6 Resource Management

6.1 Resource Management

Systems Resources

6.1.1 Identification of critical resources

System resources: hardware, software, trained personnel and supporting infrastructure.

Operating system: responsible with the management of hardware and software resources.

Primary memory

Primary memory:

- Also called immediate access storage
- Directly connected with computer
- Feeds the processor during fetch-decode-execute cycle

RAM and cache memory

Static RAM:

- fast and expensive
- used as cache

Dynamic RAM:

- slow and relatively cheap
- e.g. DDR-SDRAM

cache

- keep most frequently used programs and data
- *L1*: built on the processor, extremely fast but relatively small
- *L2*: built on the processor, not as fast as L1, more capacious
- *L3*: common to all processors on a multiprocessor chip, not as fast as L1 and L2

ROM

ROM

- Non-volatile
- Slower than RAM

- Holds *BIOS* (basic input/output system)
    - holds the *bootstrap loader*
    - manages data flow between the computer's operating system (OS) and attached devices, such as the hard disk, video adapter, keyboard, mouse and printer

Secondary storage

- mass storage
- low cost
- non-volatile
- direct access (hard disk, CD-ROM, SSD etc) and sequential access (e.g. tapes)

Processor speed

- *MIPS* (million instructions per second)
- Clock rate measured in *Hertz*

Bandwidth

- *Memory bandwidth*: rate at which data can travel from SRAM and DRAM to the processor.

Screen resolution

- Width x height in *pixels* (picture elements)
- E.g. Full HD 1920x1080 pixels, Full Ultra HD 7680x4320

Disk storage

- *HDD* (hard-disk drive)
- *SSD* (solid-state drive) – very expensive, very fast, electronic
- *SSHD* (solid-state hybrid drives) – hard-disk with solid-state cache

Sound processor

- *Sound card*: convert analogue to digital, digital to analogue, process multiple audio channels

Graphics processors

- GPU (graphics processing unit) – massively parallel processors
- Used in the entertainment industry
- Most are separate cards that use an expansion slot on the motherboard.

Network connectivity

- *NIC* (network internet card) to connect to a network; also *wireless NIC*.
- *4G* with a SIM card for tablets, smartphones

6.1.2 Availability of resources

Mainframe

- Large bandwidth
- Bulk data processing
- Large-scale transaction processing
- Great processing power, vast amounts of RAM
- Arrays of disks and backup tapes
- Hundreds of user terminals
- Runs different applications concurrently

Supercomputers

- Very fast
- Focus on mathematical calculations e.g. weather forecasting
- Performance measured in FLOPS (floating-point operations per second)

Server

- Made of software, hardware or both
- Client-server model
    - Client: asks for a service
    - Server: provides the service

PCs

- Popular OSs are MS Windows, MacOS, and LINUX

Laptops

- Long-lasting batteries
- Different weights and dimensions
- Camera, microphone, speaker
- Touch-screen

Tablets

- Touch-screen

- Virtual keyboard
- Common OSs are iOS, Android and Windows
- Hybrid tablet – detachable keyboard

PDA

- Personal Digital Assistant
- Used until 2010

Smartphone

- Many features and apps
- Wi-Fi connection etc.

Digital camera

- E.g. 30.4 Megapixel sensor
- Supports various image formats e.g. JPEG, RAW
- Can record video

6.1.3 Limitations of resources

Computer-generated imagery involves complex mathematics to produce realistic graphics. They need render-farms (high-performance computer systems).

For some problems multi-core processors are required.

6.1.4 Problems with insufficient resources

This part is about the development of operating systems.

Batch processing (in the old days it was the batch operating system): processing of transactions in a group or batch. No user interaction is required once batch processing is underway. This differentiates batch processing from *transaction processing*, which involves processing transactions one at a time and requires user interaction.

Multiprogramming: running programs in sequence but if one program is waiting for an event to happen another program is assigned the processor.

Multiprocessing: a computer using more than one CPU at a time.

Multitasking: this is multiprogramming plus *time-sharing* (each program is run for a small *time slice*, then stopped, another is executed for a time slice etc.).

Multithreading: *threads* are parts of the same program that can be run in parallel.

Multi-access: a number of users interact with the computer simultaneously. This is time-sharing.

