DATE: 21-10-2023

FINAL REPORT

OF MS WORD

OPERATING SYSTEM

RIDA ABDUL HAFEEZ

BATCH: DIT (07 TO 09)

Contents

OPERATING SYSTEM:

A. DUAL MODE OPERATION IN OS:

1) DUAL MODE OPERATION:

Dual mode operation in OS (operating system) is the ability of the system to switch between two different modes of operation for computation processes and to avoid system crashes. The Dual Mode Operations in the OS guard it against illegal users. There are two types of operations: the user and kernel modes.[9]
An error in one program can adversely affect many processes; it might modify data of another program or also can affect the operating system.[1]
For example: if a process stuck in the infinite loop then this infinite loop could affect the correct operation of other processes. So to ensure the proper execution of the operating system, there are two modes of operation:[1]

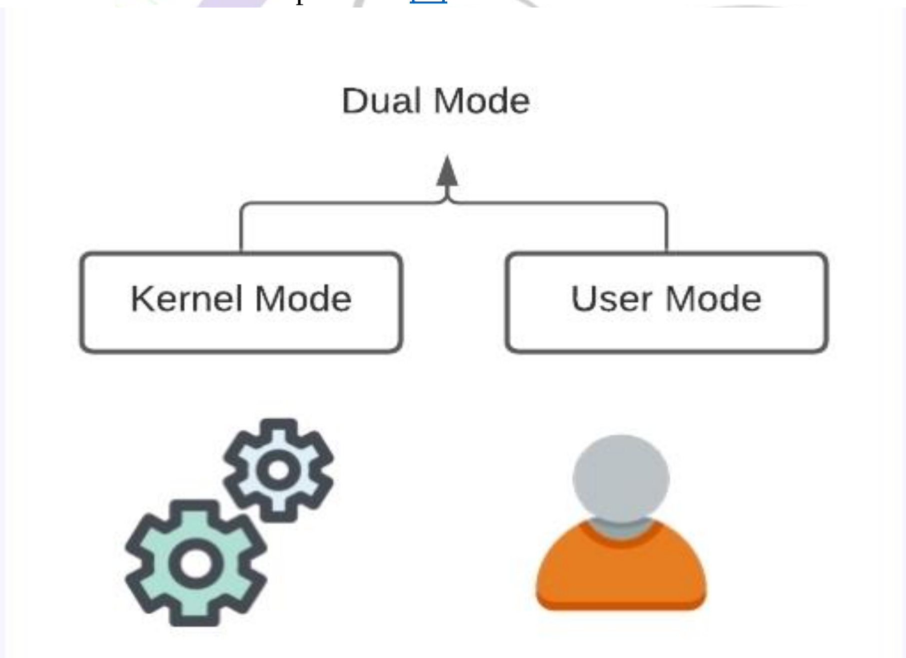


Fig: 1. MODES OF OPERATIONS[9]

a. User mode:

When the computer system is run by user applications like creating a text document or using any application program, then the system is in user mode. When the user application requests for a service from the operating system or an interrupt occurs or system call, then there will be a transition from user to kernel mode to fulfill the requests.[1]

b. Kernel Mode:

When the system boots, the hardware starts in kernel mode and when the operating system is loaded, it starts user application in user mode. To provide protection to the hardware, we have privileged instructions which execute only in kernel mode.
If the user attempts to run privileged instruction in user mode then it will treat instruction as illegal and traps to OS.
Some of the privileged instructions are:
1. Handling Interrupts
2. To switch from user mode to kernel mode.
3. Input-Output management.[1]

2) Need for Dual Mode Operations:

- Certain types of tasks do not require any type of hardware support, that's why certain types of processes are to be made hidden from the user. These tasks can be deal separately by using the Dual Mode of the operating system.
- The Kernel Level programs perform all the bottom level functions of the operating systems like memory management, process management etc., for this purpose the operating system needs to function in the Dual Mode.
- Dual Mode is necessary for specifying the access to the users only to the tasks of their use in an operating system.
- Basically, whenever the operating system works on the user applications, it held in the user mode.
- When the user requests for some hardware services, a transition from User Mode to the Kernel Mode occurs which is done by changing the mode bit from 1 to 0. Also the mode bit again changed to 1 for returning back in the User Mode.[1]

