Bachelor of Science (Honours) in Data Science and Artificial Intelligence

DA 107 – Operating Systems

Introduction

Outline

- Introduction
- Terminology
- Components of OS
- Services offered by OS
- Elements of hardware interacted by OS
- Types of OS
- Examples of OS

Introduction

Introduction

- An operating system (OS) is an important part of almost every computer system.
- It is a complex software that acts as an intermediary between computer hardware and the user. It provides a platform for running applications and managing various hardware resources
- Most computer users have had some experience with an operating system, but it is difficult to pin down precisely what an operating system.
- It is an integrated set of programs that directs and manages the components and resources of a computer system, including main memory, the CPU and the peripheral devices.

Introduction

- The OS is somewhat like a housekeeper in that it tidies, organizes and maintains the functioning of various devices.
- The task of an operating system is to manage the hardware carefully, to achieve the best possible performance.
- This is accomplished by the operating system's controlling and coordinating such resources as the CPU, other processing units, both primary memory and secondary storage, and all input/output devices.

Introduction

- The hardware provides raw computing power, and the operating system makes this power conveniently accessible to the user. This unit presses what operating system do and basics of OS.

- An operating system is an organized collection of software that controls the overall operations of a computer.

- In other words, an operating system is a program that acts as an interface between a user of a computer and computer hardware.

- The primary goal of an OS is to make the computer system convenient to use.

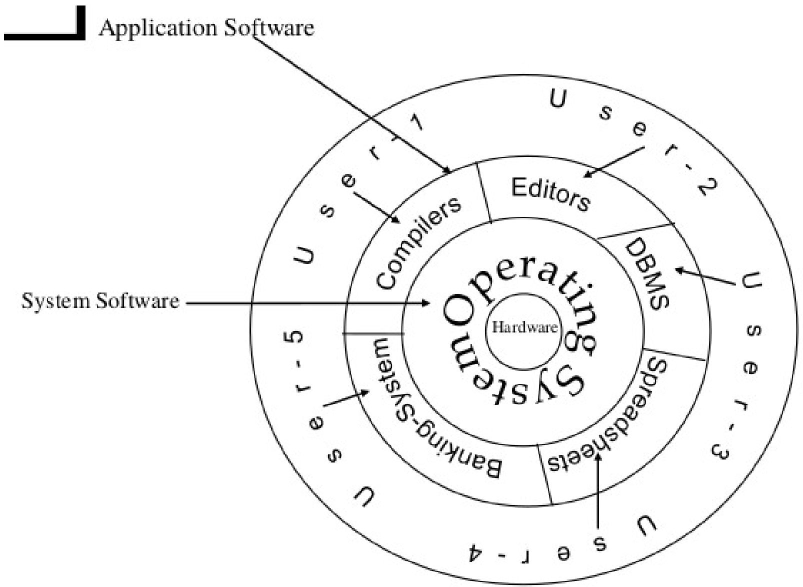
- A secondary goal is to use the computer hardware in an efficient manner.

Introduction – operating system - purpose

- It minimizes the computer user's intervention in (and concern) about machine's internal workings.
- It provides an environment in which a user may execute programs.
- It controls and co-ordinates the use of hardware among the various application programs.
- It acts as a managers of resources (hardware and software) and allocates them to specific programs.
- It controls the various I/O devices and user programs.
- It maximizes the overall efficiency and effectiveness of the system.
- It provides an environment within which other programs can do useful work.

Introduction – computer system

- A computer system consist of hardware (CPU, memory, I/O devices), system software (operating system), application programs (compilers, DBMS, Editor, spreadsheets) and the users



Terminology

Terminology

- It is important to acquaint with various terms used when discussing about operating system. These are

- Operating System: This refers to the software and files that are installed on a system so that it can boot and execute programs. It includes the kernel, administration tools, and system libraries

- Kernel: The kernel is the program that manages the system, including devices (hardware), memory, and CPU scheduling. It runs in a privileged CPU mode that allows direct access to hardware, called kernel mode.

- Process: An OS abstraction and environment for executing a program. The program normally runs in user mode, with access to kernel mode (e.g., for performing device I/O) via system calls or traps

Terminology

- **Threads**: An executable context that can be scheduled to run on a CPU. The kernel has multiple threads, and a process contains one or more.

- **Task**: A Linux runnable entity, which can refer to a process (with a single thread), a thread from a multithreaded process, or kernel threads

- **Kernel-space**: the memory address space for the kernel.

- **User-space**: the memory address space for processes

- **User-land**: user-level programs and libraries (/usr/bin, /usr/lib, . . .).

