**ITSC(S) PRE-COURSE MATERIAL**

**LIST OF TOPICS**

|  |  |  |
| --- | --- | --- |
| **SL NO** | **TOPICS** | **PAGE NO** |
| 01 | Introduction to Computer hardware | 03 |
| 02 | Introduction to Computer networks | 21 |
| 03 | OSI Model | 33 |
| 04 | Ethernet | 39 |
| 05 | Operating system Introduction | 44 |
| 06 | Domain Administration | 59 |
| 07 | System Hardening | 63 |
| 08 | Data Structures | 71 |
| 09 | Object-oriented programming | 75 |
| 10 | Relational database Fundamentals | 83 |
| 11 | Software Engineering | 87 |
| 12 | Office Automation and Collaboration | 95 |
| 13 | Malwares, Virus, Trozan | 99 |
| 14 | Information security Basics | 104 |
| 15 | Tor (The Onion Router) | 114 |
| 16 | IT Asset Management | 118 |
| 17 | Question and Answers | 120 |

**Computer hardware**is the collection of physical elements that constitutes a computer system. Computer hardware refers to the physical parts or components of a computer such as monitor, keyboard, computer data storage, hard drive disk, mouse, system unit (graphic cards, sound cards, memory, motherboard and chips), etc. all of which are physical objects that can be touched.[1] In contrast, software is information stored by hardware. Software is any set of machine-readable instructions that directs a computer's processor to perform specific operations. A combination of hardware and software forms a usable computing system.

Computing hardware evolved from machines that needed separate manual action to perform each arithmetic operation, to punched card machines, and then to stored-program computers. The history of stored-program computers relates first to computer architecture, that is, the organization of the units to perform input and output, to store data and to operate as an integrated mechanism.

The Z3 by inventor Konrad Zuse from 1941 is regarded as the first working programmable, fully automatic modern computing machine. Thus, Zuse is often regarded as the inventor of the computer.[1][2][3][4]

Before the development of the general-purpose computer, most calculations were done by humans. Mechanical tools to help humans with digital calculations were then called "calculating machines", by proprietary names, or even as they are now, calculators. It was those humans who used the machines who were then called computers. Aside from written numerals, the first aids to computation were purely mechanical devices which required the operator to set up the initial values of an elementary arithmetic operation, then manipulate the device to obtain the result. A sophisticated (and comparatively recent) example is the slide rule, in which numbers are represented as lengths on a logarithmic scale and computation is performed by setting a cursor and aligning sliding scales, thus adding those lengths. Numbers could be represented in a continuous "analog" form, for instance a voltage or some other physical property was set to be proportional to the number. Analog computers, like those designed and built by Vannevar Bush before World War II were of this type. Numbers could be represented in the form of digits, automatically manipulated by a mechanical mechanism. Although this last approach required more complex mechanisms in many cases, it made for greater precision of results.

In the United States, the development of the computer was underpinned by massive government investment in the technology for military applications during WWII and then the Cold War. The latter superpower confrontation made it possible for local manufacturers to transform their machines into commercially viable products.[5] It was the same story in Europe, where adoption of computers began largely through proactive steps taken by national governments to stimulate development and deployment of the technology.[6]

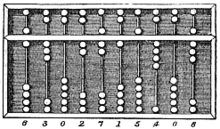
The invention of electronic amplifiers made calculating machines much faster than their mechanical or electromechanical predecessors. Vacuum tube (thermionic valve) amplifiers gave way to solid state transistors, and then rapidly to integrated circuits which continue to improve, placing millions of electrical switches (typically transistors) on a single elaborately manufactured piece of semi-conductor the size of a fingernail. By defeating the tyranny of numbers, integrated circuits made high-speed and low-cost digital computers a widespread commodity. There is an ongoing effort to make computer hardware faster, cheaper, and capable of storing more data. Computing hardware has become a platform for uses other than mere computation, such as process automation, electronic communications, equipment control, entertainment, education, etc. Each field in turn has imposed its own requirements on the hardware, which has evolved in response to those requirements, such as the role of the touch screen to create a more intuitive and natural user interface. As all computers rely on digital storage, and tend to be limited by the size and speed of memory, the history of computer data storage is tied to the development of computers.

Earliest true hardware

Devices have been used to aid computation for thousands of years, mostly using [one-to-one correspondence](http://en.wikipedia.org/wiki/One-to-one_correspondence)with [fingers](http://en.wikipedia.org/wiki/Finger_counting). The earliest counting device was probably a form of [tally stick](http://en.wikipedia.org/wiki/Tally_stick). Later record keeping aids throughout the [Fertile Crescent](http://en.wikipedia.org/wiki/Fertile_Crescent) included calculi (clay spheres, cones, etc.) which represented counts of items, probably livestock or grains, sealed in hollow unbaked clay containers.[[7]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-7)[[8]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-8) The use of [counting rods](http://en.wikipedia.org/wiki/Counting_rods) is one example.

The [abacus](http://en.wikipedia.org/wiki/Abacus) was early used for arithmetic tasks. What we now call the [Roman abacus](http://en.wikipedia.org/wiki/Roman_abacus) was used in [Babylonia](http://en.wikipedia.org/wiki/Babylonia) as early as 2400 BC. Since then, many other forms of reckoning boards or tables have been invented. In a medieval European [counting house](http://en.wikipedia.org/wiki/Counting_house), a checkered cloth would be placed on a table, and markers moved around on it according to certain rules, as an aid to calculating sums of money.

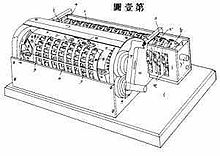
Several [analog computers](http://en.wikipedia.org/wiki/Analog_computer) were constructed in ancient and medieval times to perform astronomical calculations. These include the [Antikythera mechanism](http://en.wikipedia.org/wiki/Antikythera_mechanism) and the [astrolabe](http://en.wikipedia.org/wiki/Astrolabe) from [ancient Greece](http://en.wikipedia.org/wiki/Ancient_Greece) (c. 150–100 BC), which are generally regarded as the earliest known mechanical analog computers.[[9]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-9) [Hero of Alexandria](http://en.wikipedia.org/wiki/Hero_of_Alexandria) (c. 10–70 AD) made many complex mechanical devices including automata and a programmable cart.[[10]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-10) Other early versions of mechanical devices used to perform one or another type of calculations include the [planisphere](http://en.wikipedia.org/wiki/Planisphere) and other mechanical computing devices invented by [Abū Rayhān al-Bīrūnī](http://en.wikipedia.org/wiki/Ab%C5%AB_Rayh%C4%81n_al-B%C4%ABr%C5%ABn%C4%AB) (c. AD 1000); the [equatorium](http://en.wikipedia.org/wiki/Equatorium) and universal latitude-independent astrolabe by [Abū Ishāq Ibrāhīm al-Zarqālī](http://en.wikipedia.org/wiki/Ab%C5%AB_Ish%C4%81q_Ibr%C4%81h%C4%ABm_al-Zarq%C4%81l%C4%AB) (c. AD 1015); the astronomical analog computers of other medieval [Muslim astronomers](http://en.wikipedia.org/wiki/Islamic_astronomy) and engineers; and the [astronomical clock](http://en.wikipedia.org/wiki/Astronomical_clock) [tower](http://en.wikipedia.org/wiki/Clock_tower) of [Su Song](http://en.wikipedia.org/wiki/Su_Song) (c. AD 1090) during the [Song Dynasty](http://en.wikipedia.org/wiki/Song_Dynasty).

[](http://en.wikipedia.org/wiki/File:Abacus_6.png)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Abacus_6.png)

[Suanpan](http://en.wikipedia.org/wiki/Suanpan) (the number represented on this abacus is 6,302,715,408)

Scottish mathematician and physicist [John Napier](http://en.wikipedia.org/wiki/John_Napier) noted multiplication and division of numbers could be performed by addition and subtraction, respectively, of logarithms of those numbers. While producing the first logarithmic tables Napier needed to perform many multiplications, and it was at this point that he designed [Napier's bones](http://en.wikipedia.org/wiki/Napier%27s_bones), an abacus-like device used for multiplication and division.[[11]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-11) Since [real numbers](http://en.wikipedia.org/wiki/Real_number) can be represented as distances or intervals on a line, the [slide rule](http://en.wikipedia.org/wiki/Slide_rule) was invented in the 1620s to allow multiplication and division operations to be carried out significantly faster than was previously possible.[[12]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-12)Slide rules were used by generations of engineers and other mathematically involved professional workers, until the invention of the [pocket calculator](http://en.wikipedia.org/wiki/Pocket_calculator).[[13]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-13)

[](http://en.wikipedia.org/wiki/File:Patented_Yazu_Arithmometer.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Patented_Yazu_Arithmometer.jpg)

[Yazu](http://en.wikipedia.org/wiki/Ry%C5%8Dichi_Yazu) Arithmometer. Patented in Japan in 1903. Note the lever for turning the gears of the calculator.