3) ADVANTAGE:

i. **Protection:** Dual-mode operation provides a layer of protection between user programs and the operating system. In user mode, programs are restricted from accessing privileged resources, such as hardware devices or sensitive system data. In kernel mode, the operating system has full access to these resources, allowing it to protect the system from malicious or unauthorized access.

ii. **Stability:** Dual-mode operation helps to ensure system stability by preventing user programs from interfering with system-level operations. By restricting access to privileged resources in user mode, the operating system can prevent programs from accidentally or maliciously causing system crashes or other errors.

iii. **Flexibility:** Dual-mode operation allows the operating system to support a wide range of applications and hardware devices. By providing a well-defined interface between user programs and the operating system, it is easier to develop and deploy new applications and hardware.

iv. **Debugging:** Dual-mode operation makes it easier to debug and diagnose problems with the operating system and applications. By switching between user mode and kernel mode, developers can identify and fix issues more quickly and easily.

v. **Security:** Dual-mode operation enhances system security by preventing unauthorized access to critical system resources. User programs running in user mode cannot modify system data or perform privileged operations, reducing the risk of malware attacks or other security threats.

vi. **Efficiency:** Dual-mode operation can improve system performance by reducing overhead associated with system-level operations. By allowing user programs to access resources directly in user mode, the operating system can avoid unnecessary context switches and other performance penalties.

vii. **Compatibility:** Dual-mode operation ensures backward compatibility with legacy applications and hardware devices. By providing a standard interface for user programs to interact with the operating system, it is easier to maintain compatibility with older software and hardware.

viii. **Isolation:** Dual-mode operation provides isolation between user programs, preventing one program from interfering with another. By running each program in its own protected memory space, the operating system can prevent programs from accessing each other's data or causing conflicts.

ix. **Reliability:** Dual-mode operation enhances system reliability by preventing crashes and other errors caused by user programs. By restricting access to critical system resources, the operating system can ensure that system-level operations are performed correctly and reliably.[1]

4) Disadvantages:

i. **Performance:** Dual-mode operation can introduce overhead and reduce system performance. Switching between user mode and kernel mode requires a context switch, which can be time-consuming and can impact system performance.

ii. **Complexity:** Dual-mode operation can increase system complexity and make it more difficult to develop and maintain operating systems. The need to support both user mode and kernel mode can make it more challenging to design and implement system features and to ensure system stability.

iii. **Security:** Dual-mode operation can introduce security vulnerabilities. Malicious programs may be able to exploit vulnerabilities in the operating system to gain access to privileged resources or to execute malicious code.

iv. **Reliability:** Dual-mode operation can introduce reliability issues as it is difficult to test and verify the correct operation of both user mode and kernel mode. Bugs or errors in either mode can lead to system crashes, data corruption, or other reliability issues.

v. **Compatibility:** Dual-mode operation can create compatibility issues as different operating systems may implement different interfaces or policies for user mode and kernel mode. This can make it difficult to develop applications that are compatible with multiple operating systems or to migrate applications between different systems.

vi. **Development complexity:** Dual-mode operation requires a higher level of technical expertise and development skills to design and implement the operating system. This can increase the development time and cost for creating new operating systems or updating existing ones.

vii. **Maintenance complexity**: Dual-mode operation can make maintenance and support more complex due to the need to ensure compatibility and security across both user mode and kernel mode. This can increase the cost and time required for system updates, patches, and upgrades.[1]

5) PURPOSE OF DUAL-MODE OPERATION:

Dual-mode operation is designed to provide a layer of protection and stability to computer systems by separating user programs and the operating system into two modes: user mode and kernel mode.
User mode restricts access to privileged resources, while kernel mode has full access to these resources.[1]

6) REASONS OF SWITCHING:

There are two main reasons behind the switching between User mode and kernel mode, such as:

1. If everything were to run in a single-mode, we would end up with Microsoft's issue in the earlier versions of Windows. If a process were able to exploit a vulnerability, that process then could control the system.

