

# Application Software



# Lesson objectives

- 11.3.2.1 summarise the selection of generic application software for a range of tasks , e.g. word processor, spreadsheet, desktop publisher (DTP), presentation software, graphics packages (bit mapped and vector graphics), and justify the choices
- 11.3.2.2 assess the advantages and disadvantages of a range of generic application software
- 12.3.2.1 describe the different types of application software and the criteria for selecting appropriate software for particular purposes

# Software Types

Application  
software

System  
software

General  
Purpose  
software

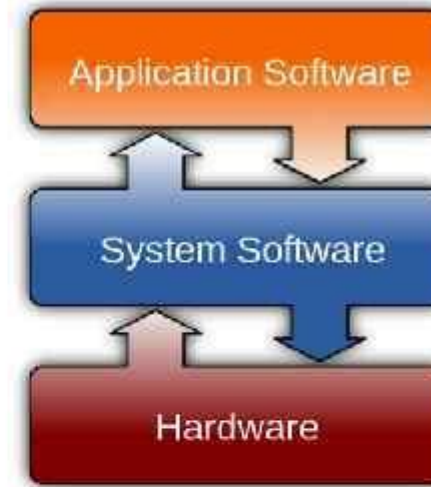
Special Purpose  
software

Bespoke  
software

# System software

A set of programs designed to control and manage the operations of the computer hardware.

The software allows application programs to execute properly



**Operating System** - make the computer hardware conveniently available to the user and also hide the complexities of the computer's operation

**Translator software**  
Assembler  
Compiler  
Interpreter

**Utility programs**  
that include file management, copy, paste, delete, file searching, disk defragmenter, disk cleanup.

**Library programs** are a compiled collection of subroutines

System software

Application software

general purpose software

special purpose software

bespoke software

# Application software

- Includes programs that do real work for the user. They are created to perform specific tasks for a user.



categories of Application software



# General Purpose software

Word processing  
Presentation  
Desktop publishing  
Spreadsheet  
Database Management  
Graphics  
Communication  
Computer Aided Design  
Games

This software is also called off the shelf. You can buy it at a shop or download it on line.

The software is written for a wide audience and not all of the features are used. The software is relatively cheap and usually well tested.

When giving examples

**NEVER use brand and company names**

Microsoft Word



Word processing



System  
software

Application  
software

general purpose  
software

special purpose  
software

bespoke  
software

# Special Purpose software

- Special purpose application software is a type of software created to execute one specific task.

For example

Movie editor, sound editor, photo editor, web page design and development.

Tax calculating system for accountants

Computer aided design for graphic designers or architects



# Bespoke software

- Bespoke application software is tailor made for a specific user and purpose. For example a factory may require software to run a robot to make cars, however, it is the only factory making that car in the world, so the software required would have to be specially built for the task.

Advantage	Disadvantages
Software is built for and will meet your precise needs	Software will be expensive as you have to cover all of the production costs
	It may take some time to develop the software
	The software is more likely to be buggy as it probably won't have thousands of clients using and testing it

System  
software

Application  
software

general purpose  
software

special purpose  
software

bespoke  
software

---

The operating system's tasks, in the most general sense, fall into six categories:

- 1) Processor management
- 2) Memory management
- 3) Device management
- 4) Storage management
- 5) Application interface
- 6) User interface

## 1. Processor Management

-  
The heart of managing the processor comes down to two related issues:

- Ensuring that each process and application receives enough of the processor's time to function properly
- Using as many processor cycles as possible for real work

The basic unit of software that the operating system deals with in scheduling the work done by the processor is either a **process** or a **thread**, depending on the operating system.

## **2. Process Control Block**

All of the information needed to keep track of a process when switching is kept in a data package called a process control block. The process control block typically contains:

- An ID number that identifies the process
- Pointers to the locations in the program and its data where processing last occurred
- Register contents
- States of various flags and switches
- Pointers to the upper and lower bounds of the memory required for the process
- A list of files opened by the process
- The priority of the process
- The status of all I/O devices needed by the process

### 3. Memory Storage and Management

When an operating system manages the computer's memory, there are two broad tasks to be accomplished:

1. Each process must have enough memory in which to execute, and it can neither run into the memory space of another process nor be run into by another process.
2. The different types of memory in the system must be used properly so that each process can run most effectively.

The first task requires the operating system to set up memory boundaries for types of software and for individual applications.

#### **4. Device management**

The path between the operating system and virtually all hardware not on the computer's [motherboard](#) goes through a special program called a driver. Much of a driver's function is to be the translator between the electrical signals of the hardware subsystems and the high-level programming languages of the operating system and application programs. Drivers take data that the operating system has defined as a file and translate them into streams of bits placed in specific locations on storage devices, or a series of laser pulses in a printer.

## 5. Application Program Interfaces

Just as drivers provide a way for applications to make use of hardware subsystems without having to know every detail of the hardware's operation, **application program interfaces** (APIs) let application programmers use functions of the computer and operating system without having to directly keep track of all the details in the CPU's operation. Let's look at the example of creating a hard disk file for holding data to see why this can be important.

## 6. User Interface

Just as the API provides a consistent way for applications to use the resources of the computer system, a **user interface** (UI) brings structure to the interaction between a user and the computer. In the last decade, almost all development in user interfaces has been in the area of the **graphical user interface** (GUI), with two models, Apple's Macintosh and Microsoft's Windows, receiving most of the attention and gaining most of the market share. The popular open-source Linux operating system also supports a graphical user interface.