Terminology

- System call: A well-defined protocol for user programs to request the kernel to perform privileged operations, including device I/O.
- Processor: Not to be confused with process, a processor is a physical chip containing one or more CPUs.
- Interrupt: A signal sent by physical devices to the kernel, usually to request servicing of I/O.

Components of OS

Components

- **Kernel**: The core of the operating system that manages the hardware resources. Responsible for process management, memory management, device management, and system calls.

- **File system**: Manages the organization, storage, retrieval, naming, sharing, and protection of files on a disk. Provides a hierarchical directory structure.

- **Process Management**: Handles the creation, scheduling, and termination of processes. Manages process synchronization and communication.

- **Memory Management**: Allocates and deallocates memory space as needed by programs. Manages virtual memory and paging

Components

- **Device Drivers**: Interfaces with hardware devices, allowing the OS to communicate with peripherals such as printers, keyboards, and storage devices. Each device typically has its own driver.

- **User Interface (UI)**: Provides a way for users to interact with the computer. Can be a command-line interface (CLI) or a graphical user interface (GUI).

- **Security Access & Control**: Enforces access control policies to ensure that only authorized users and processes can access certain resources. Includes user authentication, encryption, and other security features.

- **Networking**: Manages network communication, allowing the computer to connect to and communicate with other devices on a network. Includes protocols, network stack, and network device drivers.

Components

- **System Calls**: Interfaces for applications to request services from the operating system. Examples include file operations, process creation, and communication.
- **Shell**: The command interpreter that allows users to interact with the operating system through a command-line interface. Executes commands and scripts.
- **Utilities**: System utilities perform various tasks such as disk formatting, backup, and system monitoring. Examples include text editors, file managers, and disk management tools
- **Scheduler**: Manages the scheduling of processes, determining which process gets access to the CPU and for how long
- **Interrupt Handlers**: Manages hardware and software interrupts, allowing the OS to respond to events such as hardware errors or I/O completion

Services offered by OS

Services offered by OS

- An operating system provides a range of services to both users and applications, facilitating the effective utilization of computer hardware and creating a user-friendly computing environment.

- Program Execution: The OS loads programs into memory and manages their execution, ensuring that the CPU executes instructions in a controlled manner.

- I/O Operations: Manages input and output operations, allowing programs to read from and write to devices such as disks, keyboards, and printers

- File System Manipulation: Provides services for creating, deleting, reading and writing files. Manages file attributes and directory structures.

Services offered by OS

- **Communication Services**: Facilitates communication between processes, either within the same system or over a network, through inter-process communication (IPC) mechanisms.
- **Error Detection and Handling**: Monitors the system for errors and responds appropriately, preventing system crashes and providing error messages to users and applications.
- **Resource Allocation**: Allocates system resources, such as CPU time, memory space, and I/O devices, among competing processes to optimize system performance.
- **Security and Protection**: Implements security mechanisms to control access to system resources and protect against unauthorized access. Enforces user authentication and authorization.

Services offered by OS

- **User Interface Services**: Provides a user interface that allows users to interact with the computer, including command-line interfaces (CLI) and graphical user interfaces (GUI).

- **Networking**: Manages network connections and communication, enabling processes to communicate over a network. Includes network protocols and services.

- **Interrupt Handling**: Handles hardware and software interrupts, allowing the OS to respond to events such as I/O completion, hardware errors, and timer events.

- **Memory Management**: Allocates and deallocates memory space for processes, manages virtual memory, and ensures efficient use of system memory.

Services offered by OS

- **Process Management**: Manages processes, including process creation, scheduling, termination, and communication between processes.

- **Device Management**: Controls and communicates with hardware devices, including device drivers, to facilitate I/O operations and ensure proper functioning of peripherals.

- **Backup and Recovery**: Provides mechanisms for backing up and restoring data, allowing users to recover from system failures or data loss.

- **System Logging and Monitoring**: Logs system events, errors, and activities. Monitors system performance and provides tools for system administrators to analyze and optimize system behavior.

Hardware Elements & OS

Hardware Elements & OS

- An operating system (OS) interacts with various hardware components to manage and control the computer system. The key elements of hardware with which an operating system interacts include the following.

- Central Processing Unit (CPU): The CPU is the primary processing unit of a computer. The OS interacts with the CPU to schedule tasks, allocate resources, and manage the execution of programs.

- Memory (RAM and Cache): The OS manages system memory, including RAM (Random Access Memory) and cache memory. It allocates memory space to running processes, facilitates data exchange between RAM and storage, and handles memory paging and swapping.

Hardware Elements & OS

- **Storage Devices**: Operating systems interact with various storage devices, such as hard drives, solid-state drives (SSDs), and external storage. They manage file systems, handle input/output operations, and ensure data storage and retrieval.

