CSC 205

Operating System I (3 Units) - Comp

Introduction

Definition of Operating System

History " " "

Operating System Structure.

Operating System I

31/08/2024.

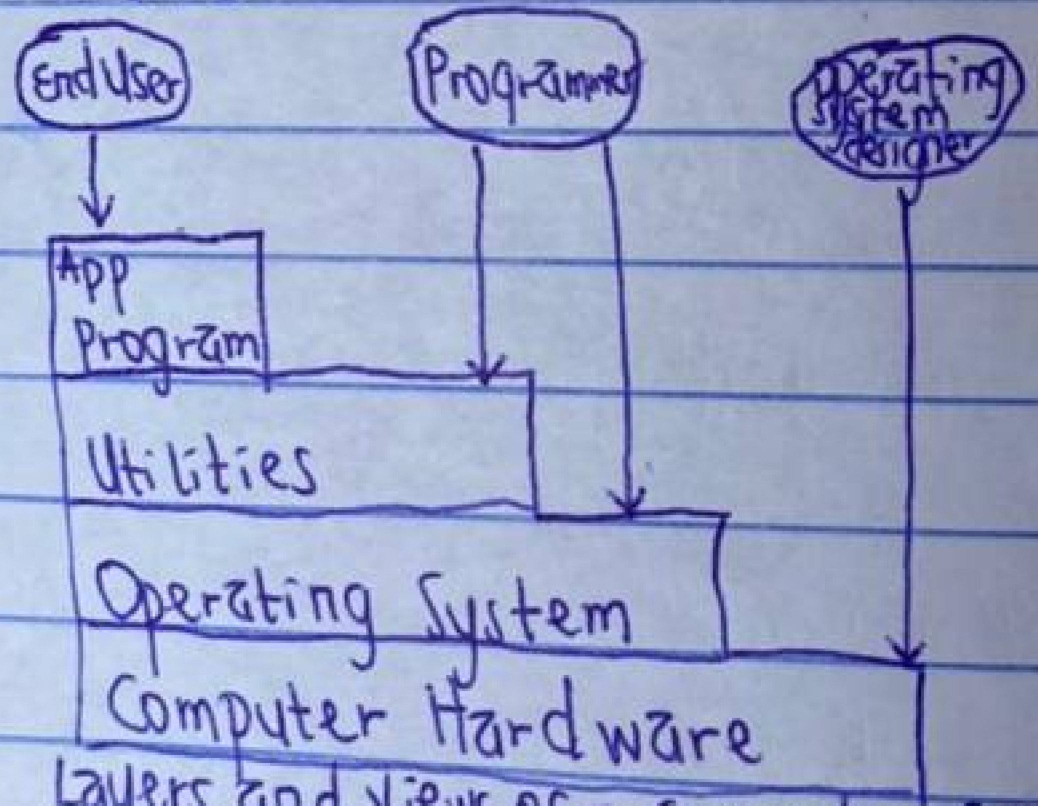
It's a program that Controls execution of Application programs and acts as an Interface between Application and Computer Hardware

Objective

Convinience

Efficiency.

Ability to Evolve.



Layers and views of a Computer System.

Provided,
Service rendered by Operating System.

It provides many Services ranging from program development, Program execution, controls of files accessing, System access, error detection and Response, Accounting, Access input/Output.

Operating System as a resource Manager.

A comput

and processing of data And also for the control of this funct
The Operating System is Responsible for Managing this
resourses.

Resources Manage by the Operating System.

- Processes
- Memory
- Files - Read/write.
- I/o devices.

Evolution of Operating System.

- Serial Processing
- Simple Batch System.
- Timesharing.
- Multiprocessing.

Serial processing is time consuming as there is no schedul
and set up time which result to Inefficiency. To Overco
this, they evolved and started Using Simple Batch Syster
To Eradicate Set up time, They came up with monitor,
Monitor was called Operating System back then, the
Innitiative of monitor eradicated the Set up time issues.
Operating went on and evolved to time sharing where ti

are issued to Operating System - hereby making it efficient.