[Wilhelm Schickard](http://en.wikipedia.org/wiki/Wilhelm_Schickard), a German [polymath](http://en.wikipedia.org/wiki/Polymath), designed a calculating clock in 1623. It made use of a single-tooth gear that was not an adequate solution for a general carry mechanism.[[14]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-14) A fire destroyed the machine during its construction in 1624 and Schickard abandoned the project. Two sketches of it were discovered in 1957, too late to have any impact on the development of mechanical calculators.[[15]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-15)

[](http://en.wikipedia.org/wiki/File:Lochkarte_Tanzorgel.jpg)In 1642, while still a teenager, [Blaise Pascal](http://en.wikipedia.org/wiki/Blaise_Pascal) started some pioneering work on calculating machines and after three years of effort and 50 prototypes[[16]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-16) he invented the [mechanical calculator](http://en.wikipedia.org/wiki/Mechanical_calculator).[[17]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-17)[[18]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-18) He built twenty of these machines (called [Pascal's Calculator](http://en.wikipedia.org/wiki/Pascal%27s_Calculator) or Pascaline) in the following ten years.[[19]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-19) Nine Pascalines have survived, most of which are on display in European museums.[[20]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-20)

[Gottfried Wilhelm von Leibniz](http://en.wikipedia.org/wiki/Gottfried_Leibniz) invented the [Stepped Reckoner](http://en.wikipedia.org/wiki/Stepped_Reckoner) and his [famous cylinders](http://en.wikipedia.org/wiki/Leibniz_wheel) around 1672 while adding direct multiplication and division to the Pascaline. Leibniz once said "It is unworthy of excellent men to lose hours like slaves in the labour of calculation which could safely be relegated to anyone else if machines were used."[[21]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-21)

Around 1820, [Charles Xavier Thomas de Colmar](http://en.wikipedia.org/wiki/Charles_Xavier_Thomas) created the first successful, mass-produced mechanical calculator, the Thomas [Arithmometer](http://en.wikipedia.org/wiki/Arithmometer), that could add, subtract, multiply, and divide.[[22]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-22) It was mainly based on Leibniz' work. Mechanical calculators, like the base-ten [addiator](http://en.wikipedia.org/wiki/Addiator), the [comptometer](http://en.wikipedia.org/wiki/Comptometer), the [Monroe](http://en.wikipedia.org/wiki/Monroe_calculator), the [Curta](http://en.wikipedia.org/wiki/Curta) and the[Addo-X](http://en.wikipedia.org/wiki/Addo-X) remained in use until the 1970s. Leibniz also described the [binary numeral system](http://en.wikipedia.org/wiki/Binary_numeral_system),[[23]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-23) a central ingredient of all modern computers. However, up to the 1940s, many subsequent designs (including [Charles Babbage](http://en.wikipedia.org/wiki/Charles_Babbage)'s machines of the 1822 and even [ENIAC](http://en.wikipedia.org/wiki/ENIAC) of 1945) were based on the decimal system;[[24]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-24) ENIAC's ring counters emulated the operation of the digit wheels of a mechanical adding machine.

In Japan, [Ryōichi Yazu](http://en.wikipedia.org/wiki/Ry%C5%8Dichi_Yazu) patented a mechanical calculator called the Yazu Arithmometer in 1903. It consisted of a single cylinder and 22 gears, and employed the mixed base-2 and base-5 number system familiar to users of the[soroban](http://en.wikipedia.org/wiki/Soroban) (Japanese abacus). Carry and end of calculation were determined automatically.[[25]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-25) More than 200 units were sold, mainly to government agencies such as the Ministry of War and agricultural experiment stations.[[26]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-26)[[27]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-27)

1801: punched card technology

In 1801, [Joseph-Marie Jacquard](http://en.wikipedia.org/wiki/Joseph_Marie_Jacquard) developed [a loom](http://en.wikipedia.org/wiki/Jacquard_loom) in which the pattern being woven was controlled by [punched cards](http://en.wikipedia.org/wiki/Punched_cards). The series of cards could be changed without changing the mechanical design of the loom. This was a landmark achievement in programmability. His machine was an improvement over similar weaving looms. Punch cards were preceded by punch bands, as in the machine proposed by [Basile Bouchon](http://en.wikipedia.org/wiki/Basile_Bouchon). These bands would inspire information recording for automatic pianos and more recently NC machine-tools.

In 1833, [Charles Babbage](http://en.wikipedia.org/wiki/Charles_Babbage) moved on from developing his [difference engine](http://en.wikipedia.org/wiki/Difference_engine) (for navigational calculations) to a general purpose design, the Analytical Engine, which drew directly on Jacquard's punched cards for its program storage.[[28]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-28) In 1837, Babbage described his[analytical engine](http://en.wikipedia.org/wiki/Analytical_engine). It was a general-purpose programmable computer, employing punch cards for input and a steam engine for power, using the positions of gears and shafts to represent numbers.[[29]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-29) His initial idea was to use punch-cards to control a machine that could calculate and print logarithmic tables with huge precision (a special purpose machine). Babbage's idea soon developed into a general-purpose programmable computer. While his design was sound and the plans were probably correct, or at least [debuggable](http://en.wikipedia.org/wiki/Debug), the project was slowed by various problems including disputes with the chief machinist building parts for it. Babbage was a difficult man to work with and argued with everyone. All the parts for his machine had to be made by hand. Small errors in each item might sometimes sum to cause large discrepancies. In a machine with thousands of parts, which required these parts to be much better than the usual tolerances needed at the time, this was a major problem. The project dissolved in disputes with the artisan who built parts and ended with the decision of the British Government to cease funding. [Ada Lovelace](http://en.wikipedia.org/wiki/Ada_Lovelace), [Lord Byron](http://en.wikipedia.org/wiki/George_Gordon_Byron,_6th_Baron_Byron)'s daughter, translated and [added notes](http://en.wikipedia.org/wiki/Ada_Byron%27s_notes_on_the_analytical_engine) to the "*Sketch of the Analytical Engine*" by [Federico Luigi, Conte Menabrea](http://en.wikipedia.org/wiki/Federico_Luigi,_Conte_Menabrea). This appears to be the first published description of programming.[[30]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-30)

A reconstruction of the [Difference Engine](http://en.wikipedia.org/wiki/Difference_Engine) II, an earlier, more limited design, has been operational since 1991 at the [London Science Museum](http://en.wikipedia.org/wiki/London_Science_Museum). With a few trivial changes, it works exactly as Babbage designed it and shows that Babbage's design ideas were correct, merely too far ahead of his time. The museum used computer-controlled machine tools to construct the necessary parts, using tolerances a good machinist of the period would have been able to achieve. Babbage's failure to complete the analytical engine can be chiefly attributed to difficulties not only of politics and financing, but also to his desire to develop an increasingly sophisticated computer and to move ahead faster than anyone else could follow.

A machine based on Babbage's difference engine was built in 1843 by [Per Georg Scheutz](http://en.wikipedia.org/wiki/Per_Georg_Scheutz) and his son Edward. An improved Scheutzian calculation engine was sold to the British government and a later model was sold to the American government and these were used successfully in the production of logarithmic tables.[[31]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-31)[[32]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-32)

Following Babbage, although unaware of his earlier work, was [Percy Ludgate](http://en.wikipedia.org/wiki/Percy_Ludgate), an accountant from Dublin, Ireland. He independently designed a programmable mechanical computer, which he described in a work that was published in 1909.

[](http://en.wikipedia.org/wiki/File:Early_SSA_accounting_operations.jpg)1880s: punched card data storage

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Early_SSA_accounting_operations.jpg)

IBM punched card Accounting Machines at the U.S. Social Security Administration in 1936.

