***Types of Software***

Operating Systems

Utility Software

Application software

Code development tools (IDEs, debuggers)

User gives command ->

Operating system works out what is meant ->

Translates it into a form the computer can understand ->

Sends the required commands to the required hardware/software components ->

Deals with any information that comes back ->

Send information back to the user.

An operating system has three main functions:

1. Manage the computer’s hardware and interface with that attached
2. Provide a user interface
3. Execute and provide services for applications software

By doing this it means the user does not need to care about what hardware is installed or how it works – we can give commands and they will work.

Largely facilitates the application layer from the OSI Model.

Almost every device has an OS

However, some may be very basic.

Routers, switches, tablets, computers, printers can even have OS

Partly this is because devices have become so complicated partly so it can communicate with other devices.

Embedded Devices

Embedded devices support very little configuration of software and usually have one specific hardware configuration.

These still have OS, but they can be simplified down.

Examples of embedded devices:

**Central heating system** – The OS in this device would need to be able to take command from one device and be able to transmit it to other heaters connected to it.

**Fitness trackers** – The OS in this device needs to be able to gather data related to your heart rate, body temporary, and the number of footsteps, which is further sent to servers like LTE.

**ATMs** – The OS in this device needs to be able to retrieve transactional data and show the correct display. It needs to communicate with other banks or ATMs. Needs to keep everything secure.

Security on a computer is handled at various levels primarily by an operating system.

Authentication

Restriction

Backup

File extension rules

Encryption

Detecting threats

Jobs the OS does - Communication:

* Providing “phone book” of devices and how to contact
* Directing traffic
* Agreeing communication protocols
* Preparing communication

**Performance**

Any resources being used to run the OS take away from the program – if an OS needs more than that’s less for the program.

Operating systems that use efficient coding – using less instructions to do the same thing – will provide a sizeable performance boost.

Links in with how well the software is written for that exact hardware. Easier to target the programming to ONE hardware configuration. Writing for multiple means often much more inefficient programs.

**OS as a platform**

The OS also affects by being able or not able to run additional software that helps improve performance.

This could be to do with programming languages (C++ a low level language efficient, Java a higher level language, is less so) or graphics as shown here :Linux and windows.

Some OS also run as a virtual machine; this means using some resources to manage a separate area that acts as though it is one system. E.g. Could create a VM that pretends to be an android phone on windows hardware.

**VMs**

Virtual machines are used to ensure even of differing hardware all apps work the same.

Means often there is a big performance overhead losing huge amounts of resources to both run AND manage

Android does this as does some Xboxes (to run windows like apps)

Drawbacks:

* Virtual machines are less efficient than real machines because they access hardware indirectly. Running VM software on top of the host operating system means that it will have to request access to storage and memory from the physical device. This process can impact speed, so it is vital to work with IT experts who know how to balance resources between physical and virtual systems.
* Because a virtual machine can be spun up in minutes, server sprawl is something administrators need to mitigate through proper processes. Because a new VM allows a developer a fresh start, we have seen instances of 20 or 30 VMs living on a network where only 3 or 4 were necessary. These semi-defunct VMs drain host resources so each organization should have a clear set of rules for provisioning new VMs and for shutting down old ones.
* While a properly structured VM can’t infect a host, a weak host system can impact its VMs. This usually happens when there are bugs in the operating system. If two or more virtual machines relate to each other, the infections could spread to others as well.

Benefits:

* Can create multiple OS environments on the same server
* Virtual machines can provide an instruction set architecture, or ISA, which is different than the hardware host. The ISA serves as the interface between software and hardware.
* When you create your virtual machine, you create a virtual hard disk. Thus, everything on that machine can crash but if it does, it won’t affect the host machine.
* There are security benefits to running virtual machines. For example, if you need to run an application of questionable security, you can run it in a guest operating system. If the application causes damage, it will affect only the guest VM, and the issue will vanish once that VM is shut down. Virtual machines also allow for better security forensics by monitoring guest operating systems for deficiencies and allowing the user to quarantine it for analysis.

**Types of OS**

What is the purpose and function (including examples of use) of the following types of OS?:

* Batch operating system
* Multitasking/time-sharing operating system
* Real-time operating system
* Network operation system
* Mobile operating system

**Batch OS**

OS executes programs or instructions on a first come, first serve basis

Commands are never given directly to the OS instead loaded in batches by one single operator

Commands and programs are prepared externally to the OS/ device and then loaded.

It’s very efficient with CPU resources if fully utilized but inefficient with human time – giving and transporting programs takes too long.

**Time-sharing operating systems**

This is where multiple people use one computer and its resources at the same time.

“Time” on the CPU is shared between users, meaning the CPU executes things very quickly swapping between users.

Here the target is to reduce response time to the user.

**Real-time operating systems**

In real time operating system response time is so fast that it can be thought of as almost instant.

This means the OS can respond to things almost instantaneously in comparison to other OS.

Ideal for things that are time critical - they usually forego secondary storage.

**Network operating systems**

Runs on a server and gives the server the capability to manage data, users, groups, security, applications, and other networking functions.

Allows shared file and printer access among multiple computers in a network.

Maintains information about where on the network information and devices are located and how to access.

**Mobile OS**

It is often two OS – baseband (which is a RTOS) and standard (e.g., Android and iOS)

According to various sources the world is now the mobile majority – not always in all countries but as a whole – even the UK and US on some days!

Nowadays there is a convergence between mobile and desktop – the main difference now being the interface device and the processor instructions set they are built for.

The difference between Batch, Time-sharing, and Real-time OS is that Batch executes instructions on a first come first served basis, whereas time-sharing allows multiple people to work on the same computer at the same time, and the real-time executes instructions so fast it’s almost instantaneous.

**Defragmentation:**

Disk defragmentation occurs when a file is broken up into pieces to fit on the disk.

When a file is spread out over several locations, it takes longer to read and write resulting in slow computer performance.

SSDs do not suffer from defragmentation- this is due to the way they store and retrieve data.