Processes → Next Class.

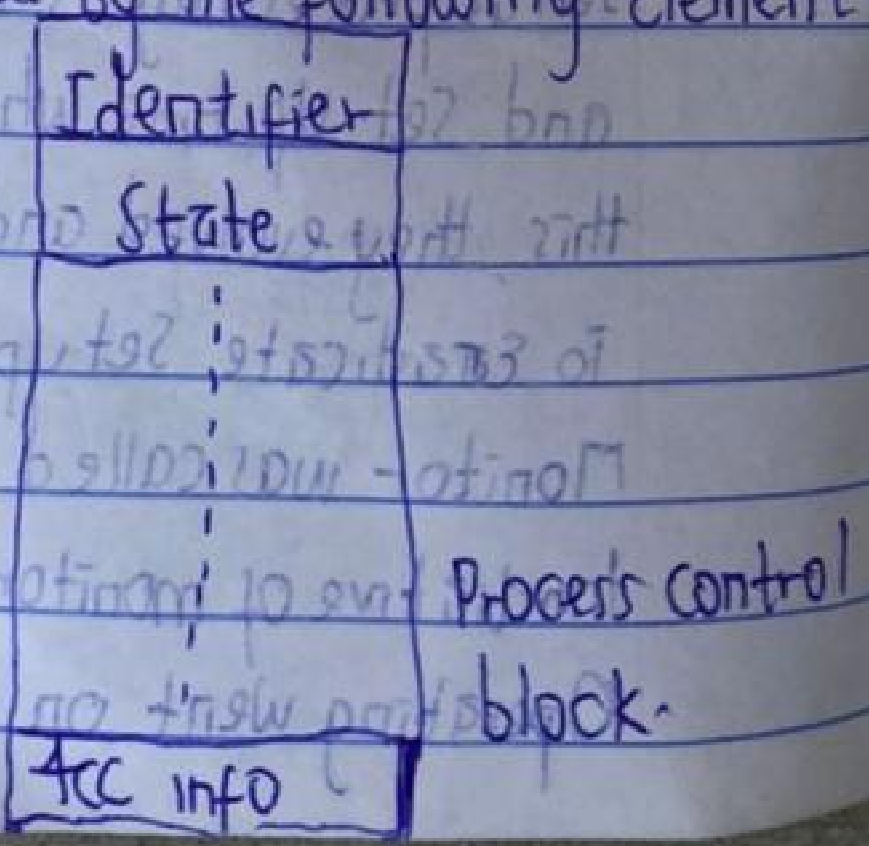
Process

A process is a program in execution. A process is also defined as an instance of a program running on a computer. It is also defined as the entity that can be assigned to and executed on a processor. We can say a process consist of two essential elements. Program code and a set of data.

What do we mean by program code? Program code are executable programs while set of data is the data associated with the code (i.e program code).

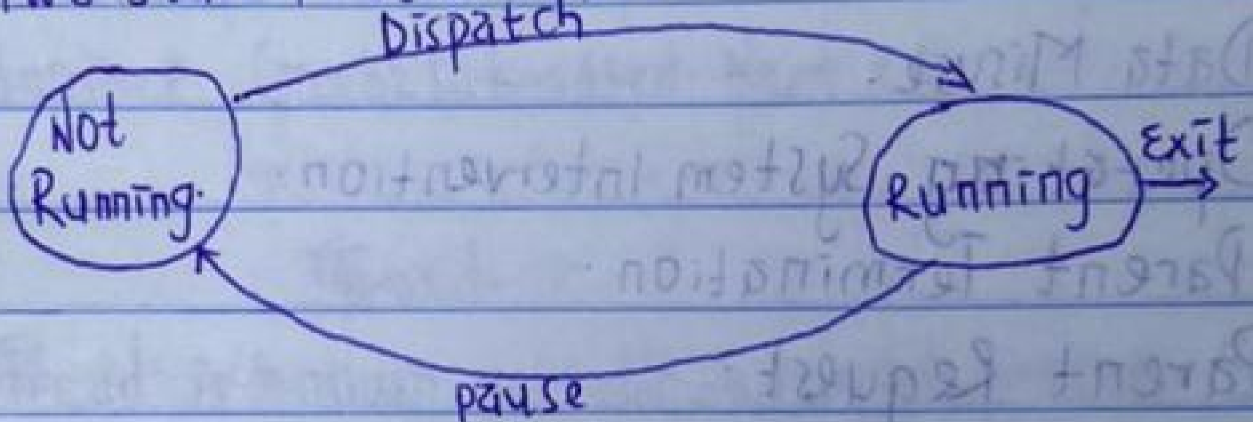
A process is characterised by the following Element.