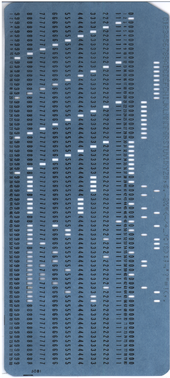
[](http://en.wikipedia.org/wiki/File:Ibm407_tabulator_1961_01.redstone.jpg)In the late 1880s, the American [Herman Hollerith](http://en.wikipedia.org/wiki/Herman_Hollerith) invented data storage on a medium that could then be read by a machine. Prior uses of machine readable media had been for control ([automatons](http://en.wikipedia.org/wiki/Automaton)such as [piano rolls](http://en.wikipedia.org/wiki/Piano_roll) or [looms](http://en.wikipedia.org/wiki/Jacquard_loom)), not data. "After some initial trials with paper tape, he settled on [punched cards](http://en.wikipedia.org/wiki/Punched_card)..."[[33]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-33) Hollerith came to use punched cards after observing how [railroad conductors](http://en.wikipedia.org/wiki/Railroad_conductor) encoded personal characteristics of each passenger with punches on their tickets. To process these punched cards he invented the [tabulator](http://en.wikipedia.org/wiki/Tabulating_machine), and the [key punch](http://en.wikipedia.org/wiki/Key_punch) machine. These three inventions were the foundation of the modern information processing industry. His machines used mechanical [relays](http://en.wikipedia.org/wiki/Relay) (and [solenoids](http://en.wikipedia.org/wiki/Solenoid)) to increment[mechanical counters](http://en.wikipedia.org/wiki/Mechanical_counter). Hollerith's method was used in the [1890 United States Census](http://en.wikipedia.org/wiki/1890_United_States_Census) and the completed results were "... finished months ahead of schedule and far under budget".[[34]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-34) Indeed, the census was processed years faster than the prior census had been. Hollerith's company eventually became the core of [IBM](http://en.wikipedia.org/wiki/International_Business_Machines). IBM developed punch card technology into a [](http://en.wikipedia.org/wiki/File:Curta01.JPG)powerful tool for business data-processing and produced an extensive line of [unit record equipment](http://en.wikipedia.org/wiki/Unit_record_equipment). By 1950, the IBM card had become ubiquitous in industry and government. The warning printed on most cards intended for circulation as documents (checks, for example), "Do not fold, [spindle](http://en.wikipedia.org/wiki/Spindle_(stationery)) or mutilate," became a catch phrase for the post-World War II era.[[35]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-35)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Ibm407_tabulator_1961_01.redstone.jpg)

Punch card Tabulator, 1961

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Blue-punch-card-front.png)

[Leslie Comrie](http://en.wikipedia.org/wiki/Leslie_Comrie)'s articles on punched card methods and [W.J. Eckert](http://en.wikipedia.org/wiki/W.J._Eckert)'s publication of *Punched Card Methods in Scientific Computation* in 1940, described punch card techniques sufficiently advanced to solve some differential equations[[36]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-36) or perform multiplication and division using floating point representations, all on punched cards and [unit record machines](http://en.wikipedia.org/wiki/Unit_record_equipment). Those same machines had been used during World War II for cryptographic statistical processing. In the image of the tabulator (see left), note the[control panel](http://en.wikipedia.org/wiki/Plugboard), which is visible on the right side of the tabulator. A row of [toggle switches](http://en.wikipedia.org/wiki/Toggle_switch) is above the control panel. The [Thomas J. Watson Astronomical Computing Bureau](http://www.columbia.edu/acis/history/), [Columbia University](http://en.wikipedia.org/wiki/Columbia_University) performed astronomical calculations representing the state of the art in [computing](http://en.wikipedia.org/wiki/Computing).[[37]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-37)

[](http://en.wikipedia.org/wiki/File:Blue-punch-card-front.png)[Computer programming in the punch card era](http://en.wikipedia.org/wiki/Computer_programming_in_the_punch_card_era) was centered in the "computer center". Computer users, for example science and engineering students at universities, would submit their programming assignments to their local computer center in the form of a deck of punched cards, one card per program line. They then had to wait for the program to be read in, queued for processing, compiled, and executed. In due course, a printout of any results, marked with the submitter's identification, would be placed in an output tray, typically in the computer center lobby. In many cases these results would be only a series of error messages, requiring yet another [edit-punch-compile-run cycle](http://en.wikipedia.org/wiki/Code_and_fix).[[38]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-38)Punched cards are still used and manufactured to this day, and their distinctive dimensions (and 80-column capacity) can still be recognized in forms, records, and programs around the world. They are the size of American paper currency in Hollerith's time, a choice he made because there was already equipment available to handle bills.

Desktop calculators

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Curta01.JPG)

The [Curta](http://en.wikipedia.org/wiki/Curta) calculator can also do multiplication and division.

By the 20th century, earlier mechanical calculators, cash registers, accounting machines, and so on were redesigned to use electric motors, with gear position as the representation for the state of a variable. The word "computer" was a job title assigned to people who used these calculators to perform mathematical calculations. By the 1920s [Lewis Fry Richardson](http://en.wikipedia.org/wiki/Lewis_Fry_Richardson)'s interest in weather prediction led him to propose [human computers](http://en.wikipedia.org/wiki/Human_computer) and [numerical analysis](http://en.wikipedia.org/wiki/Numerical_analysis) to model the weather; to this day, the most powerful computers on [Earth](http://en.wikipedia.org/wiki/Earth) are needed to adequately model its weather using the [Navier–Stokes equations](http://en.wikipedia.org/wiki/Navier%E2%80%93Stokes_equations).[[39]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-39)

Companies like [Friden](http://en.wikipedia.org/wiki/Friden,_Inc.), [Marchant Calculator](http://en.wikipedia.org/wiki/Marchant_Calculator) and [Monroe](http://en.wikipedia.org/wiki/Monroe_Calculator_Company) made desktop mechanical [calculators](http://www.oldcalculatormuseum.com/fridenstw.html) from the 1930s that could add, subtract, multiply and divide. During the [Manhattan project](http://en.wikipedia.org/wiki/Manhattan_project), future Nobel laureate [Richard Feynman](http://en.wikipedia.org/wiki/Richard_Feynman) was the supervisor of human computers who understood the use of [differential equations](http://en.wikipedia.org/wiki/Differential_equations) which were being solved for the war effort.

In 1948, the [Curta](http://en.wikipedia.org/wiki/Curta_calculator) was introduced. This was a small, portable, mechanical calculator that was about the size of a [pepper grinder](http://en.wikipedia.org/wiki/Pepper_grinder). Over time, during the 1950s and 1960s a variety of different brands of mechanical calculators appeared on the market. The first all-electronic desktop calculator was the British [ANITA Mk.VII](http://en.wikipedia.org/wiki/Sumlock_ANITA_calculator), which used a [Nixie tube](http://en.wikipedia.org/wiki/Nixie_tube) display and 177 subminiature [thyratron](http://en.wikipedia.org/wiki/Thyratron) tubes. In June 1963, Friden introduced the four-function EC-130. It had an all-transistor design, 13-digit capacity on a 5-inch (130 mm) [CRT](http://en.wikipedia.org/wiki/Cathode_ray_tube), and introduced [Reverse Polish notation](http://en.wikipedia.org/wiki/Reverse_Polish_notation) (RPN) to the calculator market at a price of $2200. The EC-132 model added square root and reciprocal functions. In 1965, [Wang Laboratories](http://en.wikipedia.org/wiki/Wang_Laboratories) produced the LOCI-2, a 10-digit transistorized desktop calculator that used a Nixie tube display and could compute [logarithms](http://en.wikipedia.org/wiki/Logarithm).

In the early days of binary vacuum-tube computers, their reliability was poor enough to justify marketing a mechanical octal version ("Binary Octal") of the Marchant desktop calculator. It was intended to check and verify calculation results of such computers.

Advanced analog computers

[](http://en.wikipedia.org/wiki/File:Cambridge_differential_analyser.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Cambridge_differential_analyser.jpg)

Cambridge differential analyzer, 1938

Before [World War II](http://en.wikipedia.org/wiki/World_War_II), mechanical and electrical [analog computers](http://en.wikipedia.org/wiki/Analog_computer)were considered the "state of the art", and many thought they were the future of computing. Analog computers take advantage of the strong similarities between the mathematics of small-scale properties—the position and motion of wheels or the voltage and current of electronic components—and the mathematics of other physical phenomena, for example, ballistic trajectories, inertia, resonance, energy transfer, momentum, and so forth. They model physical phenomena with electrical [voltages](http://en.wikipedia.org/wiki/Voltage) and [currents](http://en.wikipedia.org/wiki/Electric_current)[[40]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-40) as the analog quantities.

