Orchid International College

(Tribhuvan University)

Bijayachowk, Gaushala



CASE STUDY ON WINDOWS

Faculty of Humanities and Social Sciences

Tribhuvan University

*Kritipur, Nepal*

**Submitted By:**

**Name**: Darsheel Chudal

**Roll no: 8**

BCA 4th Sem

**Submitted To:**

**Name**: Alok Chaudhary

Orchid International College

BCA Department

**External Examiner**

**…………………………………..**

Contents

[**Introduction:** 1](#_heading=h.gjdgxs)

[**History:** 1](#_heading=h.30j0zll)

[**Features:** 2](#_heading=h.1fob9te)

**Advantage and Disadvantage** ……………………………………………………………………………………………………….3

[**Components of Windows:**](#_heading=h.3znysh7)………………………………………………………………………………………………………………5

[**Windows System Architecture:**](#_heading=h.2et92p0)…………………………………………………………………………………………………..7

[**Types of Modules:**](#_heading=h.tyjcwt)……………………………………………………………………………………………………………………8

[**Process Management:** 10](#_heading=h.1t3h5sf)

[**Creation and Termination:** 10](#_heading=h.4d34og8)

[**Process Creation:** 10](#_heading=h.2s8eyo1)

[**Process Termination:** 10](#_heading=h.17dp8vu)

[**Scheduler:** 12](#_heading=h.1ksv4uv)

[**Priority Levels:** 12](#_heading=h.1ksv4uv)

[**Scheduling Algorithms:** 12](#_heading=h.1ksv4uv)

[**Schedulers in Multi-Core Systems:** 13](#_heading=h.1ksv4uv)

**Schedulers in Real-Time Systems** [13](#_heading=h.1ksv4uv)

[**Inter-Process Communication:** 14](#_heading=h.44sinio)

[**Memory Management:** 15](#_heading=h.2jxsxqh)

[**Virtual Memory Management:** 16](#_heading=h.z337ya)

[**File system Management Approach:** 17](#_heading=h.3j2qqm3)

[**Conclusion:** 18](#_heading=h.1y810tw)

**Windows Case Study**

**Introduction:**

The journey of operating systems has been a fascinating ride, shaping the way we interact with technology. Among the pioneers in this realm stand MS-DOS and Windows, two keystones developed by Microsoft. While MS-DOS laid the groundwork with its command-line interface, Windows elevated the user experience through its graphical interface and innovative features. Let’s delve into the story of MS-DOS and Windows, exploring their significance in computing and their lasting impact on modern operating systems.

**History:**

MS-DOS (1981):

Back in 1981, Microsoft introduced MS-DOS (Microsoft Disk Operating System) as a foundational step into the operating system domain. It catered to early personal computers, offering basic functionalities like file management via command-line interactions.

Windows 1.0 (1985):

In 1985, Windows 1.0 emerged as Microsoft’s attempt to marry a graphical interface with MS-DOS. Although it featured rudimentary applications, its adoption was limited due to performance issues and restricted functionality.

Windows 2.0 (1987):

Building upon its predecessor, Windows 2.0, released in 1987, brought improvements like overlapping windows and enhanced graphics. However, it still heavily relied on MS-DOS for its core operations.

Windows 3.0 (1990):

The release of Windows 3.0 in 1990 marked a significant leap forward. With better graphics and expanded application support, it garnered widespread acclaim, establishing Windows as a formidable contender in the market.

Windows 95 (1995):

A watershed moment arrived in 1995 with the launch of Windows 95. Introducing the iconic Start menu and taskbar, it transformed the user interface landscape while enhancing multitasking and networking capabilities.

Windows Vista, 7, 8, and 10 (2007-2015):

In the subsequent years, Microsoft rolled out a series of Windows versions, each bringing its own set of enhancements. From the security-focused Vista to the unified experience of Windows 10, the journey showcased continual innovation.

**Features:**

Command-line Interface: MS-DOS operates through a text-based interface where users input commands to perform tasks like navigating directories, managing files, and running programs.

Basic File Management: It provides fundamental file management functionalities such as creating, copying, moving, and deleting files and directories.

Graphical User Interface (GUI): Windows presents a graphical interface with icons, windows, and menus, making it more user-friendly and visually appealing compared to MS-DOS's command-line interface.

Multitasking Support: Windows allows users to run multiple programs simultaneously, enabling multitasking and enhancing productivity.