- **Input/Output Devices**: Interaction with input/output devices (e.g., keyboards, mice, displays, printers) is a crucial function of an operating system. It manages communication between software applications and these devices, facilitating user input and output.

- **Network Interface**: Operating systems interact with network interfaces to enable communication over computer networks. They manage network connections, protocols, and data transfer between connected devices.

Hardware Elements & OS

- **Motherboard and System Bus**: The OS interacts with the motherboard and the system bus to facilitate communication between various hardware components. It manages data transfer and control signals between the CPU, memory, and peripheral devices.

- **Interrupt Controllers**: Interrupt controllers are used to handle hardware interrupts. The OS interacts with interrupt controllers to respond to hardware events (e.g., keyboard input, disk operations) promptly.

- **Clock and Timer**: The operating system interacts with the system clock and timers to manage time-related functions, including scheduling tasks, handling timeouts, and maintaining system time.

Hardware Elements & OS

- **Power Management Components**: Operating systems often include power management features to interact with components like CPUs and hard drives to optimize power usage and control sleep/hibernation modes.

- **Graphics Processing Unit (GPU)**: For systems with a separate GPU, the operating system interacts with the graphics hardware to render graphics, manage display settings, and support graphical user interfaces.

- **BIOS/UEFI**: The Basic Input/Output System (BIOS) or Unified Extensible Firmware Interface (UEFI) is responsible for booting the computer and initializing hardware components. The OS interacts with BIOS/UEFI during the boot process.

Types of OS

Types of OS

- Operating systems can be classified into various types based on different criteria. Here are some common classifications
- **Single-User OS**: Designed to support a single user at a time. Examples include most personal computer operating systems like Windows, macOS, and Linux distributions for desktop use.
- **Multi-User OS**: Supports multiple users simultaneously. Examples include Unix-based systems, Linux servers, and mainframe operating systems.
- **Single-Tasking OS**: Allows only one task or process to run at a time. Older operating systems like MS-DOS are examples.
- **Multi-Tasking OS**: Supports the execution of multiple tasks or processes concurrently. Modern operating systems like Windows, macOS, and Linux are multi-tasking.

Types of OS

- **Single-Processor OS**: Designed to run on systems with a single central processing unit (CPU).
- **Multi-Processor OS**: Optimized for systems with multiple CPUs or cores, allowing parallel processing.
- **Batch Processing OS**: Processes a set of tasks in batches, without user interaction. Mainframes often use batch processing.
- **Real-Time OS**: Designed for systems that require immediate response to external events. Used in embedded systems, industrial control systems, and certain scientific applications.

Types of OS

- **Open-Source OS**: The source code is available for users to view, modify, and distribute. Examples include Linux distributions (Ubuntu, Fedora) and FreeBSD.

- **Proprietary OS**: Source code is not available, and the software is owned by a company. Examples include Microsoft Windows and macOS.

Types of OS

- **Distributed OS**: Manages and coordinates multiple interconnected computers, allowing them to work together as a single system. Examples include Google's Android, which is based on a Linux kernel.
- **Network OS**: Specialized for network operations, facilitating communication between computers. Examples include Novell NetWare.
- **Embedded OS**: Designed for embedded systems, such as those found in appliances, automobiles, and industrial machines. Examples include VxWorks and FreeRTOS.
- **Mobile OS**: Optimized for mobile devices such as smartphones and tablets. Examples include Android, iOS, and Windows Mobile.
- **Time-Sharing OS**: Supports multiple users simultaneously by dividing CPU time among them. Examples include Unix and its derivatives.

Examples of OS

Examples of OS

- **Personal Computer OS**:
  - Microsoft Windows: Versions include Windows 10, Windows 8.1, Windows 7.
  - macOS: Developed by Apple for Macintosh computers.
  - Linux Distributions: Examples include Ubuntu, Fedora, Debian, CentOS.

- **Server OS**:
  - Linux Server Distributions: Such as Ubuntu Server, CentOS, Red Hat Enterprise Linux (RHEL).
  - Microsoft Windows Server: Versions include Windows Server 2019, Windows Server 2016.

- **Mobile Operating Systems**:
  - Android: Developed by Google, used on a variety of mobile devices.
  - iOS: Developed by Apple, exclusively for iPhones, iPads, and iPod Touch.
  - Windows Mobile: An older mobile OS developed by Microsoft

Examples of OS

- **Embedded OS**:
  - VxWorks: Commonly used in embedded systems, real-time applications.
  - FreeRTOS: An open-source real-time operating system for embedded systems.
  - Embedded Linux: Customized Linux distributions tailored for embedded devices.

- **Real-Time OS (RTOS)**:
  - QNX: Used in embedded systems, automotive applications, and more.
  - RTOS in Industrial Automation: Examples include VxWorks, FreeRTOS, and RTLinux.