2. Certain conditions are known as a trap, an exception or a system fault typically caused by an exceptional condition such as division by zero, invalid memory access, etc. If the process is running in kernel mode, such a trap situation can crash the entire operating system. A process in user mode that encounters a trap situation only crashes the user-mode process[2]

B. THREADS

1) Thread in Operating System:

Thread is a sequential flow of tasks within a process. Threads in an operating system can be of the same or different types. Threads are used to increase the performance of the applications.Each thread has its own program counter, stack, and set of registers. However, the threads of a single process might share the same code and data/file. Threads are also termed lightweight processes as they share common resources.
E.g.: While playing a movie on a device the audio and video are controlled by different threads in the background.[3]

2) Need OF Thread:

Thread is a sequential flow of tasks within a process. Threads in an operating system can be of the same or different types. Threads are used to increase the performance of the applications.Each thread has its own program counter, stack, and set of registers. However, the threads of a single process might share the same code and data/file. Threads are also termed lightweight processes as they share common resources.
E.g.: While playing a movie on a device the audio and video are controlled by different threads in the background.[3]

3) MULTITHREADING:

In Multithreading, the idea is to divide a single process into multiple threads instead of creating a whole new process. Multithreading is done to achieve parallelism and to improve the performance of the applications as it is faster in many ways which were discussed above. The other advantages of multithreading are mentioned below.

a) ADVANTAGES:

i. **Resource Sharing:**

Threads of a single process share the same resources such as code, data/file.

ii. **Responsiveness:**

Program responsiveness enables a program to run even if part of the program is blocked or executing a lengthy operation. Thus, increasing the responsiveness to the user.

iii. **Economy:**

It is more economical to use threads as they share the resources of a single process. On the other hand, creating processes is expensive.[3]

4) Process vs. Thread:

Process simply means any program in execution while the thread is a segment of a process. The main differences between process and thread are mentioned below:[3]

Process	Thread
Processes use more resources and hence they are termed as heavyweight processes.	Threads share resources and hence they are termed as lightweight processes.
Creation and termination times of processes are slower.	Creation and termination times of threads are faster compared to processes.
Processes have their own code and data/file.	Threads share code and data/file within a process.
Communication between processes is slower.	Communication between threads is faster.
Context Switching in processes is slower.	Context switching in threads is faster.
Processes are independent of each other.	Threads, on the other hand, are interdependent. (i.e. they can read, write or change another thread's data)
E.g.: Opening two different browsers.	E.g.: Opening two tabs in the same browser.

5) Types of Threads:

a) User-Level Thread:

User-level threads are implemented and managed by the user and the kernel is not aware of it. User-level threads are implemented using user-level libraries and the OS does not recognize these threads. User-level thread is faster to create and manage compared to kernel-level thread. Context switching in user-level threads is faster. If one user-level thread performs a blocking operation then the entire process gets blocked.

E.g.: POSIX threads, Java threads, etc.

i. Pros:

- User threads are easier to implement than kernel threads.
- Threads at the user level can be used in operating systems that do not allow threads at the kernel level.
- It is more effective and efficient.
- Context switching takes less time than kernel threads.
- It does not necessitate any changes to the operating system.
- The representation of user-level threads is relatively straightforward. The user-level process's address space contains the register, stack, PC, and mini thread control blocks.

- Threads may be easily created, switched, and synchronized without the need for process interaction. [3]

ii. **Cons:**

- Threads at the user level are not coordinated with the kernel.
- The entire operation is halted if a thread creates a page fault. [3]

b) Kernel-Level Thread:

Kernel level threads are implemented and managed by the OS. Kernel level threads are implemented using system calls and Kernel level threads are recognized by the OS. Kernel-level threads are slower to create and manage compared to user-level threads. Context switching in a kernel-level thread is slower. Even if one kernel-level thread performs a blocking operation, it does not affect other threads.