- Identifier
- State
- Priority
- Program Counter
- Memory Pointer
- Context data
- I/O Status



Trace: Trace keeps track of sequence of Instructions that execute for a paritcular process.
Dispatcher: Dispatcher is the program that switches the processor from one process to another.

Two State Process Model



State transistion diagram.

Reasons for Process Creation.
- New Batch Job
- Interactive Logon
- Created by Operating System to provide Service.
- Spawned by Existing process

Reason FOR PROCESS TERMINATION.
- Time Limit exceeded.
- Memory Unavailable.

- Bound Violation (Trying to access what is not in its boundary)
- Protection Error
- Arithmetic Error
- Time Overrun
- I/O Failure
- Invalid instruction
- Data Misuse.
- Operating System Intervention.
- Parent Termination.
- Parent Request.

FIVE STATE PROCESS MODEL

$$\begin{tikzcd}
\text{New} \arrow[r, "\text{Admit}"] & \text{Ready} \arrow[r, bend left, "\text{Dispatch}"] & \text{Running} \arrow[l, bend left, "\text{Timeout}"] \arrow[r, "\text{Release}"] \arrow[dl, "\text{Event wait}"] & \text{Exit} \\
& \text{Blocked} \arrow[u, "\text{Event occurs.}"] & &
\end{tikzcd}$$

Possible State transition for a Process.

- Null $\longrightarrow$ New.
- New $\longrightarrow$ Ready (Admitted).
- Ready $\longrightarrow$ Running.
- Running $\longrightarrow$ Exit
- Running $\rightarrow$ Ready
- Running $\rightarrow$ Blocked.

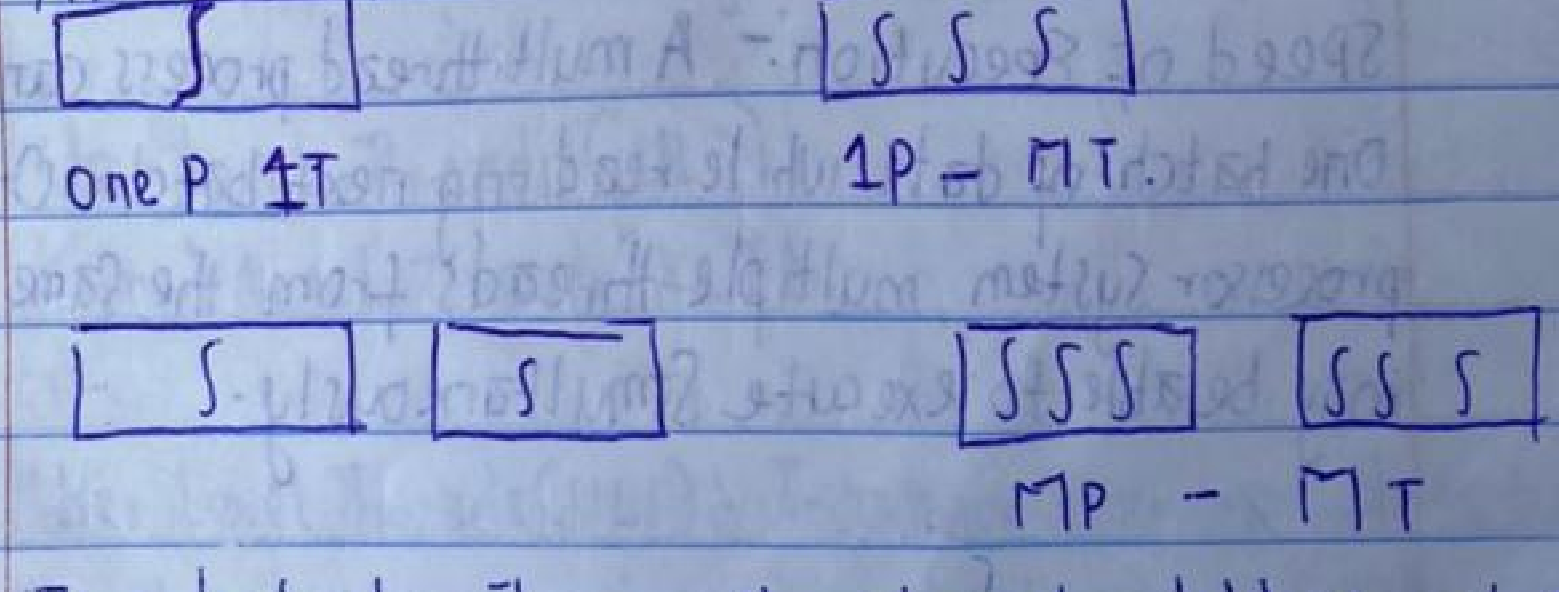
- Blocked -> Ready.
- Ready -> Exit (Process Timeout)
- Block -> Exit