6.1.5 The role of the OS

OS:

- Can be *single-user* or *multi-user*.
- Most OSs are written in C or C++.
- OS belongs to *system software*.
- Coordinates all the resources, software, and hardware.

Managing memory and processes

Memory leak: a case when memory is used and when its use is finished it is not deleted. More generally we have *resource leak* which is a resource which has not been released by a program which has finished using it.

Memory management: how the OS assigns memory to processes in memory.

Logical address: the address as appears in the program. *Physical address* is the real address inside the RAM. *Address binding* is the mapping of a logical address to a physical address. The address binding is done by the OS.

Virtual memory:

- RAM uses part of secondary storage, so it is virtually increased.
- RAM is divided in *pages*.
- If a page is needed from secondary storage, it is *swapped* with a page in RAM.
- Excessive page swapping causes *thrashing*.

Interrupt: message from a device or program that requests OS attention. Present program is stopped, and *interrupt handler* takes over.

Process: a program during execution.

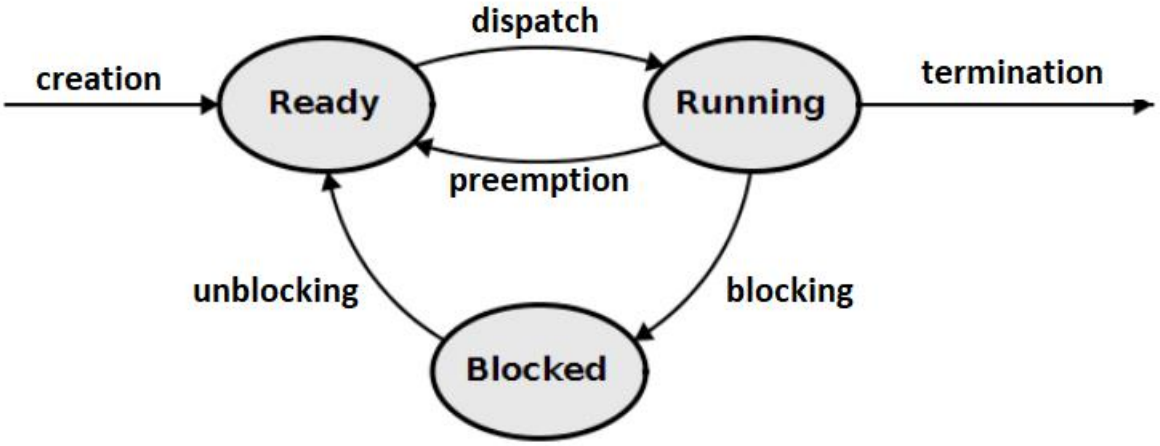
Scheduler (process scheduler): program (part of the OS) that selects the next process to run. It also decides when to stop a program that is using the CPU.

Preemptive Scheduling: it is the OS that decides which process and when will have the availability of the processor (or a processor).

Non-Preemptive Scheduling (cooperative scheduling): it is the process that currently is using the CPU that decides to release it for other processes.

A process can be in one of three *states*:

- Executing
- Ready (waiting)
- Blocked (suspended)



Moving from one state to another	…to executing	…to ready	…to blocked
From executing to…	------	Pre-emption stops a temporary process.	The process stops because it is waiting for an event to happen.
From ready to…	The dispatcher sends and order for the execution of the process.	------	------
From blocked to…	------	The event that the process was waiting for had happened.	------

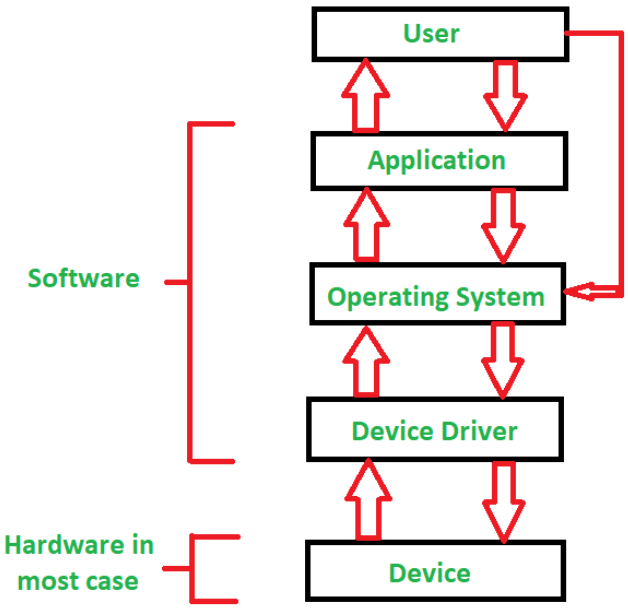
Pre-emption is the act, by the OS, of stopping a process that is using the CPU. This procedure is performed by the scheduler.