**Advantages and Disadvantages:**

**MS-DOS (Microsoft Disk Operating System):**

Advantages:

Simplicity: MS-DOS boasts simplicity, making it approachable for users seeking a straightforward computing experience.

Performance: Its lightweight nature translates to efficient performance, particularly on older hardware.

Compatibility: Many legacy applications are designed to run on MS-DOS, ensuring compatibility with older software.

Disadvantages:

Limited Functionality: Lack of modern features like multitasking and a graphical interface can hinder productivity.

Command-Line Interface: For those accustomed to graphical interfaces, the command-line setup might pose a learning curve.

Hardware Support: Compatibility issues with newer hardware may restrict its usage on contemporary devices.

**Windows:**

Advantages:

User-Friendly Interface: Windows offers an intuitive graphical interface, appealing to a wide spectrum of users.

Multitasking: Its multitasking prowess allows users to juggle multiple tasks seamlessly, enhancing productivity.

Hardware and Software Support: Windows boasts extensive compatibility, supporting a diverse range of hardware and software.

Disadvantages:

Resource Intensive: The robust features of Windows come at the cost of resource consumption, demanding higher hardware specifications.

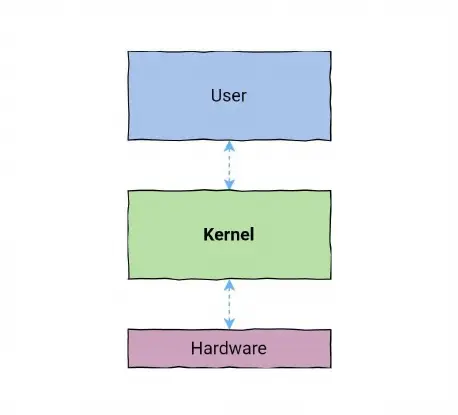
Security Concerns: Its popularity makes Windows a prime target for malware, necessitating vigilant security measures.

Cost: Acquiring Windows licenses, especially for professional versions, can be a significant investment, posing a barrier for some users.

# 

# **Components of Windows:**

1. **Kernel**

****

**Function:**

* Manages system resources such as CPU, memory, and input/output operations.
* Provides essential services required for the operation of the operating system.

**Key Features:**

* Task scheduling, memory management, and interrupt handling.
* Supports multi-tasking, allowing multiple processes to run simultaneously.

**User Mode**

**Function:**

* Houses user applications and ensures system stability by isolating user processes from the kernel.

**Key Features:**

* Provides a safe environment for executing user applications without affecting system stability.
* Handles user interactions through the Graphical User Interface (GUI).

1. **Hardware Abstraction Layer (HAL)**

**Function:**

* Acts as a bridge between the operating system and hardware components, abstracting hardware-specific details.

**Key Features:**

* Enables the operating system to communicate with different hardware configurations without needing to know the specifics of each device.
* Facilitates portability by allowing the operating system to run on various hardware platforms with minimal changes.

**Device Drivers**

**Function:**

* Facilitate communication between the operating system and hardware components by providing a standardized interface.

**Key Features:**

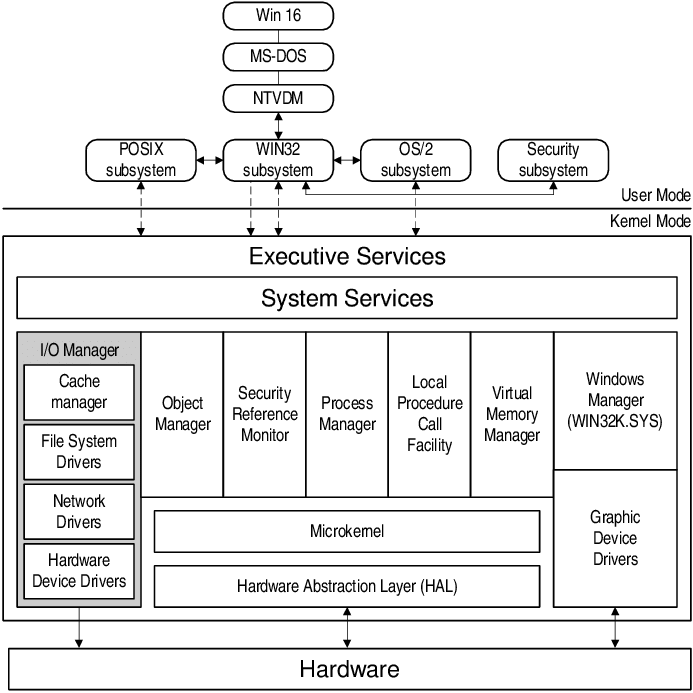
* Enable plug-and-play functionality, allowing the operating system to automatically detect and configure hardware devices.
* Implement specific protocols and commands required for controlling and managing hardware components.

