time operating system has well-defined, fixed time constraints otherwise system will fail.
3. For example, scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, home-appliances controllers, air traffic control system etc.

Difference between hard real time and soft real time operating system :

S. No.	Characteristic	Hard real time	Soft real time
1.	Response time	Hard required	Soft desired
2.	Peak-load performance	Predictable	Degraded
3.	Control of pace	Environment	Computer
4.	Safety	Often critical	Non-critical
5.	Size of data files	Small/medium	Large
6.	Redundancy type	Active	Checkpoint-recovery
7.	Data integrity	Short-term	Long-term
8.	Error detection	Autonomous	User assisted

Que 1.4. **Discuss essential properties of time-sharing operating system, real time operating system and distributed operating system.**

AKTU 2012-13, Marks 05

Answer

Properties of time-sharing operating system :

1. Uses CPU scheduling and multi-programming to provide economical interactive use of a system.
2. The CPU switches rapidly from one user to another *i.e.*, the CPU is shared between a number of interactive users.
3. Instead of having a job defined by spooled card images, each program reads its next control instructions from the terminal and output is normally printed immediately on the screen.

Properties of real time operating system :

1. Real time systems are usually dedicated, embedded systems.
2. They typically read from and react to sensor data.
3. The system must guarantee response to events within fixed periods of time to ensure correct performance.

Properties of distributed operating system :

1. Distributes computation among several physical processors.
2. The processors do not share memory or a clock, instead, each processor has its own local memory.
3. They communicate with each other through various communication lines.

Que 1.5. **Explain the following terms :**

i. Multi-programming system
ii. Multi-processor system

Answer

i. Multi-programming system : Refer Q. 1.2, Page 1–3B, Unit-1.

ii. Multi-processor system :

1. Multi-processor systems have more than one processor in close communication. They share the computer bus, system and input-output devices and sometimes memory.
2. In multi-processing system, it is possible for two processes to run in parallel.
3. Multi-processor systems are of two types : symmetric multi-processing and asymmetric multi-processing.

ii. Multi-threading :

1. Multi-threading extends the idea of multi-tasking into applications, so we can sub-divide specific operations within a single application into individual threads.
2. Each of the threads can run in parallel.
3. The OS divides processing time not only among different applications, but also among each thread within an application.
4. In a multi-threaded program, an example application might be divided into four threads : a user interface thread, a data acquisition thread, network communication, and a logging thread.
5. We can prioritize each of these, so that they operate independently.
6. Thus, in multi-threaded applications, multiple tasks can progress in parallel with other applications that are running on the system.

Que 1.7. Write down the difference between multi-processing and multi-programming operating system.

AKTU 2013-14, Marks 05

Answer

S. No.	Multi-processing	Multi-programming
1.	Multi-processing refers to processing of multiple processes at same time by multiple CPUs.	Multi-programming keeps several programs in main memory at the same time and execute them concurrently utilizing single CPU.
2.	It utilizes multiple CPUs.	It utilizes single CPU.
3.	It permits parallel processing.	Context switching takes place.

4.	Less time taken to process the jobs.	More time taken to process the jobs.
5.	It facilitates much efficient utilization of devices of the computer system.	Less efficient than multi-processing.
6.	Usually more expensive.	Such systems are less expensive.

Que 1.8. **Describe the differences between symmetric and asymmetric multi-processing.** **AKTU 2014-15, Marks 05**

Answer

S.No.	Basis	Symmetric multi-processing	Asymmetric multi-processing
1.	Architecture	All processor in symmetric multi-processing has the same architecture.	All processor in asymmetric multi-processing may have same or different architecture.
2.	Communication	All processors communicate with another processor by a shared memory.	Processors need not to communicate as they are controlled by the master processor.
3.	Failure	If a processor fails, the computing capacity of the system reduces.	If a master processor fails, a slave is turned to the master processor to continue the execution. If a slave processor fails, its task is switched to other processors.
4.	Ease	Symmetric multi-processor is complex.	Asymmetric multi-processor is simple.

Que 1.9. **What is spooling ?**

OR

What is spooling ? What are the advantage of spooling over buffering ?

2.	Only one program is in execution at a time, any time-critical device management is not required, which simplifies the I/O management.	Multi-programming operating system allow sharing of I/O devices among multiple users, more sophisticated I/O management is required.
3.	Since files are also accessed in a serial manner, a concurrency control mechanism for file access is not required, making file management also a very simple matter in a batch operating system.	File management in multi-programming operating system must provide advanced protection, and concurrency control methods.
4.	The users are not allowed to interact with the computer system.	The programs should be scheduled in such a way that the CPU remains busy for maximum amount of time.
5.	**Example :** Execution of batch files such as autoexec.bat by the computer.	**Example :** Interleaved execution of two or more different and independent programs by the same computer.

PART-2

Operating System Structure, Layered Structure, System Components, Operating System, Services Re-entrant Kernels, Monolithic and Micro-kernel System.

CONCEPT OUTLINE : PART-2

1. The spooling operation uses a disk as a very large buffer.
2. Spooling is capable of overlapping I/O operation for one job with processor operations for another job.

Que 1.10. **Differentiate between batch processing system and multi-programming system with example.**

Answer

S. No.	Batch processing system	Multi-programming system
1.	In batch systems, the instructions, data and some control information are submitted to the computer operator in the form of a job.	Multi-programming system allow concurrent execution of multiple programs, and hence multi-programming operating system require more sophisticated scheduling algorithms.

More results

What is called segmentation?

What are the two major differences between segmentation and paging?

Difference between Paging and Segmentation

Paging	Segmentation
A page is of the fixed block size.	A segment is of variable size.
It may lead to internal fragmentation.	It may lead to external fragmentation.
In Paging, the hardware decides the page size.	The segment size is specified by the user.

7 more rows • 28-Sept-2022

https://www.guru99.com › paging-v...

Difference between Paging and Segmentation - Guru99