Managing peripherals

Device driver: a program for a specific hardware device to facilitate communication between the OS and the device.

Abstraction occurs when a particular view hides the details and complexities. In the diagram below, the device driver is an abstraction for the OS. It hides the

complexities of the device. For the OS, communicating with the device driver is much simpler than communicating with the device.



Managing hardware interfaces

For the user, the application is an abstraction that hides the details of the hardware.

Device manager: a Microsoft Windows application that allows users to view and control the hardware attached to the computer. When a piece of hardware is not working, the offending hardware is highlighted for the user to deal with. The list of hardware can be sorted by various criteria.

Event viewer: a component of Microsoft's Windows NT that lets administrators and users view the event logs (i.e. reports of what is happening in the system e.g. failure to start a component) on a local or remote machine.

6.1.6 and

6.1.7 OS resource management techniques

Scheduling

Task scheduler: application used to create and manage common tasks that the computer should carry out at specific times e.g. virus scans, backups, defragmentation etc.

Process scheduler:

- Decides which process to be assigned the CPU

- Decides when process should be removed from the CPU.
- Three types:
  - Short term
  - Medium-term
  - Long-term
- Scheduling criteria
  - *CPU utilization* (avoid as much as possible the CPU to become idle)
  - *Throughput* (amount of work in a unit time)
  - *Turnaround time* (the total time taken between the submission of a program for execution and the return of the complete output to the customer)
  - *Waiting time*: the total time spent by the process in the ready state waiting for CPU
  - *Response time*: how much time passes between the submission of a process until the start of its execution.
- Three scheduler types:
  - *Short term*:
    - This is also known as CPU Scheduler.
    - Runs very frequently.
    - The primary aim of this scheduler is to enhance CPU performance and increase process execution rate.
    - Decides which of the ready, in-memory processes is to be executed (allocated a CPU) after a clock interrupt, an I/O interrupt, an operating system call or another form of signal.
  - *Medium term*:
    - Removes the processes from memory (and from active contention for the CPU).
    - Performs swapping in the virtual memory environment.
  - *Long term*:
    - Also called admission scheduler.
    - Decides which jobs or processes are to be admitted to the ready queue (in main memory) i.e., when an attempt is made to execute a program, its admission to the set of currently executing processes is either authorized or delayed by the long-term scheduler.
    - It dictates how the split between I/O-intensive and CPU-intensive processes is to be handled.

- The long-term scheduler is responsible for controlling the degree of multiprogramming.

Context Switch

- Switching the CPU to another process.
- It requires saving the state of the old process and loading the saved state for the new process. This task is known as a *Context Switch*.
- The context of a process is represented in the *Process Control Block* (*PCB*) of a process; it includes:
  - the value of the CPU registers,
  - the process state (ready or blocked)
  - memory-management information
- When a context switch occurs, the Kernel saves the context of the old process in its PCB and loads the saved context of the new process scheduled to run.
- Context switch time is overhead.
- Typical speeds range from 1 to 1000 microseconds.
- Context Switching has become such a performance bottleneck that programmers are using new structures (threads) to avoid it whenever and wherever possible.

CPU scheduling algorithms:

- First-come first-serve (FCFS)
- Shortest-job-first (SJF)
- Priority scheduling
- Round Robin
- Multilevel queue scheduling
- Multilevel feedback queue scheduling

First-come first-serve (FCFS)

- The process which arrives first, gets executed first, or we can say that the process which requests the CPU first, gets the CPU allocated first.
- Is just like FIFO (First in First out)
- This is used in Batch Systems.
- It is easy to understand and implement as a program.
- The scheduler selects process from the head of the queue.
- A perfect real-life example of FCFS scheduling is buying tickets at ticket counter.

- Problems with FCFS Scheduling
    - It is non pre-emptive algorithm, which means the process priority doesn't matter.
    - Not optimal Average Waiting Time.
    - Resources utilization in parallel is not possible, and hence there is poor resource (CPU, I/O etc) utilization.