Assignment
- Discuss the Condition for transistion from One state to Another. (To be submitted Next Class).

Thread.

Thread is known as a lightweight process. Every process has two Characteristics
Resource Ownership & Schedule. Resources can be Memory
When a process is making Use of resources it retain/ Acquire a Resources Ownership.

THREAD - PROCESS RELATIONSHIP.



Thread also has its own thread control block just as process has its own Process control Block.

Threads. 21-09-2024

Examples/uses of thread in a single User.

Foreground and Background ~~word~~ Problem : In a spreadsheet program One thread could display many and read user's input where another thread execute user Commands and Update the spread-heet. This arrangement increases the speed of the Application by allowing the program to prompt for the Next Command before the previous command is executed

Asynchronous Element : Can be Implemented As thread as a protection Against power failure. One can design a ~~One~~ word document to write to its RAM Buffer to disk Once every minutes. The thread can be created whose so job is to periodically backup data of a ram.

Speed of Execution :- A multithread process can compute One batch of data while reading next batch. On a multi-processor system, multiple threads from the same processes may be able to execute simultaneously.

KEY STATE OF A THREAD.

- Running
- Ready
- Blocked.

We ~~have~~ don't have Suspended State for thread.

The basic thread Operation Associated with a Change in thread State

- Spawn:- When a process give birth to another process it affect the State of thread.
- Blocked
- Unblocked
- Finished.

USER LEVEL AND KERNEL LEVEL THREADS.

There are two broad categories of thread Implementation which are

- User Level Threads (ULT)
- Kernel Level Thread (KLT)

1 User Level Threads (ULT):- In a pure User level thread facility all the work of thread Management is done by the Applicaton and the kernel is not Aware of the existence of threads. Any

application can be programmed to the multithread by Using a thread Library.

There are a number of advantages of ULT.

1 Thread Switching does not require Kernel mode priviledges.
2 Scheduling can be Application Specific
3 It ~~has~~ can ~~to~~ run on any OS.

It has two disadvantages. They are

1 Many System Calls are blocking:- Therefore when a ULT execute a system not only is that thread blocked but also all the threads within the process are blocked.
2. In a pure ULT a multithreaded Application cannot take advantage of Multiprocessing.

There are two major Solution to ULT disadvantages

1 Avoid thread and Adopt the Use of process.
2. By Using a technique called Jacketing.

The purpose of Jacketing is to Converting blocking. System call into non-blocking system calls when System call are made the Jacket will remove the blocked thread in Order not to affect the Unblocked threads.