Centrally, these analog systems work by creating electrical '[analogs](http://en.wikipedia.org/wiki/Analogy)' of other systems, allowing users to predict behavior of the systems of interest by observing the electrical analogs.[[41]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-41) The most useful of the analogies was the way the small-scale behavior could be represented with integral and differential equations, and could be thus used to solve those equations. An ingenious example of such a machine, using water as the analog quantity, was the [water integrator](http://en.wikipedia.org/wiki/Water_integrator)built in 1928; an electrical example is the [Mallock machine](http://en.wikipedia.org/wiki/Mallock_machine) built in 1941. A [planimeter](http://en.wikipedia.org/wiki/Planimeter) is a device which does integrals, using [distance](http://en.wikipedia.org/wiki/Distance) as the analog quantity. Unlike modern digital computers, analog computers are not very flexible, and need to be rewired manually to switch them from working on one problem to another. Analog computers had an advantage over early digital computers in that they could be used to solve complex problems using behavioral analogues while the earliest attempts at digital computers were quite limited.

Some of the most widely deployed analog computers included devices for aiming weapons, such as the [Norden bombsight](http://en.wikipedia.org/wiki/Norden_bombsight),[[42]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-42) and [fire-control systems](http://en.wikipedia.org/wiki/Fire-control_system),[[43]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-43) such as [Arthur Pollen](http://en.wikipedia.org/wiki/Arthur_Pollen)'s Argo system for naval vessels. Some stayed in use for decades after World War II; the [Mark I Fire Control Computer](http://en.wikipedia.org/wiki/Mark_I_Fire_Control_Computer) was deployed by the [United States Navy](http://en.wikipedia.org/wiki/United_States_Navy)on a variety of ships from [destroyers](http://en.wikipedia.org/wiki/Destroyer) to [battleships](http://en.wikipedia.org/wiki/Battleship). Other analog computers included the [Heathkit](http://en.wikipedia.org/wiki/Heathkit) EC-1, and the hydraulic [MONIAC Computer](http://en.wikipedia.org/wiki/MONIAC_Computer) which modeled econometric flows.[[44]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-44)

The art of mechanical analog computing reached its zenith with the [differential analyzer](http://en.wikipedia.org/wiki/Differential_analyzer),[[45]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-45) built by H. L. Hazen and [Vannevar Bush](http://en.wikipedia.org/wiki/Vannevar_Bush) at [MIT](http://en.wikipedia.org/wiki/MIT) starting in 1927, which in turn built on the mechanical integrators invented in 1876 by[James Thomson](http://en.wikipedia.org/wiki/James_Thomson_(engineer)) and the torque amplifiers invented by H. W. Nieman. A dozen of these devices were built before their obsolescence was obvious; the most powerful was constructed at the [University of Pennsylvania](http://en.wikipedia.org/wiki/University_of_Pennsylvania)'s[Moore School of Electrical Engineering](http://en.wikipedia.org/wiki/Moore_School_of_Electrical_Engineering), where the [ENIAC](http://en.wikipedia.org/wiki/ENIAC) was built. Digital electronic computers like the ENIAC spelled the end for most analog computing machines, but hybrid analog computers, controlled by digital electronics, remained in substantial use into the 1950s and 1960s, and later in some specialized applications.

Early electronic digital computation

[](http://en.wikipedia.org/wiki/File:Punched_tape_puncher.JPG)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Punched_tape_puncher.JPG)

Friden paper tape punch. [Punched tape](http://en.wikipedia.org/wiki/Punched_tape)programs would be much longer than the short fragment of yellow paper tape shown.

The era of modern computing began with a flurry of development before and during World War II.

At first electromechanical components such as relays were employed. [George Stibitz](http://en.wikipedia.org/wiki/George_Stibitz) is internationally recognized as one of the fathers of the modern digital computer. While working at Bell Labs in November 1937, Stibitz invented and built a relay-based calculator that he dubbed the "Model K" (for "kitchen table", on which he had assembled it), which was the first to calculate using [binary form](http://en.wikipedia.org/wiki/Binary_numeral_system).[[46]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-46)

However, [electronic circuit](http://en.wikipedia.org/wiki/Electronic_circuit) elements replaced their mechanical and electromechanical equivalents, and digital calculations replaced analog calculations. Machines such as the [Z3](http://en.wikipedia.org/wiki/Z3_(computer)), the [Atanasoff–Berry Computer](http://en.wikipedia.org/wiki/Atanasoff%E2%80%93Berry_Computer), the [Colossus computers](http://en.wikipedia.org/wiki/Colossus_computer), and the [ENIAC](http://en.wikipedia.org/wiki/ENIAC) were built by hand using circuits containing relays or valves (vacuum tubes), and often used [punched cards](http://en.wikipedia.org/wiki/Punched_card) or [punched paper tape](http://en.wikipedia.org/wiki/Punched_tape) for input and as the main (non-volatile) storage medium. Defining a single point in the series as the "first computer" misses many subtleties (see the table "Defining characteristics of some early digital computers of the 1940s" below).

**Turing**

[Alan Turing](http://en.wikipedia.org/wiki/Alan_Turing)'s 1936 paper[[47]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-47) proved enormously influential in computing and [computer science](http://en.wikipedia.org/wiki/Computer_science) in two ways. Its main purpose was to prove that there were problems (namely the [halting problem](http://en.wikipedia.org/wiki/Halting_problem)) that could not be solved by any sequential process. In doing so, Turing provided a definition of a universal computer which executes a program stored on tape. This construct came to be called a [Turing machine](http://en.wikipedia.org/wiki/Turing_machine).[[48]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-48) Except for the limitations imposed by their finite memory stores, modern computers are said to be [Turing-complete](http://en.wikipedia.org/wiki/Turing-complete), which is to say, they have [algorithm](http://en.wikipedia.org/wiki/Algorithm) execution capability equivalent to a [universal Turing machine](http://en.wikipedia.org/wiki/Universal_Turing_machine).

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Largetape.jpg)

[](http://en.wikipedia.org/wiki/File:Largetape.jpg)Half-inch (12.7 mm) [magnetic tape](http://en.wikipedia.org/wiki/Magnetic_tape), originally written with [7 tracks](http://en.wikipedia.org/wiki/IBM_7_track) and later [9-tracks](http://en.wikipedia.org/wiki/9_track_tape).

For a computing machine to be a practical general-purpose computer, there must be some convenient read-write mechanism, punched tape, for example. With knowledge of Alan Turing's theoretical 'universal computing machine' [John von Neumann](http://en.wikipedia.org/wiki/John_von_Neumann)defined an architecture which uses the same [memory](http://en.wikipedia.org/wiki/Computer_memory) both to store programs and data: virtually all contemporary computers use this architecture (or some variant). While it is theoretically possible to implement a full computer entirely mechanically (as Babbage's design showed), electronics made possible the speed and later the miniaturization that characterize modern computers.

There were three parallel streams of computer development in the World War II era; the first stream largely ignored, and the second stream deliberately kept secret. The first was the German work of[Konrad Zuse](http://en.wikipedia.org/wiki/Konrad_Zuse). The second was the secret development of the Colossus computers in the UK. Neither of these had much influence on the various computing projects in the United States, but some of the technology led, via Turing and others, to the first commercial electronic computer. The third stream of computer development was Eckert and Mauchly's ENIAC and EDVAC, which was widely publicized.[[49]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-49)[[50]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-50)

**Zuse**

[](http://en.wikipedia.org/wiki/File:Zuse_Z1.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Zuse_Z1.jpg)

A reproduction of Zuse's Z1 computer

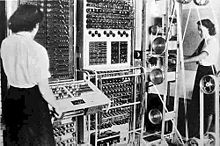
Working in isolation in Germany, [Konrad Zuse](http://en.wikipedia.org/wiki/Konrad_Zuse) started construction in 1936 of his first Z-series calculators featuring memory and (initially limited) programmability. Zuse's purely mechanical, but already binary [Z1](http://en.wikipedia.org/wiki/Z1_(computer)), finished in 1938, never worked reliably due to problems with the precision of parts.

Zuse's later machine, the [Z3](http://en.wikipedia.org/wiki/Z3_(computer)),[[51]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-51) was finished in 1941. It was based on telephone relays and did work satisfactorily. The Z3 thus became the world's first functional program-controlled, all-purpose, digital computer. In many ways it was quite similar to modern machines, pioneering numerous advances, such as [floating point numbers](http://en.wikipedia.org/wiki/Floating_point_number). Replacement of the hard-to-implement decimal system (used in [Charles Babbage](http://en.wikipedia.org/wiki/Charles_Babbage)'s earlier design) by the simpler [binary](http://en.wikipedia.org/wiki/Binary_numeral_system) system meant that Zuse's machines were easier to build and potentially more reliable, given the technologies available at that time.