Shortest-Job First

- Shortest burst time (CPU time) or duration first.
- This is the best approach to minimize waiting time.
- This is used in Batch Systems.
- It is of two types:
    - Non pre-emptive
    - Pre-emptive
- To successfully implement it, the burst time/duration time of the processes should be known to the processor in advance, which is practically not feasible all the time.
- Can lead Starvation and can be solved using the concept of aging.

Priority Scheduling

- The process which is most urgent is processed first.
- Processes with same priority are executed in FCFS manner.
- The priority of process, when internally defined, can be decided based on memory requirements, time limits, number of open files, ratio of I/O burst to CPU burst etc.
- External priorities are set based on criteria outside the operating system, like the importance of the process, funds paid for the computer resource use, etc.
- Priority scheduling can be of two types:
    - Pre-emptive
    - Non pre-emptive

Round Robin

- Means one after the other, no priorities.
- A fixed time is allotted to each process, called quantum, for execution.
- Once a process is executed for given time-period that process is preemptied and other process executes for given time-period.
- Context switching is used to save states of preemptied processes.

Multilevel Queue Scheduling

- This class of scheduling algorithms has been created for situations in which processes are easily classified into different groups e.g. interactive and background processes.
- A multi-level queue scheduling algorithm partitions the ready queue into several separate queues.
- The foreground queue might be scheduled by Round Robin algorithm, while the background queue is scheduled by an FCFS algorithm.
- In addition, there must be scheduling among the queues. For example: The foreground queue may have absolute priority over the background queue.

Multilevel feedback queue scheduling

- In multilevel queue scheduling processes are permanently assigned to a queue on entry to the system and do not move between queues. This setup has the advantage of low scheduling overhead, but the disadvantage of being inflexible.
- Multilevel feedback queue scheduling, however, allows a process to move between queues.
- The idea is to separate processes with different CPU-burst characteristics. If a process uses too much CPU time, it will be moved to a lower-priority queue. Similarly, a process that waits too long in a lower-priority queue may be moved to a higher-priority queue. This form of aging prevents starvation.
- In general, a multilevel feedback queue scheduler is defined by the following parameters:
    - The number of queues.
    - The scheduling algorithm for each queue.
    - The method used to determine when to upgrade a process to a higher-priority queue.
    - The method used to determine when to demote a process to a lower-priority queue.
    - The method used to determine which queue a process will enter when that process needs service.
- It is the most general scheme and the most complex.

Policies and account management

User account defines the *privileges* (e.g. a system administrator has the privilege to install software) and *access rights* (e.g. read, write, delete, modify a file) of a particular user. A *username* and a *password* are used.

Interrupts

Interrupt: signal to the processor indicating that an event needs attention e.g. printer out of paper.

Processor performs:

1. Suspends current program e.g. A.
2. Saves state of program e.g. values of registers.
3. Executes *interrupt handler* (*interrupt service routine*, *ISR*) associated with the interrupt.
4. Continues running A or a process indicated by the scheduler.

2 types of interrupts:

1. *Hardware interrupt* e.g. moving the mouse
2. *Software interrupt* e.g. divide by zero (called an *exception* or *trap*)

Interrupts have different priorities.

Polling

This is the periodic checking of devices. Also called *busy waiting*.

6.1.8 Dedicated OS for a device

Two general approaches:

1. Take an existing OS and adapt it for the device.
2. Design an OS that will fit the particular needs of the device exactly.

Examples of dedicated OSs:

Android

- Mobile OS
- Developed by Google21
- Designed mainly for touchscreen mobile devices

Symbian

- Designed for smartphones.
- Originally developed for PDAs.
- Defunct

Others

- TinyOS (for low-power wireless devices)
- Tizen (for smartphones, PCs, cameras etc.)
- eCos

6.1.9 OS and complexity hiding

Drive letters

e.g. C for hard-disk or SSHD

Virtual memory

Input devices

The Java virtual machine

Java virtual machine (JVM) is a virtual machine (program) that enables a computer to run Java programs.

Deadlock

It is a situation that can arise when two processes hold resources and request other resources, from each other, at the same time. Process A holds a resource that process B wants, while process A requests a resource that process B holds. The result is that neither can continue.