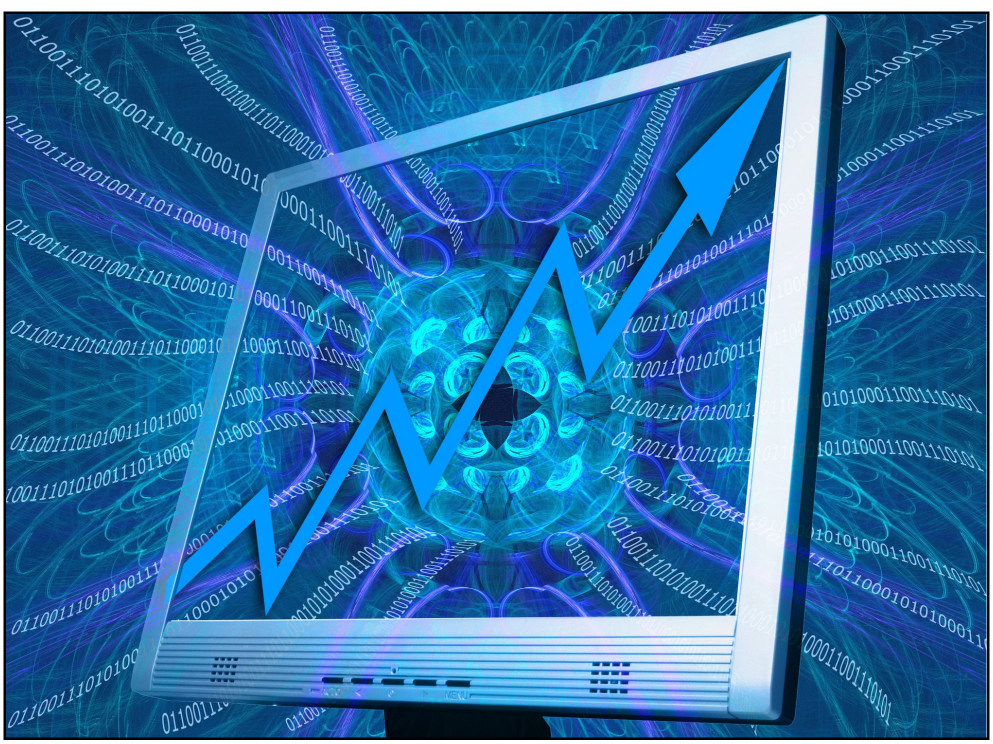
The Easy Guide To

Computer

Operating Systems



By Larry Miller

Table of Contents

INTRODUCTION

Our body couldn't function without our brains. The brain tells the various pieces of our body how to work and how to interact. Without a brain, we wouldn't be able to do anything at all.

An operating system is kind of like the brain of a computer. You have a bunch of hardware like the CPU tower, the monitor, and the keyboard, but without a CPU, they can't do anything but power up and turn on. The operating system organizes files and lets the hardware know what it should do.

In the early days of computers, there was just one operating system. As computers progressed, the OS turned into MS-DOS, but computers really weren't capable of doing much without software. Then Bill Gates came along.

With the founding of Microsoft, the computer operating system came into its own and brought computers to new levels of functioning and technology. Although the brand names of operating systems are few, they do perform different tasks depending on the requirements of the computer user.

While the dominant OS today would be Microsoft Windows, there are other types of operating systems that offer different features. Those would include Linux, UNIX, and OS X.

In our technological age, there are operating systems in more than just computers. Many of the sophisticated new cell phones have their own operating systems, and wireless access points have their own OS to provide wireless internet to customers. In fact, the computer in a cell phone today is more powerful than a computer was twenty years ago.

As you can see, the operating system technology has evolved and is continuing to evolve. It seems like Microsoft is always coming out with a new and better operating system which leads people to wonder whether or not the system they are currently using is really the best one.

It can be confusing. But it doesn't have to be. In the pages of this book, we'll explore operating system in depth. You'll learn about what they do, how they work, and what needs specific systems can meet. Ultimately, the choice is a matter of preference, but it helps to be informed on what you are really getting when choosing an OS.

WHAT IS AN OPERATING SYSTEM?

An operating system – commonly referred to as an OS – is a set of computer programs that manage the hardware and software resources of a computer. The OS processes electronic devices with a rational response to commands that are approved by the system.

At the foundation of all system software, an operating system performs basic tasks like controlling and allocating memory, prioritizing system requests, controlling input and output devices, facilitating the network, and managing files. The OS can also provide a graphical user interface for higher functions. Essentially, the OS forms a platform for other system software as well as application software.

The operating system is the most important program that runs on a computer. Without an operating system, your computer would not work. It would not be able to process requests for print, simple calculations, or any other function. It is really the brain that runs the equipment.