E.g.: Window Solaris. [3]

i. **Pros:**

- All threads are completely aware of the kernel-level thread.
- The scheduler may decide to devote extra CPU time to threads with large numerical values.
- Applications that happen to block the frequency should use the kernel-level thread. [3]

ii. **Cons:**

- All threads are managed and scheduled by the kernel thread.
- Kernel threads are more complex to build than user threads.
- Kernel-level threads are slower than user-level threads. [3]

6) Components of Threads:

1. Stack space
2. Register set
3. Program counter. [3]

7) **Benefits of Threads:**

- Threads improve the overall performance of a program.
- Threads increases the responsiveness of the program.
- Context switching time in threads is faster.
- Threads share the same memory and resources within a process.
- Communication is faster in threads.
- Threads provide concurrency within a process.
- Enhanced throughout of the system.
- Since different threads can run parallel, threading enables the utilization of the multiprocessor architecture to a greater extent and increases efficiency. [3]

C. PERSONAL CONTROL BLOCK (PCB)

1) INTRODUCION:

While creating a process, the operating system performs several operations. To identify the processes, it assigns a process identification number (PID) to each process. As the operating system supports multi-programming, it needs to keep track of all the processes. For this task, the process control block (PCB) is used to track the process's execution status. Each block of memory contains information about the process state, program counter, stack pointer, status of opened files, scheduling algorithms, etc.
All this information is required and must be saved when the process is switched from one state to another. When the process makes a transition from one state to another, the operating system must update information in the process's PCB. A process control block (PCB) contains information about the process, i.e. registers, quantum, priority, etc. The process table is an array of PCBs that means logically contains a PCB for all of the current processes in the system.

a) **Pointer:** It is a stack pointer that is required to be saved when the process is switched from one state to another to retain the current position of the process.

b) **Process state:** It stores the respective state of the process.

c) **Process number**: Every process is assigned a unique id known as process ID or PID which stores the process identifier.

d) **Program counter:** It stores the counter, which contains the address of the next instruction that is to be executed for the process.

e) **Register:** These are the CPU registers which include the accumulator, base, registers, and general-purpose registers.

f) **Memory limits:** This field contains the information about memory management system used by the operating system. This may include page tables, segment tables, etc.

g) **Open files list:** This information includes the list of files opened for a process. [4]

Pointer
Process State
Process Number
Program Counter
Registers
Memory Limits
List of open Files
.....

Process Control Block

FIG: 1. PROCESS CONTROL BLOCK (PCB) [6]

2) WHY IT IS USED?

The primary purpose of a PCB is to enable the OS to manage multiple processes efficiently by keeping track of the state of each process and allocating system resources accordingly. When a process is created, the OS creates a PCB for that process and stores all the necessary information about the process in it. The OS then uses the information in the PCB to manage the process and ensure that it runs efficiently.

PCBs are used by the OS for many tasks, including process scheduling, process synchronization, and process communication. The information stored in a PCB is used by the OS to make decisions about which process to run next, which resources to allocate, and when to switch between processes.

In summary, the PCB is a crucial component of modern operating systems that enables the efficient management of processes. Process Control Block in OS, It provides the OS with all the necessary information about a process to ensure that it runs efficiently and effectively. [5]

3) Advantages:

a) Efficient process management:

The process table and PCB provide an efficient way to manage processes in an operating system. The process table contains all the information about each process, while the PCB contains the current state of the process, such as the program counter and CPU registers.

b) Resource management:

The process table and PCB allow the operating system to manage system resources, such as memory and CPU time, efficiently. By keeping track of each process's resource usage, the operating system can ensure that all processes have access to the resources they need.

c) Process synchronization:

The process table and PCB can be used to synchronize processes in an operating system. The PCB contains information about each process's synchronization state, such as its waiting status and the resources it is waiting for.

d) Process scheduling:

The process table and PCB can be used to schedule processes for execution. By keeping track of each process's state and resource usage, the operating system can determine which processes should be executed next. [5]