Programs were fed into Z3 on punched films. Conditional jumps were missing, but since the 1990s it has been proved theoretically that Z3 was still a [universal computer](http://en.wikipedia.org/wiki/Turing_machine) (as always, ignoring physical storage limitations). In two 1936 [patent](http://en.wikipedia.org/wiki/Patent) applications, Konrad Zuse also anticipated that machine instructions could be stored in the same storage used for data—the key insight of what became known as the [von Neumann architecture](http://en.wikipedia.org/wiki/Von_Neumann_architecture), first implemented in the British [SSEM](http://en.wikipedia.org/wiki/Manchester_Small-Scale_Experimental_Machine) of 1948.[[52]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-52) Zuse also claimed to have designed the first higher-level[programming language](http://en.wikipedia.org/wiki/Programming_language), which he named [Plankalkül](http://en.wikipedia.org/wiki/Plankalk%C3%BCl), in 1945 (published in 1948) although it was implemented for the first time in 1998 and then again in 2000 by a team around [Raúl Rojas](http://en.wikipedia.org/wiki/Ra%C3%BAl_Rojas) at the [Free University of Berlin](http://en.wikipedia.org/wiki/Free_University_of_Berlin).

Zuse suffered setbacks during World War II when some of his machines were destroyed in the course of [Allied](http://en.wikipedia.org/wiki/Allies_of_World_War_II)bombing campaigns. Apparently his work remained largely unknown to engineers in the UK and US until much later, although at least IBM was aware of it as it financed his post-war startup company in 1946 in return for an option on Zuse's patents.

**Colossus**

[](http://en.wikipedia.org/wiki/File:Colossus.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Colossus.jpg)

Colossus was used to break German ciphers during World War II.

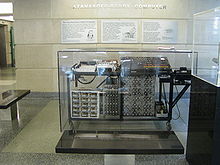
During World War II, the British at [Bletchley Park](http://en.wikipedia.org/wiki/Bletchley_Park) (40 miles north of London) achieved a number of successes at breaking encrypted German military communications. The German encryption machine,[Enigma](http://en.wikipedia.org/wiki/Enigma_(machine)), was attacked with the help of electro-mechanical machines called [*bombes*](http://en.wikipedia.org/wiki/Bombe). The bombe, designed by [Alan Turing](http://en.wikipedia.org/wiki/Alan_Turing) and [Gordon Welchman](http://en.wikipedia.org/wiki/Gordon_Welchman), after the Polish cryptographic [*bomba*](http://en.wikipedia.org/wiki/Bomba_(cryptography)) by [Marian Rejewski](http://en.wikipedia.org/wiki/Marian_Rejewski) (1938), came into productive use in 1941.[[53]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-53) They ruled out possible Enigma settings by performing chains of logical deductions implemented electrically. Most possibilities led to a contradiction, and the few remaining could be tested by hand.

The Germans also developed a series of teleprinter encryption systems, quite different from Enigma. The [Lorenz SZ 40/42](http://en.wikipedia.org/wiki/Lorenz_SZ_40/42)machine was used for high-level Army communications, termed "Tunny" by the British. The first intercepts of Lorenz messages began in 1941. As part of an attack on Tunny, [Max Newman](http://en.wikipedia.org/wiki/Max_Newman) and his colleagues helped specify the Colossus.[[54]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-54) The Mk I Colossus was built between March and December 1943 by [Tommy Flowers](http://en.wikipedia.org/wiki/Tommy_Flowers) and his colleagues at the [Post Office Research Station](http://en.wikipedia.org/wiki/Post_Office_Research_Station) at [Dollis Hill](http://en.wikipedia.org/wiki/Dollis_Hill) in London and then shipped to [Bletchley Park](http://en.wikipedia.org/wiki/Bletchley_Park) in January 1944.

Colossus was the world's first electronic programmable computing device. It used a large number of valves (vacuum tubes). It had paper-tape input and was capable of being configured to perform a variety of [boolean logical](http://en.wikipedia.org/wiki/Boolean_logic) operations on its data, but it was not [Turing-complete](http://en.wikipedia.org/wiki/Turing-complete). Nine Mk II Colossi were built (The Mk I was converted to a Mk II making ten machines in total). Details of their existence, design, and use were kept secret well into the 1970s. [Winston Churchill](http://en.wikipedia.org/wiki/Winston_Churchill) personally issued an order for their destruction into pieces no larger than a man's hand, to keep secret that the British were capable of cracking Lorenz during the oncoming cold war. Two of the machines were transferred to the newly formed [GCHQ](http://en.wikipedia.org/wiki/GCHQ) and the others were destroyed. As a result the machines were not included in many histories of computing. A reconstructed working copy of one of the Colossus machines is now on display at Bletchley Park.

**American developments**

In 1937, [Claude Shannon](http://en.wikipedia.org/wiki/Claude_Shannon) showed there is a [one-to-one correspondence](http://en.wikipedia.org/wiki/One-to-one_correspondence) between the concepts of [Boolean logic](http://en.wikipedia.org/wiki/Boolean_logic)and certain electrical circuits, now called [logic gates](http://en.wikipedia.org/wiki/Logic_gate), which are now ubiquitous in digital computers.[[55]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-55) In his master's thesis[[56]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-56) at [MIT](http://en.wikipedia.org/wiki/Massachusetts_Institute_of_Technology), for the first time in history, Shannon showed that electronic relays and switches can realize the [expressions](http://en.wikipedia.org/wiki/Expression_(mathematics)) of [Boolean algebra](http://en.wikipedia.org/wiki/Boolean_algebra_(logic)). Entitled [*A Symbolic Analysis of Relay and Switching Circuits*](http://en.wikipedia.org/wiki/A_Symbolic_Analysis_of_Relay_and_Switching_Circuits), Shannon's thesis essentially founded practical [digital circuit](http://en.wikipedia.org/wiki/Digital_circuit) design. George Stibitz completed a relay-based computer he dubbed the "Model K" at [Bell Labs](http://en.wikipedia.org/wiki/Bell_Labs) in November 1937. Bell Labs authorized a full research program in late 1938 with Stibitz at the helm. Their [*Complex Number Calculator*](http://en.wikipedia.org/wiki/Complex_Number_Calculator),[[57]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-57) completed January 8, 1940, was able to calculate [complex numbers](http://en.wikipedia.org/wiki/Complex_numbers). In a demonstration to the [American Mathematical Society](http://en.wikipedia.org/wiki/American_Mathematical_Society) conference at[Dartmouth College](http://en.wikipedia.org/wiki/Dartmouth_College) on September 11, 1940, Stibitz was able to send the Complex Number Calculator remote commands over telephone lines by a [teletype](http://en.wikipedia.org/wiki/Teletype). It was the first computing machine ever used remotely, in this case over a phone line. Some participants in the conference who witnessed the demonstration were [John von Neumann](http://en.wikipedia.org/wiki/John_von_Neumann), John Mauchly, and [Norbert Wiener](http://en.wikipedia.org/wiki/Norbert_Wiener), who wrote about it in their memoirs.

[](http://en.wikipedia.org/wiki/File:Atanasoff-Berry_Computer_at_Durhum_Center.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Atanasoff-Berry_Computer_at_Durhum_Center.jpg)

[Atanasoff–Berry Computer](http://en.wikipedia.org/wiki/Atanasoff%E2%80%93Berry_Computer) replica at 1st floor of Durham Center, [Iowa State University](http://en.wikipedia.org/wiki/Iowa_State_University)

In 1939, John Vincent Atanasoff and Clifford E. Berry of Iowa State University developed the [Atanasoff–Berry Computer](http://en.wikipedia.org/wiki/Atanasoff%E2%80%93Berry_Computer) (ABC),[[58]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-58) The Atanasoff-Berry Computer was the world's first electronic digital computer.[[59]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-59) The design used over 300 vacuum tubes and employed capacitors fixed in a mechanically rotating drum for memory. Though the ABC machine was not programmable, it was the first to use electronic tubes in an adder. ENIAC co-inventor John Mauchly examined the ABC in June 1941, and its influence on the design of the later ENIAC machine is a matter of contention among computer historians. The ABC was largely forgotten until it became the focus of the lawsuit [*Honeywell v. Sperry Rand*](http://en.wikipedia.org/wiki/Honeywell_v._Sperry_Rand), the ruling of which invalidated the ENIAC patent (and several others) as, among many reasons, having been anticipated by Atanasoff's work.