For larger system, the OS has great responsibilities than with a PC. In larger systems, the operating system is kind of like a traffic cop. It makes sure that different users and programs running at the same time on different systems don't interfere with each other. It also acts as a security guard making sure that unauthorized users are not able to access the system.

There are four classifications of a computer operating system. They are:

- **Multi-User:** Allows two or more users to run programs at the same time. Some operating systems permit hundreds or even thousands of concurrent users

- **Multi-Processing:** Supports running a program on more than one CPU

- **Multi-Tasking:** Allows more than one program to run concurrently

- **Multi-Threading:** Allows different parts of a single program to run concurrently

- **Real Time:** Responds to input instantly. General-purpose operating systems, such as DOS and UNIX, are not real-time.

Operating systems provide a software platform on top of which other programs, called application programs, can run. The application programs must be written to run on top of a particular operating system.

Your choice of operating system, therefore, determines to a great extent the applications you can run. For PCs, the most popular operating systems are DOS, OS/2, and Windows, but others are available, such as Linux.

In any device that has an operating system, there's usually a way to make changes to how the device works. This is far from a happy accident; one of the reasons operating systems are made out of portable code rather than permanent physical circuits is so that they can be changed or modified without having to scrap the whole device.

For a desktop computer user, this means you can add a new security update, system patch, new application or often even a new operating system entirely rather than junk your computer and start again with a new one when you need to make a change.

As long as you understand how an operating system works and know how to get at it, you can in many cases change some of the ways it behaves. And, it's as true of your cell phone as it is of your computer.

So, essentially, when you turn on your computer, the first program is a set of instructions kept in the computer's read only memory. These instructions examine the system hardware to make sure everything is functioning properly. This power-on self test check the CPU, the memory, and the basic input/output systems (BIOS) for errors and stores the result in a special memory location.

Once the test has successfully completed, the software loaded in ROM (sometimes called the BIOS or firmware) will begin to activate the computer's disk drives. In most modern computers, when the computer activates the hard disk drive, it finds the first piece of the operating system: the bootstrap loader.

The bootstrap loader is a small program that has a single function: It loads the operating system into memory and allows it to begin operation. In the most basic form, the bootstrap loader sets up the small driver programs that interface with and control the various hardware subsystems of the computer.

It sets up the divisions of memory that hold the operating system, user information and applications. It establishes the data structures that will hold the myriad signals, flags and semaphores that are used to communicate within and between the subsystems and applications of the computer. Then it turns control of the computer over to the operating system.

It might be helpful for you to know the history of operating systems.

HISTORY OF OPERATING SYSTEMS

By Larry Miller

The earliest of computers didn't have an operating system. By the early 1960's, commercial computer vendors were supplying quite extensive tools for streamlining the development, scheduling and execution of jobs on batch processing systems.

Through the 1960's, several concepts were developed which drove the development of operating systems. The IBM System 360 produced a family of mainframe computer that served consumers with differing capacities and prices. A single operating system was planned for these computers rather than developing generic programs for every individual model.

This concept of a single OS that will fit an entire product line was crucial for the success of System 360. In fact, IBM's current mainframe operating systems are distant relatives of this original system. The advantage to this is that applications written for the OS 360 can still be run on modern machines.

The OS 360 system also contained another important advance affecting today's computers: the development of a hard disk permanent storage device which IBM called DASD.

A second key development was the concept of time sharing. Time sharing involves sharing the resources of expensive computers among multiple computer users interacting in real time with the system. What that essentially means is that all of the users have the illusion of exclusive access to the machine. The most famous of time sharing system was called Multics.

By Larry Miller

Multics served as an inspiration to a number of operating systems developed in the 1970's. Most notably was the Unix system. Another commercially popular mini-computer operating system was VMS.

The first microcomputers did not have the capacity or need for the elaborate operating systems that had originally been developed for mainframes and minis. Smaller operating systems were developed and often loaded from ROM and known as Monitors.

One notable early disk-based OS was CP/M which was supported on many early micro-computers and was largely cloned when MS-DOS was created. MS-DOS became wildly popular as the operating system chosen for the IBM PC.

The successive operating systems that came from MS-DOS made Microsoft one of the most profitable companies in the world with the development of Windows. The only other alternative throughout the 1980's was Mac OS which was tied intimately to the Apple McIntosh computer.

By the 1990s, the microcomputer had evolved to the point where it became increasingly desirable. Everyone wanted a home computer. Microsoft had already come out with Windows 95 and 98, but people longed for more power and more options. Microsoft's response to this change was the development of Windows NT which served as the basis for Microsoft's desktop operating system line that launched in 2001.

Apple was also rebuilding their own operating system on top of Unix core as Mac OS X also released in 2001 developing one of the business world's greatest rivalries.