4) Disadvantages:

a) Overhead:

The process table and PCB can introduce overhead and reduce system performance. The operating system must maintain the process table and PCB for each process, which can consume system resources.

b) Complexity:

The process table and PCB can increase system complexity and make it more challenging to develop and maintain operating systems. The need to manage and synchronize multiple processes can make it more difficult to design and implement system features and ensure system stability.

c) Scalability:

The process table and PCB may not scale well for large-scale systems with many processes. As the number of processes increases, the process table and PCB can become larger and more difficult to manage efficiently.

d) Security:

The process table and PCB can introduce security risks if they are not implemented correctly. Malicious programs can potentially access or modify the process table and PCB to gain unauthorized access to system resources or cause system instability. [5]

1

This field includes information about the amount of CPU used, time constraints, jobs or process number, etc. The process control block stores the register content also known as execution content of the processor when it was blocked from running. This execution content architecture enables the operating system to restore a process's execution context when the process returns to the running state. When the process makes a transition from one state to another, the operating system updates its information in the process's PCB. The operating system maintains pointers to each process's PCB in a process table so that it can access the PCB quickly. [4]

D. Clustered Operating System:

1) INTRODUCTION:

Cluster systems are similar to parallel systems because both systems use multiple CPUs. The primary difference is that clustered systems are made up of two or more independent systems linked together. They have independent computer systems and a shared storage media, and all systems work together to complete all tasks. All cluster nodes use two different approaches to interact with one another, like message passing interface (MPI) and parallel virtual machine (PVM).

Cluster operating systems are a combination of software and hardware clusters. Hardware clusters aid in the sharing of high-performance disks among all computer systems, while software clusters give a better environment for all systems to operate. A cluster system consists of various nodes, each of which contains its cluster software. The cluster software is installed on each node in the clustered system, and it monitors the cluster system and ensures that it is operating properly. If one of the clustered system's nodes fails, the other nodes take over its storage and resources and try to restart. [7]

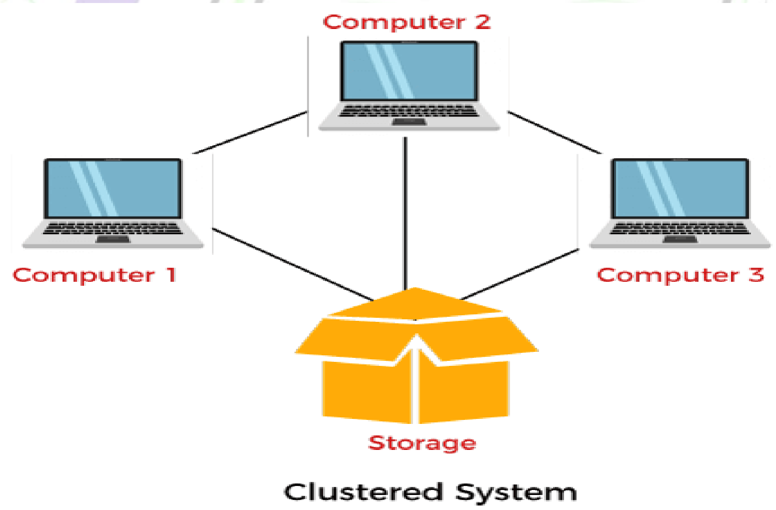


FIG: 1. CLUSTERED SYSTEM [7]

2) TYPES OF CLUSTERS:

There are two clusters available to make a more efficient cluster. These are as follows:

a) Software Cluster

b) Hardware Cluster

a) SOFTWARE CLUSTER:

The Software Clusters allows all the systems to work together.

b) HARDWARE CLUSTER:

It helps to allow high-performance disk sharing among systems.

