11.1 History of Operating Systems & 11.2 Processes-Reading

Notebook:	How Computers Work [CM1030]		
Created:	2019-10-09 10:09 AM	**Updated:**	2020-01-06 10:41 AM
Author:	SUKHJIT MANN		

Cornell Notes	**Topic:** 11.1 History of Operating Systems 11.2 Processes Reading	Course: BSc Computer Science Class: How Computer Work [CM1030]-Reading Date: December 10 2019

Essential Question:

What were early computers like and what are the processes that simulate to give the illusion of simultaneous execution in computers?

Questions/Cues:

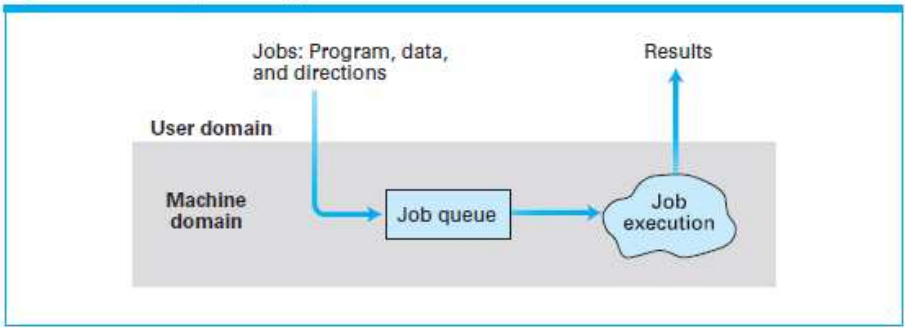
- What is a job?
- What happened when several users needed to use a machine in the early days?
- What was the initial job of an Operating System?
- What was the role of an Operator?
- What is Batch Processing?
- What is an Queue?
- What is Interactive Processing?
- What is real-time processing?
- What is time-sharing?
- What is Multi-programming?
- What is Multi-tasking?
- What is Load Balancing?
- What is Scaling?
- What is the role of the Scheduler within System's Kernel in terms of processes?
- What is the role of the Dispatcher within System's Kernel in terms of processes?
- What it meant by ready or waiting in regards to a process?
- What is a process/context switch?
- What is an interrupt and an interrupt handler?

Notes

- Job = the execution of a comp prog
    - handled as an isolated activity
        - machine was prepared for executing the prog, the prog was executed, then all tapes, punched cards, etc. had to be retrieved before the next prog preparation could begin
- When several users needed to share a machine, sign-up sheets were used to reserve machine for blocks of time in which machine was totally under user's control
- OS began as systems for simplifying prog setup & for streamlining transition between jobs

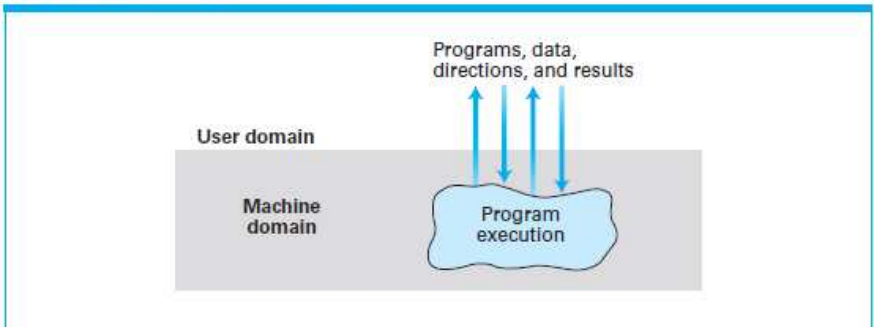
    - Early development was separation of users & equipment; eliminated people in & out of comp room
        - A comp operator was hired to operate machine. Anyone wanting a prog run was required to submit it, along with any required data & special directions about prog's requirements to operator & return later for results.
        - The operator loaded these materials into machine's mass storage where a prog called OS could read & execute them one at a time.
- Batch Processing = execution of jobs by collecting them in single batch, then executing them without further interaction with the user
    - jobs residing in Mass Storage for BP, wait for execution in a job queue
- Queue = storage organization in which objects (jobs) are ordered in first-in, first-out (FIFO) fashion.
    - No strict following of FIFO structure in most OSes, since job waiting in queue can be bumped by higher-priority job
- In early batch-processing systems, each jobs was accompanied by set of instructions explaining steps required to prepare machine for that particular job. Instructions were encoded using a system known as job control lang (JCL) & stored with job in job queue. When job selected for execution, OS printed instructions at a printer so they could be read and followed by comp operator.
- Major drawback to using comp operator as an intermediary between comp and its users is that users have no interaction with their jobs once they are submitted to operator.