In 1939, development began at IBM's Endicott laboratories on the [Harvard Mark I](http://en.wikipedia.org/wiki/Harvard_Mark_I). Known officially as the Automatic Sequence Controlled Calculator,[[60]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-60) the Mark I was a general purpose electro-mechanical computer built with IBM financing and with assistance from IBM personnel, under the direction of Harvard mathematician[Howard Aiken](http://en.wikipedia.org/wiki/Howard_Aiken). Its design was influenced by Babbage's Analytical Engine, using decimal arithmetic and storage wheels and rotary switches in addition to electromagnetic relays. It was programmable via punched paper tape, and contained several calculation units working in parallel. Later versions contained several paper tape readers and the machine could switch between readers based on a condition. Nevertheless, the machine was not quite Turing-complete. The Mark I was moved to [Harvard University](http://en.wikipedia.org/wiki/Harvard_University) and began operation in May 1944.

**ENIAC**

[](http://en.wikipedia.org/wiki/File:Eniac.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Eniac.jpg)

[ENIAC](http://en.wikipedia.org/wiki/ENIAC) performed ballistics trajectory calculations with 160 kW of power

The US-built ENIAC (Electronic Numerical Integrator and Computer) was the first electronic general-purpose computer. It combined, for the first time, the high speed of electronics with the ability to be programmed for many complex problems. It could add or subtract 5000 times a second, a thousand times faster than any other machine. It also had modules to multiply, divide, and square root. High speed memory was limited to 20 words (about 80 bytes). Built under the direction of [John Mauchly](http://en.wikipedia.org/wiki/John_Mauchly) and [J. Presper Eckert](http://en.wikipedia.org/wiki/J._Presper_Eckert) at the University of Pennsylvania, ENIAC's development and construction lasted from 1943 to full operation at the end of 1945. The machine was huge, weighing 30 tons, using 200 kilowatts of electric power and contained over 18,000 vacuum tubes, 1,500 relays, and hundreds of thousands of resistors, capacitors, and inductors.[[61]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-Eniac-61)One of the major engineering feats was to minimize tube burnout, which was a common problem at that time. The machine was in almost constant use for the next ten years.

ENIAC was unambiguously a Turing-complete device. It could compute any problem (that would fit in memory). A "program" on the ENIAC, however, was defined by the states of its patch cables and switches, a far cry from the[stored program](http://en.wikipedia.org/wiki/Stored_program) electronic machines that came later. Once a program was written, it had to be mechanically set into the machine. [Six women did most of the programming of ENIAC.](http://en.wikipedia.org/wiki/ENIAC#Programming) (Improvements completed in 1948 made it possible to execute stored programs set in function table memory, which made programming less a "one-off" effort, and more systematic).

**Manchester "baby**

[](http://en.wikipedia.org/wiki/File:SSEM_Manchester_museum.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:SSEM_Manchester_museum.jpg)

Replica of the Small-Scale Experimental Machine (SSEM) at the [Museum of Science and Industry](http://en.wikipedia.org/wiki/Museum_of_Science_and_Industry_(Manchester)) in [Castlefield](http://en.wikipedia.org/wiki/Castlefield), [Manchester](http://en.wikipedia.org/wiki/Manchester)

The Manchester Small-Scale Experimental Machine, nicknamed*Baby*, was the world's first [stored-program computer](http://en.wikipedia.org/wiki/Stored-program_computer). It was built at the [Victoria University of Manchester](http://en.wikipedia.org/wiki/Victoria_University_of_Manchester) by [Frederic C. Williams](http://en.wikipedia.org/wiki/Frederic_Calland_Williams), [Tom Kilburn](http://en.wikipedia.org/wiki/Tom_Kilburn) and Geoff Tootill, and ran its first program on 21 June 1948.[[62]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-62)

The machine was not intended to be a practical computer but was instead designed as a [testbed](http://en.wikipedia.org/wiki/Testbed) for the [Williams tube](http://en.wikipedia.org/wiki/Williams_tube), an early form of computer memory. Although considered "small and primitive" by the standards of its time, it was the first working machine to contain all of the elements essential to a modern electronic computer.[[63]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-EarlyComputers-63) As soon as the SSEM had demonstrated the feasibility of its design, a project was initiated at the university to develop it into a more usable computer, the [Manchester Mark 1](http://en.wikipedia.org/wiki/Manchester_Mark_1). The Mark 1 in turn quickly became the prototype for the [Ferranti Mark 1](http://en.wikipedia.org/wiki/Ferranti_Mark_1), the world's first commercially available general-purpose computer.[[64]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-NapperMK1-64)

The SSEM had a 32-[bit](http://en.wikipedia.org/wiki/Bit) [word](http://en.wikipedia.org/wiki/Word_(data_type)) length and a [memory](http://en.wikipedia.org/wiki/Computer_memory) of 32 words. As it was designed to be the simplest possible stored-program computer, the only arithmetic operations implemented in hardware were [subtraction](http://en.wikipedia.org/wiki/Subtraction) and[negation](http://en.wikipedia.org/wiki/Negation); other arithmetic operations were implemented in software. The first of three programs written for the machine found the highest [proper divisor](http://en.wikipedia.org/wiki/Proper_divisor) of 218 (262,144), a calculation that was known would take a long time to run—and so prove the computer's reliability—by testing every integer from 218 − 1 downwards, as division was implemented by repeated subtraction of the divisor. The program consisted of 17 instructions and ran for 52 minutes before reaching the correct answer of 131,072, after the SSEM had performed 3.5 million operations (for an effective CPU speed of 1.1 [kIPS](http://en.wikipedia.org/wiki/Instructions_per_second)).

Commercial computers

The first commercial computer was the [Ferranti Mark 1](http://en.wikipedia.org/wiki/Ferranti_Mark_1), which was delivered to the [University of Manchester](http://en.wikipedia.org/wiki/University_of_Manchester) in February 1951. It was based on the [Manchester Mark 1](http://en.wikipedia.org/wiki/Manchester_Mark_1). The main improvements over the Manchester Mark 1 were in the size of the [primary storage](http://en.wikipedia.org/wiki/Primary_storage) (using [random access](http://en.wikipedia.org/wiki/Random-access_memory) [Williams tubes](http://en.wikipedia.org/wiki/Williams_tubes)), [secondary storage](http://en.wikipedia.org/wiki/Secondary_storage) (using a[magnetic drum](http://en.wikipedia.org/wiki/Drum_memory)), a faster multiplier, and additional instructions. The basic cycle time was 1.2 milliseconds, and a multiplication could be completed in about 2.16 milliseconds. The multiplier used almost a quarter of the machine's 4,050 vacuum tubes (valves).[[65]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-65) A second machine was purchased by the [University of Toronto](http://en.wikipedia.org/wiki/University_of_Toronto), before the design was revised into the [Mark 1 Star](http://en.wikipedia.org/wiki/Ferranti_Mark_1#Mark_1_Star). At least seven of these later machines were delivered between 1953 and 1957, one of them to [Shell](http://en.wikipedia.org/wiki/Royal_Dutch_Shell) labs in [Amsterdam](http://en.wikipedia.org/wiki/Amsterdam).[[66]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-66)

In October 1947, the directors of [J. Lyons & Company](http://en.wikipedia.org/wiki/J._Lyons_and_Co.), a British catering company famous for its teashops but with strong interests in new office management techniques, decided to take an active role in promoting the commercial development of computers. The [LEO I](http://en.wikipedia.org/wiki/LEO_computer) computer became operational in April 1951 [[67]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-67) and ran the world's first regular routine office computer [job](http://en.wikipedia.org/wiki/Job_(software)). On 17 November 1951, the J. Lyons company began weekly operation of a bakery valuations job on the LEO (Lyons Electronic Office). This was the first business [application](http://en.wikipedia.org/wiki/Category:Application_software)to go live on a stored program computer.[[68]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-68)