- **Network Operating Systems**:
  - Novell NetWare: Historically used for networking services.
  - Windows Server with Active Directory: Provides network services in Windows environments.

Examples of OS

- **Mainframe Operating Systems**:
  - **IBM z/OS**: Operating system for IBM mainframes.
  - **Unisys OS 2200**: Operating system for Unisys mainframes.

- **Distributed Operating Systems**:
  - **Google's Android**: Based on a modified Linux kernel and used in a wide range of devices.
  - **Distributed versions of Unix**: Examples include AIX (IBM), Solaris (Oracle), and HP-UX (Hewlett Packard).

- **Time-Sharing Operating Systems**:
  - **Unix**: A multi-user, multitasking operating system. Variants include Linux, BSD, and others.
  - **Multics**: An influential but historic time-sharing system.

- **Clustered Operating Systems**:
  - **Microsoft Windows Server Failover Clustering**: Supports high availability.
  - **Linux-HA (High Availability)**: Linux clustering solutions.

Desktop OS Vs Server OS

Desktop OS Vs Server OS

- Personal operating systems (OS) and server operating systems serve distinct purposes and are designed to meet different requirements. Key differences between a personal OS and a server OS are:

- User Interface:
    - Personal OS: Typically includes a graphical user interface (GUI) designed for individual users. Examples include Windows, macOS, and various Linux desktop environments.

    - Server OS: Often operates without a graphical user interface, especially in server deployments. Server OS is configured and managed remotely using command-line interfaces or web-based management tools.

Desktop OS Vs Server OS - User Interaction

- Designed for direct user interaction, providing a desktop environment, file management, and application execution for individual users.

- Primarily focuses on background services and handling requests from networked clients. Minimal direct user interaction occurs on the server itself.

Desktop OS Vs Server OS - Hardware resource allocation

- Prioritizes resources for individual user tasks and applications, often favoring responsiveness and interactive performance.

- Prioritizes resource allocation for network services, background processes, and multiple simultaneous connections. Emphasizes stability, reliability, and efficient use of hardware resources.

Desktop OS Vs Server OS - Services and applications

- Focuses on user-centric applications, entertainment, and productivity tools. Includes software for tasks like word processing, web browsing, and multimedia consumption.

- Designed to run server-based applications and services, such as web servers, database servers, email servers, and domain controllers.

Desktop OS Vs Server OS - Concurrency & Multi-User Support

- Designed for single-user or limited multi-user environments (e.g., a family computer). Supports concurrent execution of applications for a small number of users.

- Built to handle multiple simultaneous connections and user requests. Supports a large number of concurrent users accessing services such as web hosting, file sharing, and database management.

Desktop OS Vs Server OS – Security & access control

- Security features primarily focus on protecting the individual user's data and privacy.

- Emphasizes security measures to protect networked resources, control access to sensitive data, and defend against unauthorized access. Often includes features like user authentication, access controls, and encryption.

Desktop OS Vs Server OS - Resource Scaling

- Geared toward running on a single device or a small number of devices, such as a personal computer or laptop

- Engineered to scale across multiple servers or virtual machines to handle increasing workloads and provide high availability.

Desktop OS Vs Server OS – Redundancy & reliability

- **Reliability is important but may not require the same level of redundancy and fault tolerance as server environments.**

- **Prioritizes redundancy and fault tolerance to ensure continuous operation. Often includes features like clustering, load balancing, and backup systems.**

Mainframe OS

Mainframe OS

- A mainframe operating system (OS) is designed specifically for large, high-performance computing environments known as mainframes.

- Mainframes are powerful computers that are capable of handling a vast amount of data, supporting a large number of simultaneous users, and running critical business applications.

- The key characteristics of a mainframe OS are:

- **Scale and Performance**: Designed to run on large-scale, high-performance mainframe computers. Mainframes are capable of processing massive amounts of data and supporting thousands of concurrent users.

- **Concurrency and Multi-User Support**: Optimized for concurrent execution of numerous tasks and supports a large number of simultaneous users. It can efficiently handle multiple and diverse workloads.

Mainframe OS

- **Resource Management**: Implements sophisticated resource management to efficiently allocate and utilize system resources such as CPU, memory, and I/O for diverse workloads.

- **Reliability and Availability**: Emphasizes high reliability and availability, often providing features like fault tolerance, redundancy, and hot-swappable components.

- **Data Processing and Transaction Processing**: Well-suited for batch processing, transaction processing, and handling large databases. Commonly used in critical business applications such as banking, finance, and logistics.

- **Cost and Licensing**: Typically involves high upfront costs but can be cost-effective for large organizations with substantial computing needs.

Thank You!