These components work together to provide a seamless and efficient computing experience on Windows-based systems. The kernel manages the low-level system operations, the user mode ensures application stability, the HAL provides hardware abstraction for compatibility, and device drivers enable communication with hardware devices.

Top of Form

## **Windows System Architecture:**

Windows utilizes a hybrid architecture, merging elements of both monolithic and microkernel designs. This structure optimizes performance and flexibility, catering to diverse computing needs.



.

# **Module Types:**

# In the realm of software and operating systems, modules represent discrete units or components that fulfill specific functions within a broader system framework. These modules aid in code organization, enhance maintainability, and foster collaboration among developers. Here are several typical types of modules encountered in this context:

# **Kernel Modules:**

# Responsible for direct interaction with hardware and the management of system resources.

# Examples encompass device drivers and file system handlers.

# **User Mode Modules:**

# Operate in user mode, addressing specific user-facing tasks or applications.

# Examples include user interface components, application modules, and system utilities.

# **Library Modules:**

# Assemblages of precompiled functions and procedures reusable across multiple programs.

# Instances encompass standard libraries like C standard library, .NET Framework libraries, and Java libraries.

# **Driver Modules:**

# Facilitate communication between the operating system and hardware devices.

# Examples consist of printer drivers, graphics drivers, and network drivers.

# **Service Modules:**

# Operate in the background, offering specific services to applications or other modules.

# Examples cover networking services, database services, and security services.

# **Configuration Modules:**

# Manage system configuration settings and parameters.

# Examples include configuration files, registry settings, and initialization scripts.

# **Utility Modules:**

# Provide supplementary tools and utilities for system maintenance and troubleshooting.

# Examples involve disk management tools, backup utilities, and diagnostic tools.

# **Process Management:**

# Process management in Windows is pivotal for optimizing system resource utilization, ensuring smooth task execution, and upholding system stability and reliability. It plays a pivotal role in handling multiple tasks concurrently and adapting to diverse computing requirements. The following activities are integral to process management:

# **Creation and Termination:**

# In the Windows operating system, processes serve as fundamental units of execution, representing running applications and system services. The creation and termination of processes are fundamental aspects managed by the operating system to ensure efficient resource utilization and system stability.

# **Process Creation:**

# Windows initiates process creation through system calls, such as CreateProcess or CreateThread. Upon launching an application, the operating system allocates necessary resources, including memory, CPU time, and system handles, to instantiate the process.

# During the creation process, the following steps typically occur:

# Resource Allocation: The operating system allocates memory space, assigns a unique process identifier (PID), and reserves system resources required for the new process.

# Initialization: The operating system initializes the process by setting up its environment, loading necessary libraries, and preparing it for execution.

# Execution: Once initialized, the process enters the execution state, where it begins executing its instructions and interacting with the system and other processes.

# **Process Termination:**

# Process termination involves concluding the execution of a process and releasing associated resources back to the system. Proper termination is crucial to maintain system stability and ensure efficient resource management.

# Windows provides mechanisms for both voluntary and involuntary process termination:

# **Voluntary Termination:**

# Exit Process: Applications can gracefully terminate themselves by invoking the ExitProcess function, enabling them to release resources and perform cleanup operations.

# User Termination: Users can close applications through graphical interfaces, triggering the termination of associated processes.

# **Involuntary Termination:**

# Task Manager: In cases where an application becomes unresponsive or encounters errors, users can terminate the process using the Task Manager, forcibly ending its execution.

# System Policies: Administrators can enforce system policies to terminate processes that violate predefined rules or consume excessive resources.

# Upon termination, the operating system performs the following actions:

# Resource Deallocation: The system releases memory, handles, and other resources allocated to the terminated process, making them available for other processes.

# Cleanup: Any temporary files, locks, or other system resources associated with the process are cleaned up to maintain system integrity.

# Notification: In some cases, the operating system notifies parent processes or system services about the termination event, enabling them to handle the situation accordingly.