In June 1951, the [UNIVAC I](http://en.wikipedia.org/wiki/UNIVAC_I) (Universal Automatic Computer) was delivered to the [U.S. Census Bureau](http://en.wikipedia.org/wiki/United_States_Census_Bureau). Remington Rand eventually sold 46 machines at more than $1 million each ($8.99 million as of 2013).[[69]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-inflation-US-69)UNIVAC was the first "mass produced" computer. It used 5,200 vacuum tubes and consumed 125 kW of power. Its primary storage was [serial-access](http://en.wikipedia.org/wiki/Sequential_access) mercury delay lines capable of storing 1,000 words of 11decimal digits plus sign (72-bit words). A key feature of the UNIVAC system was a newly invented type of metal magnetic tape, and a high-speed tape unit, for non-volatile storage. Magnetic tape is still used in many computers.[[70]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-70) In 1952, IBM publicly announced the [IBM 701](http://en.wikipedia.org/wiki/IBM_701) Electronic Data Processing Machine, the first in its successful [700/7000 series](http://en.wikipedia.org/wiki/IBM_700/7000_series) and its first [IBM mainframe](http://en.wikipedia.org/wiki/IBM_mainframe) computer. The [IBM 704](http://en.wikipedia.org/wiki/IBM_704), introduced in 1954, used magnetic core memory, which became the standard for large machines. The first implemented high-level general purpose [programming language](http://en.wikipedia.org/wiki/Programming_language), [Fortran](http://en.wikipedia.org/wiki/Fortran), was also being developed at IBM for the 704 during 1955 and 1956 and released in early 1957. (Konrad Zuse's 1945 design of the high-level language [Plankalkül](http://en.wikipedia.org/wiki/Plankalk%C3%BCl) was not implemented at that time.) A volunteer [user group](http://en.wikipedia.org/wiki/User_group), which exists to this day, was founded in 1955 to [share](http://en.wikipedia.org/wiki/SHARE_(computing)) their software and experiences with the IBM 701.

[](http://en.wikipedia.org/wiki/File:IBM-650-panel.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:IBM-650-panel.jpg)

IBM 650 front panel

IBM introduced a smaller, more affordable computer in 1954 that proved very popular.[[71]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-71) The [IBM 650](http://en.wikipedia.org/wiki/IBM_650) weighed over 900 kg, the attached power supply weighed around 1350 kg and both were held in separate cabinets of roughly 1.5 meters by 0.9 meters by 1.8 meters. It cost $500,000[[72]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-72) ($4.35 million as of 2013) or could be leased for $3,500 a month ($30 thousand as of 2013).[[69]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-inflation-US-69) Its drum memory was originally 2,000 ten-digit words, later expanded to 4,000 words. Memory limitations such as this were to dominate programming for decades afterward. The program instructions were fetched from the spinning drum as the code ran. Efficient execution using drum memory was provided by a combination of hardware architecture: the instruction format included the address of the next instruction; and software: the Symbolic Optimal Assembly Program, SOAP,[[73]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-73) assigned instructions to the optimal addresses (to the extent possible by static analysis of the source program). Thus many instructions were, when needed, located in the next row of the drum to be read and additional wait time for drum rotation was not required.

[](http://en.wikipedia.org/wiki/File:IBM_350_RAMAC.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:IBM_350_RAMAC.jpg)

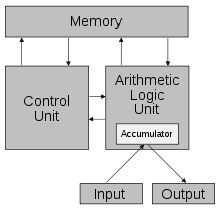
This [RAMAC](http://en.wikipedia.org/wiki/History_of_IBM_magnetic_disk_drives#Early_IBM_HDDs) [DASD](http://en.wikipedia.org/wiki/Direct_access_storage_device) is being restored at the [Computer History Museum](http://en.wikipedia.org/wiki/Computer_History_Museum)

In 1955, [Maurice Wilkes](http://en.wikipedia.org/wiki/Maurice_Wilkes) invented [microprogramming](http://en.wikipedia.org/wiki/Microprogram),[[74]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-74) which allows the base instruction set to be defined or extended by built-in programs (now called [firmware](http://en.wikipedia.org/wiki/Firmware) or [microcode](http://en.wikipedia.org/wiki/Microcode)).[[75]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-75) It was widely used in the [CPUs](http://en.wikipedia.org/wiki/Central_processing_unit) and [floating-point](http://en.wikipedia.org/wiki/Floating-point) units of [mainframe](http://en.wikipedia.org/wiki/Mainframe_computer) and other computers, such as the [Manchester](http://en.wikipedia.org/wiki/Manchester_University) [Atlas](http://en.wikipedia.org/wiki/Atlas_Computer) [[76]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-76) and the [IBM 360](http://en.wikipedia.org/wiki/IBM_360)series.[[77]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-77)

IBM introduced the first [disk storage](http://en.wikipedia.org/wiki/Disk_storage) unit (a [hard disk drive](http://en.wikipedia.org/wiki/Hard_disk_drive)), the [IBM 350 RAMAC](http://en.wikipedia.org/wiki/History_of_IBM_magnetic_disk_drives#Early_IBM_HDDs) (Random Access Method of Accounting and Control) in 1956. Using fifty 24-inch (610 mm) metal disks, with 100tracks per side, it was able to store 5[megabytes](http://en.wikipedia.org/wiki/Megabyte) of data at a cost of $10,000 per megabyte ($90 thousand as of 2013).[[69]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-inflation-US-69)[[78]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-78)

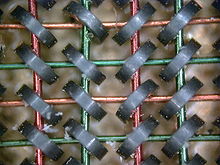
First-generation machines

*Further information:*[*List of vacuum tube computers*](http://en.wikipedia.org/wiki/List_of_vacuum_tube_computers)

[](http://en.wikipedia.org/wiki/File:Von_Neumann_architecture.svg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Von_Neumann_architecture.svg)

Design of the [von Neumann architecture](http://en.wikipedia.org/wiki/Von_Neumann_architecture)(1947)

[](http://en.wikipedia.org/wiki/File:Magnetic_core.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Magnetic_core.jpg)

[Magnetic core memory](http://en.wikipedia.org/wiki/Magnetic_core_memory). Each [core](http://en.wikipedia.org/wiki/Magnetic_core) is one [bit](http://en.wikipedia.org/wiki/Bit).

[](http://en.wikipedia.org/wiki/File:ENIAC_scientists_holding_various_parts_of_the_computer.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:ENIAC_scientists_holding_various_parts_of_the_computer.jpg)

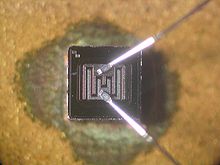
Parts from early computers: [ENIAC](http://en.wikipedia.org/wiki/ENIAC),[EDVAC](http://en.wikipedia.org/wiki/EDVAC), [ORDVAC](http://en.wikipedia.org/wiki/ORDVAC), and [BRLESC I](http://en.wikipedia.org/wiki/BRLESC)

Even before the ENIAC was finished, Eckert and Mauchly recognized its limitations and started the design of a [stored-program computer](http://en.wikipedia.org/wiki/Stored-program_computer), EDVAC. [John von Neumann](http://en.wikipedia.org/wiki/John_von_Neumann) was credited with a [widely circulated report](http://en.wikipedia.org/wiki/First_Draft_of_a_Report_on_the_EDVAC) describing the [EDVAC](http://en.wikipedia.org/wiki/EDVAC) design in which both the programs and working data were stored in a single, unified store. This basic design, denoted the [von Neumann architecture](http://en.wikipedia.org/wiki/Von_Neumann_architecture), would serve as the foundation for the worldwide development of ENIAC's successors.[[79]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-79) In this generation of equipment, temporary or working storage was provided by [acoustic delay lines](http://en.wikipedia.org/wiki/Acoustic_delay_line), which used the propagation time of sound through a medium such as liquid[mercury](http://en.wikipedia.org/wiki/Mercury_(element)) (or through a wire) to briefly store data. A series of [acoustic](http://en.wikipedia.org/wiki/Acoustics)pulses is sent along a tube; after a time, as the pulse reached the end of the tube, the circuitry detected whether the pulse represented a 1 or 0 and caused the oscillator to re-send the pulse. Others used[Williams tubes](http://en.wikipedia.org/wiki/Williams_tube), which use the ability of a small cathode-ray tube (CRT) to store and retrieve data as charged areas on the phosphor screen. By 1954, [magnetic core memory](http://en.wikipedia.org/wiki/Magnetic_core_memory)[[80]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-80) was rapidly displacing most other forms of temporary storage, and dominated the field through the mid-1970s.

