Zakho Technical Institute / IT

Operation System -

Theory

2. Type of OSs and Kernels

Lecturer:

Sipan M. Hameed

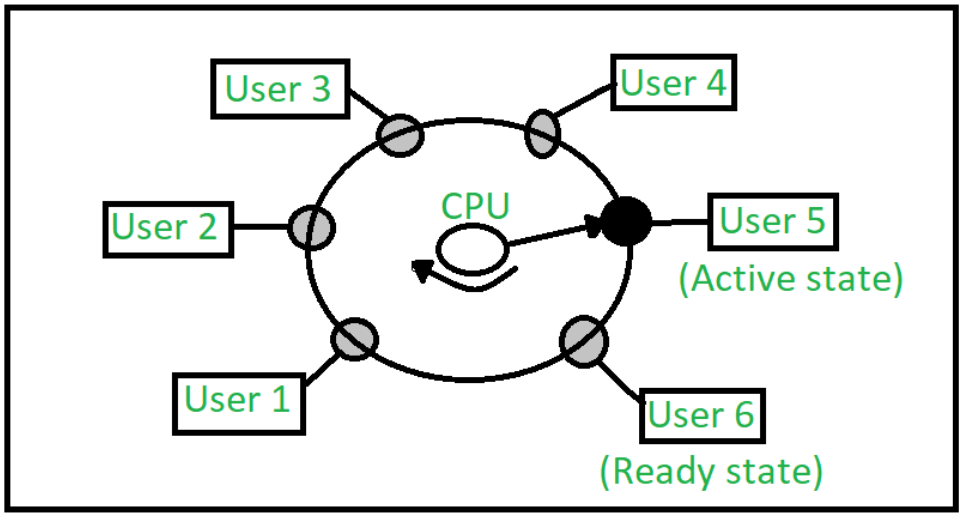
www.sipan.dev

2023-2024

Contents

Time Sharing Operating System

Multiprogrammed, batched systems provide an environment where various system resources were used effectively, but it did not provide for user interaction with computer systems. Time-sharing is a logical extension of multiprogramming. The CPU performs many tasks by switches that are so frequent that the user can interact with each program while it is running. A time-shared operating system allows multiple users to share computers simultaneously. With each action or order at a time the shared system becomes smaller, so only a little CPU time is required for each user. As the system rapidly switches from one user to another, each user is given the impression that the entire computer system is dedicated to its use, although it is being shared among multiple users.



In the above figure the user 5 is active state but user 1, user 2, user 3, and user 4 are in a waiting state whereas user 6 is in a ready state.

1. Active State – The user's program is under the control of the CPU. Only one program is available in this state.
2. Ready State – The user program is ready to execute but it is waiting for its turn to get the CPU. More than one user can be in a ready state at a time.
3. Waiting State – The user's program is waiting for some input/output operation. More than one user can be in a waiting state at a time.

Advantages of Time Sharing OS

1. Each task gets an equal opportunity.
2. Fewer chances of duplication of software.
3. CPU idle time can be reduced.

Disadvantages of Time Sharing OS

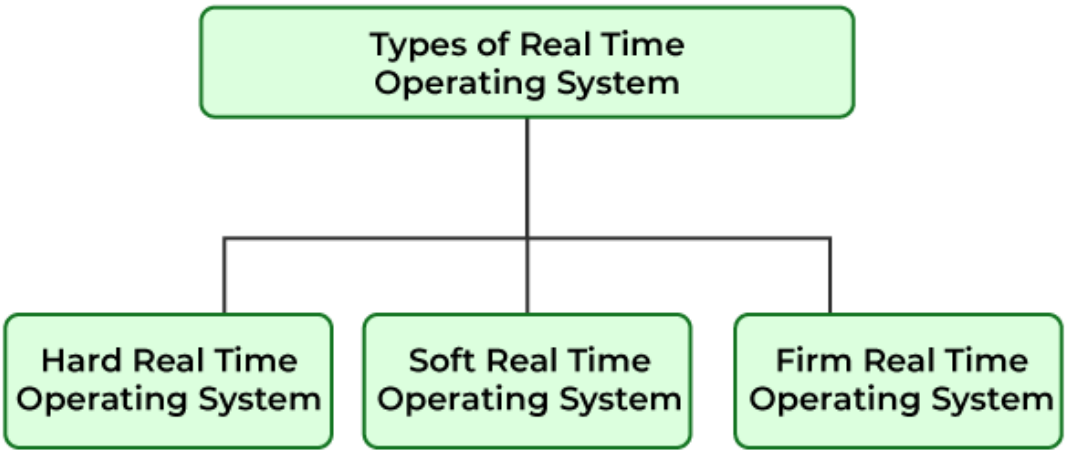
1. Reliability problem.
2. One must have to take of the security and integrity of user programs and data.
3. Data communication problem.

Real Time Operating System (RTOS)

Real-time operating systems (RTOS) are used in environments where a large number of events, mostly external to the computer system, must be accepted and processed in a short time or within certain deadlines. such applications are industrial control, telephone switching equipment, flight control, and real-time simulations. With an RTOS, the processing time is measured in tenths of seconds. This system is time-bound and has a fixed deadline. The processing in this type of system must occur within the specified constraints. Otherwise, This will lead to system failure.