3) TYPES OF CLUSTERED SYSTEM:

i. Asymmetric Clustering System
ii. Symmetric Clustering System
iii. Parallel Cluster System

4) ADVANTAGES:

Various advantages of Clustered Operating System are as follows:

i. High Availability:

Although every node in a cluster is a standalone computer, the failure of a single node doesn't mean a loss of service. A single node could be pulled down for maintenance while the remaining clusters take on a load of that single node.

ii. Cost Efficiency:

When compared to highly reliable and larger storage mainframe computers, these types of cluster computing systems are thought to be more cost-effective and cheaper. Furthermore, most of these systems outperform mainframe computer systems in terms of performance.

iii. Additional Scalability:

A cluster is set up in such a way that more systems could be added to it in minor increments. Clusters may add systems in a horizontal fashion. It means that additional systems could be added to clusters to improve their performance, fault tolerance, and redundancy.

iv. Fault Tolerance:

Clustered systems are quite fault-tolerance, and the loss of a single node does not result in the system's failure. They might also have one or more nodes in hot standby mode, which allows them to replace failed nodes.

v. Performance:

The clusters are commonly used to improve the availability and performance over the single computer systems, whereas usually being much more cost-effective than the single computer system of comparable speed or availability.

vi. Processing Speed:

The processing speed is also similar to mainframe systems and other types of supercomputers on the market.

5) Disadvantages:

Various disadvantages of the Clustered Operating System are as follows:

i. Cost-Effective:

One major disadvantage of this design is that it is not cost-effective. The cost is high, and the cluster will be more expensive than a non-clustered server management design since it requires good hardware and a design.

ii. Required Resources:

Clustering necessitates the use of additional servers and hardware, making monitoring and maintenance difficult. As a result, infrastructure must be improved.

iii. Maintenance:

It isn't easy to system establishment, monitor, and maintenance this system. [7]

E. MEMORY MANAGEMENT IN OS:

Memory management is the process of controlling and coordinating a computer's main memory. It ensures that blocks of memory space are properly managed and allocated so the operating system (OS), applications and other running processes have the memory they need to carry out their operations.

As part of this activity, memory management takes into account the capacity limitations of the memory device itself, de-allocating memory space when it is no longer needed or extending that space through virtual memory. Memory management strives to optimize memory usage so the CPU can efficiently access the instructions and data it needs to execute the various. [8]

1. PAGING IN OPERATING SYSTEM:

Paging is a memory management scheme that eliminates the need for a contiguous allocation of physical memory. The process of retrieving processes in the form of pages from the secondary storage into the main memory is known as paging. The basic purpose of paging is to separate each procedure into pages. Additionally, frames will be used to split the main memory. This scheme permits the physical address space of a process to be non – contiguous.

[10]

a. PROCESS OF PAGING IN OS:

In Operating Systems, Paging is a storage mechanism used to retrieve processes from the secondary storage into the main memory in the form of pages. The main idea behind the paging is to divide each process in the form of pages. The main memory will also be divided in the form of frames

One page of the process is to be stored in one of the frames of the memory. The pages can be stored at the different locations of the memory but the priority is always to find the contiguous frames or holes. Pages of the process are brought into the main memory only when they are required otherwise they reside in the secondary storage. [11]

2. MEMORY ALLOCATION:

To gain proper memory utilization, memory allocation must be allocated efficient manner. One of the simplest methods for allocating memory is to divide memory into several fixed-sized partitions and each partition contains exactly one process. Thus, the degree of multiprogramming is obtained by the number of partitions.

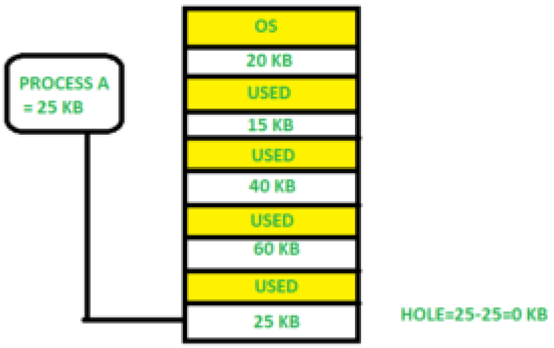
a. **Multiple partition allocation:** In this method, a process is selected from the input queue and loaded into the free partition. When the process terminates, the partition becomes available for other processes.
b. **Fixed partition allocation:** In this method, the operating system maintains a table that indicates which parts of memory are available and which are occupied by processes. Initially, all memory is available for user processes and is considered one large block of available memory. This available memory is known as a "Hole". When the process arrives and needs memory, we search for a hole that is large enough to store this process. If the requirement is fulfilled then we allocate memory to process, otherwise keeping the rest available to satisfy future requests. While allocating a memory sometimes dynamic storage allocation problems occur, which concerns how to satisfy a request of size n from a list of free holes. There are some solutions to this problem:

i. ***First Fit:***
In the First Fit, the first available free hole fulfils the requirement of the process allocated.

First Fit

Here, in this diagram, a 40 KB memory block is the first available free hole that can store process A (size of 25 KB), because the first two blocks did not have sufficient memory space.

ii. ***Best Fit:***
In the Best Fit, allocate the smallest hole that is big enough to process requirements. For this, we search the entire list, unless the list is ordered by size.

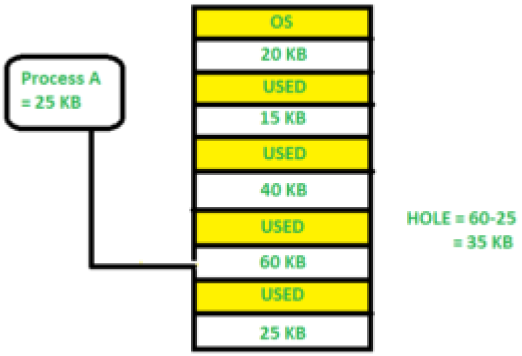


Best Fit

Here in this example, first, we traverse the complete list and find the last hole 25KB is the best suitable hole for Process A(size 25KB). In this method, memory utilization is maximum as compared to other memory allocation techniques.

iii. ***Worst Fit :***

In the Worst Fit, allocate the largest available hole to process. This method produces the largest leftover hole.



Worst Fit

Here in this example, Process a (Size 25 KB) is allocated to the largest available memory block which is 60KB. Inefficient memory utilization is a major issue in the worst fit. [12]

3. MEMORY DE-ALLOCATION:

When a process no longer needs a particular memory block, the operating system de-allocates that memory and marks it as free. This allows the freed memory to be reused for other processes. De-allocating memory involves updating relevant data structures, such as the free memory list. [13]

4. PAGE REPLACEMENT:

Page replacement is needed in the operating systems that use virtual memory using Demand Paging. As we know in Demand paging, only a set of pages of a process is loaded into the memory. This is done so that we can have more processes in the memory at the same time.

When a page that is residing in virtual memory is requested by a process for its execution, the Operating System needs to decide which page will be replaced by this requested page. This process is known as page replacement and is a vital component in virtual memory management. [14]

REFERENCES:

1) https://www.geeksforgeeks.org/dual-mode-operations-os/
2) https://www.javatpoint.com/dual-mode-operations-in-operating-system
3) https://www.scaler.com/topics/operating-system/threads-in-operating-system/
4) https://www.geeksforgeeks.org/process-table-and-process-control-block-pcb/
5) https://datatrained.com/post/process-control-block-in-os/#:~:text=OS%20(PCB)%3F-,A%20Process%20Control%20Block%20in%20OS%20(PCB)%20is%20a%20data,resources%20used%20by%20the%20process.
6) https://quescol.com/operating-system/process-control-block
7) https://www.javatpoint.com/clustered-operating-system
8) https://www.techtarget.com/whatis/definition/memory-managements processes
9) https://www.codingninjas.com/studio/library/dual-mode-in-os
10) https://www.geeksforgeeks.org/paging-in-operating-system/
11) https://www.javatpoint.com/os-paging-with-example#:~:text=In%20Operating%20Systems%2C%20Paging%20is,in%20the%20form%20of%20frames.
12) https://www.geeksforgeeks.org/memory-management-in-operating-system/
13) Chat GPT
14) https://www.scaler.com/topics/operating-system/page-replacement-algorithm/