Today, our operating systems usually have a graphical user interface (GUI) which uses a pointing device such as a mouse of stylus for input in addition to the keyboard. Older systems – and we mean REALLY OLD – use a command line interface asking for commands to be entered via the keyboard.

Both models are centered on a "shell" which accepts and processes commands from the user. The user may be asked to click on a button or type in a command upon an on-screen prompt.

By far, the most common operating system in use today is Windows XP, but Microsoft has just released their newest Windows project – Windows Vista. Linux is also another popular OS as is Unix. We'll explore them later on in the book, but each offers its own particular advantages and disadvantages.

Considering the boom of the technology market, it's really a surprise that there are so few operating systems in existence. There really isn't an easy explanation for this, but it is a reality. It would only seem logical that with all of the different computer manufacturers out there, there would be more of a choice for an OS than what there is. It is certainly another anomaly in the world of computer technology.

By Larry Miller

So what exactly do operating systems do? Since they really are the “brain” of the computer, they do quite a bit!

By Larry Miller

WHAT AN OPERATING SYSTEM DOES

As a user, you normally interact with the operating system through a set of commands. For example, the DOS operating system contains commands such as COPY and RENAME for copying files and changing the names of files, respectively.

The commands are accepted and executed by a part of the operating system called the command processor or command line interpreter. Graphical user interfaces allow you to enter commands by pointing and clicking at objects that appear on the screen.

But that really doesn't address the various ways that operating systems make your computer work easier and more efficiently. Their specific capacities are what make them help your computer operate as a user-friendly device. Let's look specifically at what an operating system does.

Process Management

Every program running on a computer whether it is a background service or an application is called a process. As long as von Neumann architecture is used to build a computer, only one process per CPU can be run at one time.

Older computer operating systems such as MS-DOS did not try to bypass this limit with the exception of interrupt processing and only one process could be run under them. Mainframe operating systems have had multi-tasking capabilities since the early 1960's. Modern operating systems enable concurrent execution of many processes at once via multi-tasking even with one CPU.

Process management is an operating system's way of dealing with running multiple processes at once. Since most computers contain one processor with one core, multi-tasking is done by simply switching processes quickly. Depending on the operating system, as more processes run, either each time slice will become smaller or there will be a longer delay before each process given a chance to run.

Process management involves computing and distributing CPU time as well as other resources. Most operating systems allow a process to be assigned a priority which affects its allocation of CPU time. Interactive operating systems also employ some level of feedback in which the task with which the user is working receives higher priority.

Interrupt driven processes will normally run at a very high priority. In many systems, there is a background process such as the System Idle Process in Windows which will run when no other process is waiting for the CPU.

It's tempting to think of a process as an application, but that gives an incomplete picture of how processes relate to the operating system and hardware. The application you see (word processor, spreadsheet or game) is, indeed, a process, but that application may cause several other processes to begin, for tasks like communications with other devices or other computers.

There are also numerous processes that run without giving you direct evidence that they ever exist. For example, Windows XP and UNIX can have dozens of background processes running to handle the network, memory management, disk management, virus checking and so on.

A process, then, is software that performs some action and can be controlled -- by a user, by other applications or by the operating system.

It is processes, rather than applications, that the operating system controls and schedules for execution by the CPU. In a single-tasking system, the schedule is straightforward. The operating system allows the application to begin running, suspending the execution only long enough to deal with interrupts and user input.

Interrupts are special signals sent by hardware or software to the CPU. It's as if some part of the computer suddenly raised its hand to ask for the CPU's attention in a lively meeting. Sometimes the operating system will schedule the priority of processes so that interrupts are masked -- that is, the operating system will ignore the interrupts from some sources so that a particular job can be finished as quickly as possible.

There are some interrupts such as those from error conditions or problems with memory that are so important that they can't be ignored. These non-maskable interrupts (NMIs) must be dealt with immediately, regardless of the other tasks at hand.

While interrupts add some complication to the execution of processes in a single-tasking system, the job of the operating system becomes much more complicated in a multi-tasking system. Now, the operating system must arrange the execution of applications so that you believe that there are several things happening at once.

This is complicated because the CPU can only do one thing at a time. In order to give the appearance of lots of things happening at the same time, the operating system has to switch between different processes thousands of times a second. Here's how it happens:

- A process occupies a certain amount of RAM. It also makes use of registers, stacks and queues within the CPU and operating-system memory space.

- When two processes are multi-tasking, the operating system allots a certain number of CPU execution cycles to one program.

- After that number of cycles, the operating system makes copies of all the registers, stacks and queues used by the processes and note the point at which the process paused in its execution.

- It then loads all the registers, stacks and queues used by the second process and allow it a certain number of CPU cycles.

- When those are complete, it makes copies of all the registers, stacks and queues used by the second program, and load the first program.

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