Figure 3.1 Batch processing



- Interactive Processing = allows a prog being executed to carry on a dialogue with the user through remote terminals
    - In early days, terminals were a electronic typewritter by which user could type input and read comp's response that was printed on paper
    - To be successful, interactive processing must have that the actions of comp be sufficiently fast to coordinate with needs of users rather than forcing user to conform to machine's timetable.

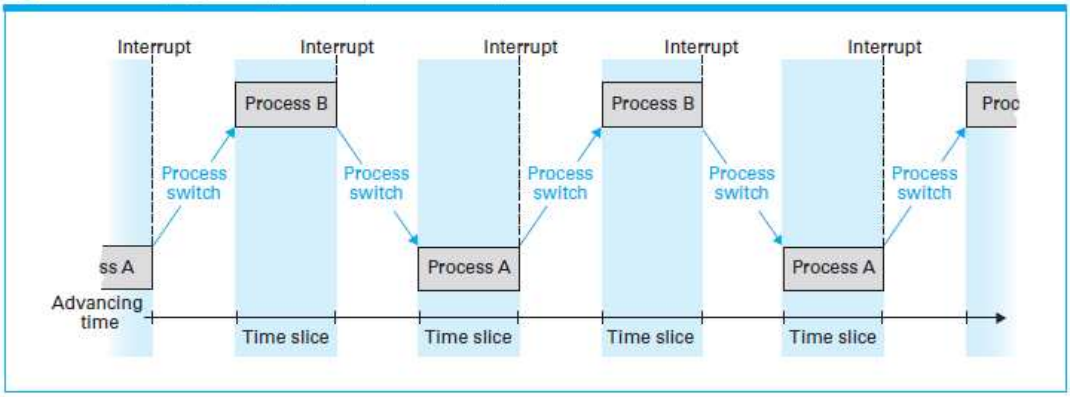
Figure 3.2 Interactive processing



- Real-time processing = Actions performed by a comp occur in real-time. Comp performs task in real-time means that the comp performs the task in accordance with deadlines in its (external real-world) environment.

- Time-sharing is feature in which OSes are designed to provide service to multiple users at the same time.
- Multi-programming= A way to implement time-sharing in which time is divided into intervals & then execution of each job is restricted to only one interval at a time. At end of each interval, current job is temporarily set aside & another is allowed to execute during next interval. By rapidly shuffling jobs back & forth, illusion of several jobs executing simultaneously is created.
    - Depending on job being executed, early time-sharing systems were able to provide real-time processing to as many as 30 users simultaneously.
- Multi-tasking = one user executing numerous task simultaneously.
- Load Balancing = dynamically allocating tasks to the various processors so that all processors are used efficiently)
- Scaling = Breaking tasks into a number of subtasks compatible with the number of processors available.
- Process = activity of executing a prog under the control of OS
    - Process state = current status of the activity
        - state includes current position in prog being executed (value of program counter) as well as the values in other CPU registers & associated memory cells; process state is a snapshot of machine at particular time
- Tasks associated with coordinating execution of processes are handled by scheduler & dispatcher within OS Kernel
    - Scheduler maintains record of processes present in comp system, introduces new processes to this pool, & removes completed processes from pool.
        - To keep track of all processes, scheduler maintains block of info in main memory called process table. Each time execution of prog is requested, scheduler creates new entry for that process in process table
            - Entry contains info such as memory area assigned to process (obtained from memory manager), priority of process & whether process is ready or waiting
- Process is ready if it's in a state in which its progress can continue; waiting if its progress is currently delayed until some external event occurs.
- Dispatcher, the component of kernel that oversees execution of scheduled processes. In time-sharing/multi-tasking system, this accomplished by multi-programming; dividing time in short segments, each called time slice (measured in mill or microseconds) & then switching CPU's attention among the processes as each is allowed to execute for one time slice.
- Process/context switch = procedure of changing from one process to another

Figure 3.6 Multiprogramming between process A and process B



- Each time dispatcher awards time slice to process, it starts timer circuit that will indicate end of slice by generating signal called an interrupt. CPU completes its machine cycle, saves its position in current process, & begins executing prog called interrupt handler which is stored at predetermined location in main memory. IH part of dispatcher, tells dispatcher how to deal with interrupt signal

Summary

In this week, we learned about the processes behind the scenes that execute in a timely manner to provide fast & easy access to a user using a computer.