# By effectively managing the creation and termination of processes, Windows ensures system reliability, responsiveness, and optimal performance.

# **Scheduler:**

The scheduler is a vital component of the Windows operating system, responsible for efficiently managing and allocating system resources, primarily CPU time, among competing processes. Its primary role is to determine which processes should run, in what order, and for how long, ensuring optimal system performance and responsiveness.

**Priority Levels:**

Windows employs a priority-based scheduling mechanism to prioritize processes and allocate CPU time accordingly. Each process is assigned a priority level, ranging from real-time (highest priority) to low (lowest priority), which dictates its access to CPU resources.

* **Real-Time Priority:** Processes with real-time priority have the highest precedence and are allocated CPU time without delay, making them suitable for critical system tasks.
* **High, Above Normal, Normal, Below Normal, and Low Priorities:** These priority levels cater to user applications, system services, and background tasks, respectively, ensuring fair CPU allocation and responsiveness.

**Scheduling Algorithms:**

Windows utilizes various scheduling algorithms to optimize CPU utilization and system responsiveness:

1. **Preemptive Scheduling:** The scheduler can preempt lower-priority processes to allocate CPU time to higher-priority processes, ensuring critical tasks are executed promptly.
2. **Round-Robin Scheduling:** This algorithm allocates CPU time in a circular manner, giving each process a fixed time slice to execute its instructions, promoting fairness among competing processes.
3. **Priority-Based Scheduling:** Processes with higher priority levels are given precedence over lower-priority processes, ensuring critical tasks receive timely execution.
4. **Multilevel Queue Scheduling:** This approach categorizes processes into different queues based on their priority levels and assigns CPU time to each queue, balancing performance and responsiveness across the system.

**Schedulers in Multi-Core Systems:**

In multi-core and multi-processor systems, Windows employs a **Multiprocessor Scheduling** algorithm to distribute processes across available CPU cores, maximizing parallelism and system throughput. This ensures efficient utilization of hardware resources and enhances system performance.

**Schedulers in Real-Time Systems:**

For real-time systems requiring deterministic response times, Windows offers a **Real-Time Scheduler** that guarantees timely execution of critical tasks by prioritizing them over non-time-critical processes, ensuring system reliability and stability.

# **Inter-Process Communication:**

# Window provides a comprehensive set of IPC mechanisms to facilitate efficient communication and data exchange between processes.

# **Named Pipes:**

# Enable bidirectional communication between processes on the same computer, offering a reliable and efficient data transfer method.

# **Mailslots:**

# Support message-based communication, allowing processes to send and receive messages asynchronously.

# **Sockets:**

# Facilitate network communication using TCP/IP protocols, enabling processes to communicate over local networks or the internet.

# These IPC mechanisms play a vital role in enabling collaboration, synchronization, and integration among applications, ensuring seamless interoperability and enhanced functionality.

# **Memory Management:**

Windows employs a sophisticated memory management system designed to optimize physical memory utilization, facilitate multitasking, and support memory-intensive applications effectively. By leveraging various memory management techniques, Windows ensures efficient allocation, utilization, and management of system memory, contributing to enhanced system performance and stability.

* **Paging:**  
  Paging is a crucial memory management technique used by Windows to manage memory resources efficiently. It involves swapping data between Random Access Memory (RAM) and storage devices, such as hard drives or SSDs, to accommodate larger applications and processes that exceed the available physical memory capacity. By dynamically moving data between RAM and storage, paging enables the system to maintain responsiveness and performance, even when dealing with memory-intensive workloads.
* **Segmentation:**  
  Segmentation is another key memory management strategy employed by Windows to organize memory into segments, each designated for specific tasks or data types. By dividing memory into distinct segments, segmentation improves system performance by reducing memory access times, minimizing fragmentation, and optimizing data storage and retrieval. This organized approach to memory management ensures efficient use of memory resources, enhances data locality, and contributes to overall system responsiveness and stability.

Together, these memory management techniques enable Windows to achieve optimal memory allocation, minimize resource contention, and ensure seamless execution of applications and processes, regardless of their size or complexity. By prioritizing memory-intensive tasks and efficiently managing memory resources, Windows delivers a smooth and responsive computing experience, even under demanding workloads.

# **Virtual Memory Management:**

# The Memory Manager in Windows serves as a cornerstone for managing both virtual and physical memory, ensuring seamless process execution, efficient memory utilization, and optimal system performance. Virtual memory management is essential for supporting multitasking, accommodating large applications, and handling diverse computing workloads effectively.