KERNEL LEVEL THREAD

In KLT, all the work of thread management is done by the kernel. This approach will conquered the two ULT disadvantage.

CONCURRENCY.

Multiprogramming is a rudimentary form of parallel processing in which Several programs are run at the Same time on a Uniprocessor

Multiprocessing is a mode of Operation in which two or more processors in a computer Simultaneously process two or more different portions of the Same program.

Distributed processing is a Setup in which Multiple Individual central processing Units (CPU) work on the Same programs, functions or Systems to provide more capability for a computer or Other device.

Multiproмming :- The management of multiple processes within a Uniprocessor.

Multiprocessing :- The management of multiple processes within multiple processor.

Distributed Processing:- The management of multiple distributed Computer Systems.

KEY TERMS IN CONCURRENCY

- Atomic Operation
- Critical Section
- Dead Lock.
- Live Lock
- Mutual Exclusion.
- Race Condition.
- Starvation.

1 Atomic Operation: A Sequence of One or more statement that appears to be indivisible i.e. no other process can see an intermediate State or Interupt the Operation.

2 Critical Section:- A section of Code within a process that requires access to shared resources and that must not be executed while another process is in a Corresponding Section of Code.

3 Dead lock: A situation in which two or more processes are Unable to proceed because each is waiting for the Other to do something.

4 Live lock:- A situation in which two or More processes continously Change their State in response to changes in the other process without doing Useful work.

5 Mutual Exclusion:- Requirement that one process is in a Critical Section that accesses shared resources no Other process may be in a Critical Section that access any of those Shared Resources.

6 Race Condition:- A Situation in which multiple threads or processes read and Write a shared data item and the final result depends on the relative timing of their execution.

7 Starvation:- A Situation in which a runable process is Over-Looked dифinitely by the Scheduler Although it is able to proceed but it is never Chosen.

Principles of Concurrency 05/10/2021

* It is difficult for the OS to manage the allocation of memory resources.
It is very difficult to locate the programming error because result is non determinitic.
Sharing of Global Resources.

Consider the following Procedure:

```
Void echo()
{
Chin = getchar();
Chout = Chin;
 Putchar(Chout);
```

1 Process P_1 invokes echo procedure and is interrupted Immediately After getchar returns its value and stored it in chin. At this point, the most recently entered characters is stored in variable Chin.

2 Process P_2 is activated and invoke the echo procedures which runs to Conclusion, and display a single xter y on the screen

- Process P_1 resumed, by this time, the value x has been Overwritten in chin and lost. Instead chin contains y, this is transfereed to chout and displayed.

Design And Management issue that has to do with the Existence of concurrency

1) Operating System must be able to keep track of various processes. This done with the Use of process control block.

2) Operating system must allocate and disallocate various resources for each active process resources Like processor time, memory, files, I/o devices.

3) Operating System must protect data and physical resources of each process against Unintended inference by other process.

4) The functioning of a process and the output to be produced must be Independent of the speed at which it execution is carried Out relative to the speed of other Concurrent processes.

Process Interaction

1) Process Unaware of each Other.
2) Processes Indirectly aware of each Other.
3) Processes Directly aware of each Other.

Potential Control problem

Common Concurrency Mechanisms.

- Semaphores:- It is an integer Value Used for signaling among processes. Only three Opreations may be performed on a Semaphore.
- Binary Semaphore
- Mutex
- condition variable
- Monitor
- Event flags
- mailboxes/messages
- Spinlocks.

DeadLock can be defined as the permanent blocking of a set of processes that either compete for System resources or communicate with each Other

Consumable Resources is One that be Created (produced) and destroyed (Consumed) Typically, there is no Limit on the number of Consumable resources of a particular type

Resource Allocation Graphs is a Useful tool in characterizing the allocation of resources to processes ~~&~~ ~~the~~.

The conditions for Deadlock

possi & exist Mutual exclusion
possi & exist 2 Hold and wait
possi & exist 3 No Preemption.
existence 4 Circular Wait.