Examples of real-time operating systems are airline traffic control systems, Command Control Systems, airline reservation systems, Heart pacemakers, Network Multimedia Systems, robots, etc.

The real-time operating systems can be of 3 types –



Types of RTOS

1. **Hard Real-Time Operating System:** These operating systems guarantee that critical tasks are completed within a range of time.

   For example, a robot is hired to weld a car body. If the robot welds too early or too late, the car cannot be sold, so it is a hard real-time system that requires complete car welding by the robot hardly on time., scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.
2. **Soft real-time operating system**: This operating system provides some relaxation in the time limit.

   For example – Multimedia systems, digital audio systems, etc. Explicit, programmer-defined, and controlled processes are encountered in real-time systems. A separate process is changed by handling a single external event. The process is activated upon the occurrence of the related event signaled by an interrupt.
3. **Firm Real-time Operating System**: RTOS of this type have to follow deadlines as well. In spite of its small impact, missing a deadline can have unintended consequences, including a reduction in the quality of the product. Example: Multimedia applications.

Advantages of RTOS

The advantages of real-time operating systems are as follows-

1. **Maximum consumption**
2. **Task Shifting**
3. **Focus On Application:**
4. **Real-Time Operating System In Embedded System:**
5. **Error Free**
6. **Memory Allocation**

Disadvantages of RTOS

The disadvantages of real-time operating systems are as follows:

1. **Limited Tasks**
2. **Use Heavy System Resources**
3. **Complex Algorithms**
4. **Device Driver And Interrupt signals**
5. **Thread Priority**
6. **Minimum Switching**

Comparison of Regular and Real-Time operating systems:

Regular OS	Real-Time OS (RTOS)
Complex	Simple
Best effort	Guaranteed response
Fairness	Strict Timing constraints
Average Bandwidth	Minimum and maximum limits
Unknown components	Components are known
Unpredictable behavior	Predictable behavior
Plug and play	RTOS is upgradeable

Network Operating System

Functions of the NOS (Network Operating System)

The following are the main functions of NOS:

- Creating and managing user accounts on the network.
- Controlling access to resources on the network.
- Provide communication services between the devices on the network.
- Monitor and troubleshoot the network.
- Configuring and Managing the resources on the network.

Types of Network Operating Systems

There are mainly two types of networks OS:

1. **Peer to Peer**
2. **Client/server**

advantages of NOS

- Highly stable due to central server.
- Provide good security.
- Upgradation of new technology and hardware can be easily implemented in the network.
- Provide remote access to servers from different locations.

Disadvantages of Network Operating Systems

- Depend on the central location to perform the operations.
- High cost to buying server.
- Regular updating and maintenance are required.

Examples of Network Operating Systems

1. Microsoft Windows Server
2. UNIX/Linux

Kernel in Operating System

Kernel is central component of an operating system that manages operations of computer and hardware. It basically manages operations of memory and CPU time. It is core component of an operating system. Kernel acts as a bridge between applications and data processing performed at hardware level using inter-process communication and system calls.

Kernel loads first into memory when an operating system is loaded and remains into memory until operating system is shut down again. It is responsible for various tasks such as disk management, task management, and memory management.