# **Virtual Address Mapping:**

# Virtual Address Mapping is a fundamental aspect of virtual memory management in Windows. The Memory Manager translates virtual addresses generated by processes into corresponding physical addresses in memory. This translation process facilitates efficient memory access and data retrieval, allowing processes to operate as if they have access to a large, contiguous block of memory, even when physical memory is limited. By providing a layer of abstraction between virtual and physical memory, virtual address mapping enables flexible memory allocation and utilization, enhancing system responsiveness and performance.

# **Memory Optimization:**

# Memory Optimization encompasses various strategies employed by the Memory Manager to manage memory allocation, deallocation, and swapping effectively. These strategies aim to minimize memory fragmentation, maximize available memory, and optimize system performance under varying workloads. By dynamically adjusting memory allocations, swapping out less frequently used data to disk, and prioritizing memory-intensive tasks, the Memory Manager ensures efficient use of memory resources, reduces resource contention, and maintains system stability and responsiveness.

# In summary, the efficient management of virtual memory by the Memory Manager is crucial for supporting multitasking, accommodating large applications, and handling diverse computing workloads. By seamlessly integrating virtual and physical memory management techniques, Windows delivers a robust and responsive computing environment that can adapt to the demands of modern computing scenarios, ensuring a smooth and efficient user experience.

# **File system Management Approach:**

### File System Management Approach: Windows boasts support for a diverse array of file systems, each meticulously designed to cater to specific use cases, scenarios, and requirements. This versatile approach to file system management ensures data integrity, reliability, and accessibility across different storage media and devices, providing users with flexibility and convenience in managing their data.

### FAT32:

### FAT32 is a widely recognized file system known for its broad compatibility with various devices and platforms. It is particularly suitable for removable storage devices like USB flash drives and SD cards, as well as scenarios requiring smaller file sizes. While FAT32 lacks some of the advanced features found in other file systems, its universal compatibility makes it an excellent choice for cross-platform data sharing and accessibility.

### NTFS:

### NTFS (New Technology File System) is a robust and feature-rich file system offering advanced capabilities such as file permissions, encryption, and journaling. These features provide enhanced data protection, integrity, and security, making NTFS ideal for systems requiring stringent data management and security protocols. NTFS's ability to handle large file sizes and support for file compression further enhances its versatility and suitability for a wide range of applications and scenarios.

### exFAT:

### exFAT (Extended File Allocation Table) is a modern file system optimized for flash drives, large files, and compatibility across different operating systems and devices. Designed as an extension of the FAT32 file system, exFAT addresses the limitations of FAT32 by supporting larger file sizes, improved error handling, and increased storage capacity. Its cross-platform compatibility and performance make exFAT an excellent choice for modern storage solutions, including high-capacity flash drives and external hard drives.

### In conclusion, the support for multiple file systems in Windows underscores Microsoft's commitment to providing users with versatile and reliable storage solutions. Whether you're using removable storage, managing sensitive data, or requiring cross-platform compatibility, Windows offers a file system tailored to meet your specific needs, ensuring seamless data management and accessibility across diverse computing environments.

# 

# **Conclusion:**

Microsoft's journey from the foundational MS-DOS to the sophisticated Windows operating systems showcases a steadfast commitment to innovation, adaptability, and user-centric design. This evolutionary progression has profoundly influenced the digital landscape, establishing industry benchmarks and reshaping modern computing experiences.

* **MS-DOS:**

MS-DOS served as the cornerstone of personal computing, representing Microsoft's initial foray into the operating system domain. It introduced users to command-line interfaces, file management, and fundamental computing functionalities, laying the groundwork for the subsequent digital revolution.

* **Windows:**

Windows revolutionized computing with its pioneering graphical user interface (GUI), intuitive design, and advanced features. It fundamentally changed how users interacted with computers, making computing more accessible, user-friendly, and immersive. With each iteration, Windows has introduced innovative technologies, improved performance, and expanded capabilities, shaping the trajectory of computing and setting new industry standards.

Through relentless innovation, continual evolution, and a steadfast pursuit of excellence, Microsoft has cemented its position as a leader in the operating system market. By empowering users worldwide with cutting-edge technologies, intuitive interfaces, and dependable solutions, Microsoft remains at the forefront of driving progress, fostering creativity, and redefining the way we live, work, and connect in the digital era.