Level 3 operating system

- A **system software:** consists of all programs that allow the computer and peripheral devices to function smoothly. Is made up of: **operating system and system utilities.**

- **Operating system:** Is a program; it helps run other programs. Provides an interface to the user & other programs which require the use of the computer's resources.
    - Step 1: Turn on computer and cpu will read bios.
    - Step 2: Initiates power on self-test which test if hardware is faulty eg: Bootable disc.
    - Step 3: Load OS
    - Step 4: customise and configure system
    - Step 5: load system utilities
    - Step 6: authenticate user

- Categories of Operating system are:
    - **Client stand online operating system:** used by a single user does not need to be connected to other systems to run eg: windows 7 osx
    - **Server operating system:** used in client/server network environments eg: linux, unix, windows nt server
    - **Embedded operating system:** found on rom chips in portable or dedicated devices (iphone os android os blackberry os)

- The fundamental **functions** of operating system are:
    - User Interface and user management:
        - Allows for: Gui
        - Login & logout
        - Accounting info
        - Access control: who gets privileges
        - Account maintenance
    - I/o Management: connects all device
        - Input & output services eg: mouse
        - Device management
    - File system management:
        - Providing an abstract view of secondary storage to user
        - Backup/recovery
        - Creation/deletion
        - Volume maintenance
    - Process management:
        - The running of programs
        - Process creation and deletion (think system32 kill task management)
        - Process loading
    - Memory management:
        - Involves memory allocation, swapping (main memory to cache for frequently used), and address translation (big
        - Simplest os needs to perform memory management
    - Cpu management
        - Cpu scheduling (if pipelining issues)

- Interrupt handling stop programs (reset button)

https://en.wikipedia.org/wiki/Track_(disk_drive)

• Each side is divided into tracks

• Tracks are numbered: 0 (outermost), 1(the next one), 2, and so on.

• A track is divided into sectors: smallest block that can be read

- The sectors are accessed by specifying: platter, side, track, and sector within a track.
- Each disk within a hard drive is called platter

• May contain a boot sector: contains operating system. Boot loader loads it eg: hard drive usb with os

- Found in sector 0

- Disc problems that may arise:
  - Program may crash and bad sectors cause problems
  - They can fix directory but can not recovery lost info

- **File system:** deals with how data is stored and retrieved (it's a group so fat32 not .txt). So files are part of file system. So it is where files are stored (organised)
  - Concerns: Storage of files
  - Hierarchical organisation (deals Folder, subfolder)
  - Manipulation (editing file, creating file, change access rights of files etc)
  - Access of files
  - Retrieval of files
  - Part of Os system. Depends on operating system some operating systems organise/store files using different methods ie fat32 both
  - Allows device independence: so you can use files on other devices

  - EG: FAT32 and NTSF

- What can you do with a file system/functions
  - Creating a file:
    - Find space for file
    - Add entry to directory
  - Writing a file: (more data)
    - Find space in device
    - Maintain a write pointer into the file
    - Use a system call to specify the data to be written

    - Reading:
        - Maintain read point into the file
        - Use a system call to specify the data to be read
    - Deleting a file:
        - Removes entry
        - Updates free space information (so 1gb to now 2gb free)
    - Truncating (to shorten, or lengthen): So make files shorter by resizing by deleting information. All about making files shorter/larger
        - Causes: Change of entry
        - Causes: updating the free information (so either 1gb to 2gb or 3gb to 2 gb)
    - Seeking: (seek within a file. So excel is the file you want to jump from column a to column b that's seeking)
        - Maintains seek pointer within the file

- Os dependent file system:

- Characteristic differences between file systems:

    - Maximum file size
    - Maximum name size
    - Compression/encryption availability
    - Access checking
    - Directory structure (hows it stored)
    - File allocation

- File system Information:
    - So all file system must store information about the files (properties)
    - The meta data stored include:
        - Name
        - Size
        - Distributor
        - Permission
        - Location

- Three main types of disk file allocation: (1. So if we have file 1 2 3 how do we put those files and if I 1 file that's large and can go across multiple sectors how do you organise it?)
- **Contiguous:**
    - The simplest method
    - Linear ordering
    - Each file placed in contiguous cluster
    - Large enough space needed for whole file (need 1 large block to take full file)
    - Performance impact bad
- **Linked file allocation: (solves issues of need 1 block for full file)**
    - Each file block links directly to next file across block but linking is time consuming as it needs to go 1 to 1 mapping.

    - Sequential file access is still fast (block 1 to block 2)
    - Non-sequential becomes slow
    - Files become fragmented so slows down access
- **Indexed file location: (so points where to go instead of linking)**
    - Files don't have to be contiguous
    - Multiple link blocks can be used if file is large
    - A link block keeps track of where file parts are stored

- **Files:**
    - Files are a collection of bytes on the secondary storage.
    - A large file would be sitting in memory: in data clusters
    - May not be contiguous (sharing)
    - So the files (collection of bytes) are translated by a program to execute function
    - File extension ie (fat32) tells os which applications you will need to open
    - Part of file system

When we want to do a job/run application we create a process and that process will be thrown to cpu in a multithreaded way (so job split up in many process) and the operating system we decide when to perform task. Each process/thread will have a state.

- **Process states:**
    - **Active:** a process is said to be active if it is running in cpu
    - **Ready:** a process can be in ready state if it could use a cpu if one was available
    - **Blocked:** a process is in this state when the process needs to wait for some event to happen before it can proceed
    - **Terminate:** process completes

- **Round robin scheduling:** process in which the processes are to be managed. So how exactly are the processes done what order how etc.
    - Everyone will get a fair chance
    - The simplest way to schedule the process
    - CPU divided into time splices
    - A real clock generates pulses at regular and frequent intervals

- **Memory management** (performed by operating system): the two main ways to manage memory is using: **frames or segmentation**

- **Memory management using frames: (fixed)**
    - The physical memory ram is divided into equal size pages.

- When a program (now converted to processes) are loaded in memory the processes will use up more than 1 frame. The processes will be divided up into equal size pieces.
- The pieces will try and fill the equal size pages. But normally the processes will not have the same size as an exact multiple of page frame.

- **Internal fragmentation:** is where a page is used but not filled fully and therefore not available anymore. Ie: a 5k program fills 2.5 pages. (half page is not filled).

- **Memory management using segments:**

  - The physical memory in ram is divided into variable sized segments (so the memory is divided according to how much the program needs
  - Process are loaded in memory, they are divided into variable sized logical segments of memory

- **External fragmentation:** when a process is completed and new process comes but the new process does not fill the page frame completely thus creating new unfilled fragments.

- **Virtual memory:**
  - Proxy for ram found on hard disc.
  - Used when lot of programs opened
  - Pages are created on hardisc that are same size as pages on ram
  - The page frame on hardisc and page frame on ram may want to interchange or the page on hardisc gets added to ram page. This is to execute the large processes
  - Page frame keeps track of which frame is in ram and which in disk