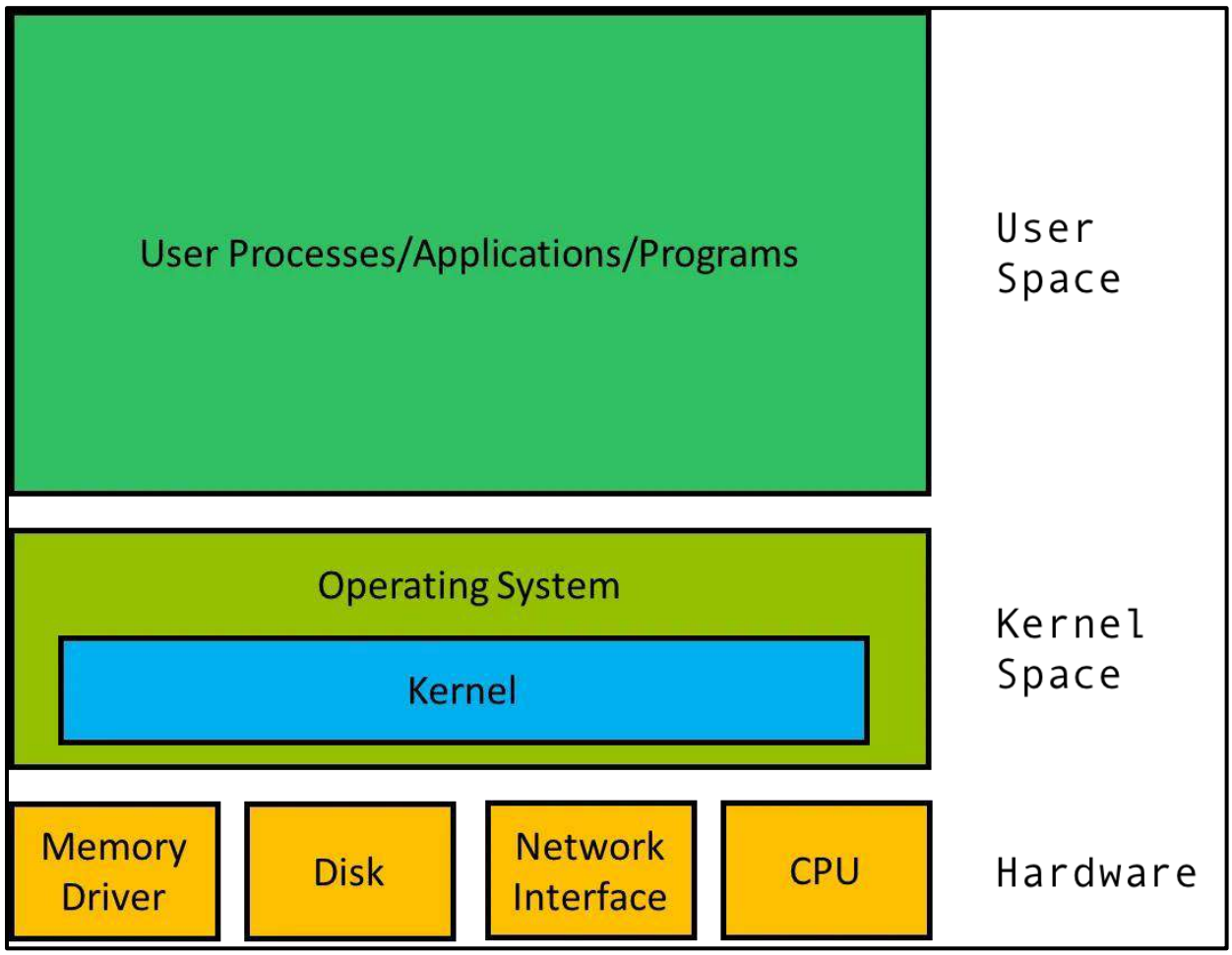


Figure show user space and kernel space

Objectives of Kernel :

1. To establish communication between user level application and hardware.
2. To decide state of incoming processes.
3. To control disk management.
4. To control memory management.
5. To control task management.

There are five types of kernels :

1. A micro kernel, which only contains basic functionality;
2. A monolithic kernel, which contains many device drivers.
3. Hybrid Kernel
4. Exokernel
5. Nanokernel

1. Monolithic Kernel –

It is one of types of kernels where all operating system services operate in kernel space. It has dependencies between systems components. It has huge lines of code which is complex.

Example:

Unix, Linux, Open VMS, XTS-400.

2. Micro Kernel –

It is kernel types which has minimalist approach. It has virtual memory and thread scheduling. It is more stable with less services in kernel space. It puts rest in user space.

It is use in small OS.

Example :

Mach, L4, AmigaOS, Minix, K42.

3. Hybrid Kernel –

It is the combination of both monolithic kernel and microkernel. It has speed and design of monolithic kernel and modularity and stability of microkernel.

Example :

Windows NT, Netware, BeOS.

4. Exo Kernel

It is the type of kernel which follows end-to-end principle. It has fewest hardware abstractions as possible. It allocates physical resources to applications.

Example : Nemesis, ExOS.

5. Nano Kernel –

It is the type of kernel that offers hardware abstraction but without system services. Micro Kernel also does not have system services therefore the Micro Kernel and Nano Kernel have become analogous.

Example : EROS etc.

Differences between Microkernel and Monolithic Kernel :

S. No.	Parameters	Microkernel	Monolithic kernel
1.	**Address Space**	In microkernel, user services and kernel services are kept in separate address space.	In monolithic kernel, both user services and kernel services are kept in the same address space.
2.	**Design and Implementation**	OS is complex to design.	OS is easy to design and implement.
3.	**Size**	Microkernel are smaller in size.	Monolithic kernel is larger than microkernel.
4.	**Functionality**	Easier to add new functionalities.	Difficult to add new functionalities.
5.	**Coding**	To design a microkernel, more code is required.	Less code when compared to microkernel
6.	**Failure**	Failure of one component does not effect the working of micro kernel.	Failure of one component in a monolithic kernel leads to the failure of the entire system.
7.	**Processing Speed**	Execution speed is low.	Execution speed is high.
8.	**Extend**	It is easy to extend Microkernel.	It is not easy to extend monolithic kernel.
9.	**Communication**	To implement IPC messaging queues are used by the communication microkernels.	Signals and Sockets are utilized to implement IPC in monolithic kernels.
10.	**Debugging**	Debugging is simple.	Debugging is difficult.
11.	**Maintain**	It is simple to maintain.	Extra time and resources are needed for maintenance.
12.	**Message passing and Context switching**	Message forwarding and context switching are required by the microkernel.	Message passing and context switching are not required while the kernel is working.
13.	**Services**	The kernel only offers IPC and low-level device management services.	The Kernel contains all of the operating system’s services.
14.	**Example**	**Example :** Mac OS X.	**Example :** Microsoft Windows 95, Linux