EDVAC was the first stored-program computer designed; however it was not the first to run. Eckert and Mauchly left the project and its construction floundered. The first working von Neumann machine was the Manchester "Baby" or [Small-Scale Experimental Machine](http://en.wikipedia.org/wiki/Small-Scale_Experimental_Machine), developed by [Frederic C. Williams](http://en.wikipedia.org/wiki/Frederic_Calland_Williams) and [Tom Kilburn](http://en.wikipedia.org/wiki/Tom_Kilburn) at the [University of Manchester](http://en.wikipedia.org/wiki/University_of_Manchester) in 1948 as a test bed for the [Williams tube](http://en.wikipedia.org/wiki/Williams_tube);[[81]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-81) it was followed in 1949 by the [Manchester Mark 1](http://en.wikipedia.org/wiki/Manchester_Mark_1) computer, a complete system, using Williams tube and [magnetic drum](http://en.wikipedia.org/wiki/Magnetic_drum) memory, and introducing [index registers](http://en.wikipedia.org/wiki/Index_register).[[82]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-82) The other contender for the title "first digital stored-program computer" had been [EDSAC](http://en.wikipedia.org/wiki/EDSAC), designed and constructed at the [University of Cambridge](http://en.wikipedia.org/wiki/University_of_Cambridge). Operational less than one year after the Manchester "Baby", it was also capable of tackling real problems. EDSAC was actually inspired by plans for EDVAC (Electronic Discrete Variable Automatic Computer), the successor to ENIAC; these plans were already in place by the time ENIAC was successfully operational. Unlike ENIAC, which used parallel processing, EDVAC used a single processing unit. This design was simpler and was the first to be implemented in each succeeding wave of miniaturization, and increased reliability. Some view Manchester Mark 1 / EDSAC / EDVAC as the "Eves" from which nearly all current computers derive their architecture. Manchester University's machine became the prototype for the [Ferranti Mark 1](http://en.wikipedia.org/wiki/Ferranti_Mark_1). The first Ferranti Mark 1 machine was delivered to the University in February 1951 and at least nine others were sold between 1951 and 1957.

The first universal programmable computer in the Soviet Union was created by a team of scientists under direction of [Sergei Alekseyevich Lebedev](http://en.wikipedia.org/wiki/Sergei_Alekseyevich_Lebedev) from [Kiev Institute of Electrotechnology](http://en.wikipedia.org/wiki/Kiev_Institute_of_Electrotechnology), [Soviet Union](http://en.wikipedia.org/wiki/Soviet_Union) (now [Ukraine](http://en.wikipedia.org/wiki/Ukraine)). The computer [MESM](http://en.wikipedia.org/wiki/History_of_computer_hardware_in_communist_countries#MESM) (*МЭСМ*, *Small Electronic Calculating Machine*) became operational in 1950. It had about 6,000 vacuum tubes and consumed 25 kW of power. It could perform approximately 3,000 operations per second. Another early machine was [CSIRAC](http://en.wikipedia.org/wiki/CSIRAC), an Australian design that ran its first test program in 1949. CSIRAC is the oldest computer still in existence and the first to have been used to play digital music.[[83]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-83)

Second generation: transistors

[](http://en.wikipedia.org/wiki/File:Transistor-die-KSY34.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Transistor-die-KSY34.jpg)A [bipolar junction transistor](http://en.wikipedia.org/wiki/Bipolar_junction_transistor)

The bipolar [transistor](http://en.wikipedia.org/wiki/Transistor) was invented in 1947. From 1955 onwards transistors replaced [vacuum tubes](http://en.wikipedia.org/wiki/Vacuum_tube) in computer designs,[[84]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-84) giving rise to the "second generation" of computers. Initially the only devices available were [germanium](http://en.wikipedia.org/wiki/Germanium) [point-contact transistors](http://en.wikipedia.org/wiki/Point-contact_transistor), which although less reliable than the vacuum tubes they replaced had the advantage of consuming far less power.[[85]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-85) The first [transistorised computer](http://en.wikipedia.org/wiki/Transistor_computer) was built at the [University of Manchester](http://en.wikipedia.org/wiki/University_of_Manchester) and was operational by 1953;[[86]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-LavingtonP37-86) a second version was completed there in April 1955. The later machine used 200 transistors and 1,300 [solid-state](http://en.wikipedia.org/wiki/Solid-state_(electronics)) [diodes](http://en.wikipedia.org/wiki/Diode) and had a power consumption of 150 watts. However, it still required valves to generate the clock waveforms at 125 kHz and to read and write on the magnetic [drum memory](http://en.wikipedia.org/wiki/Drum_memory), whereas the[Harwell CADET](http://en.wikipedia.org/wiki/Harwell_CADET) operated without any valves by using a lower clock frequency, of 58 kHz when it became operational in February 1955.[[87]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-87) Problems with the reliability of early batches of point contact and alloyed junction transistors meant that the machine's [mean time between failures](http://en.wikipedia.org/wiki/Mean_time_between_failures) was about 90 minutes, but this improved once the more reliable [bipolar junction transistors](http://en.wikipedia.org/wiki/Bipolar_junction_transistor) became available.[[88]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-88)

Compared to vacuum tubes, transistors have many advantages: they are smaller, and require less power than vacuum tubes, so give off less heat. Silicon junction transistors were much more reliable than vacuum tubes and had longer, indefinite, service life. Transistorized computers could contain tens of thousands of binary logic circuits in a relatively compact space. Transistors greatly reduced computers' size, initial cost, and [operating cost](http://en.wikipedia.org/wiki/Operating_cost). Typically, second-generation computers were composed of large numbers of [printed circuit boards](http://en.wikipedia.org/wiki/Printed_circuit_board) such as the [IBM Standard Modular System](http://en.wikipedia.org/wiki/IBM_Standard_Modular_System)[[89]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-IBM_SMS-89) each carrying one to four [logic gates](http://en.wikipedia.org/wiki/Logic_gate) or [flip-flops](http://en.wikipedia.org/wiki/Flip-flop_(electronics)).

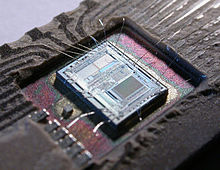
A second generation computer, the [IBM 1401](http://en.wikipedia.org/wiki/IBM_1401), captured about one third of the world market. IBM installed more than ten thousand 1401s between 1960 and 1964.

Transistorized electronics improved not only the [CPU](http://en.wikipedia.org/wiki/Central_processing_units) (Central Processing Unit), but also the [peripheral devices](http://en.wikipedia.org/wiki/Peripheral). The second generation [disk data storage units](http://en.wikipedia.org/wiki/Disk_storage) were able to store tens of millions of letters and digits. Next to the[fixed disk](http://en.wikipedia.org/wiki/Fixed_disk) storage units, connected to the CPU via high-speed data transmission, were removable disk data storage units. A removable [disk pack](http://en.wikipedia.org/wiki/Disk_pack) can be easily exchanged with another pack in a few seconds. Even if the removable disks' capacity is smaller than fixed disks, their interchangeability guarantees a nearly unlimited quantity of data close at hand. [Magnetic tape](http://en.wikipedia.org/wiki/Magnetic_tape_data_storage) provided archival capability for this data, at a lower cost than disk.

Many second-generation CPUs delegated peripheral device communications to a secondary processor. For example, while the communication processor controlled [card reading and punching](http://en.wikipedia.org/wiki/Unit_record_equipment), the main CPU executed calculations and binary [branch instructions](http://en.wikipedia.org/wiki/Branch_(computer_science)). One [databus](http://en.wikipedia.org/wiki/Bus_(computing)) would bear data between the main CPU and core memory at the CPU's [fetch-execute cycle](http://en.wikipedia.org/wiki/Fetch-execute_cycle) rate, and other databusses would typically serve the peripheral devices. On the [PDP-1](http://en.wikipedia.org/wiki/PDP-1), the core memory's cycle time was 5 microseconds; consequently most arithmetic instructions took 10 microseconds (100,000 operations per second) because most operations took at least two memory cycles; one for the instruction, one for the [operand](http://en.wikipedia.org/wiki/Operand) data fetch.

During the second generation [remote terminal](http://en.wikipedia.org/wiki/Remote_Digital_Terminal) units (often in the form of [teletype machines](http://en.wikipedia.org/wiki/Teleprinter) like a [Friden Flexowriter](http://en.wikipedia.org/wiki/Friden_Flexowriter)) saw greatly increased use.[[90]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-90) Telephone connections provided sufficient speed for early remote terminals and allowed hundreds of kilometers separation between remote-terminals and the computing center. Eventually these stand-alone computer networks would be generalized into an interconnected [*network of networks*](http://en.wikipedia.org/wiki/History_of_the_Internet)—the Internet.[[91]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-91)

Post-1960: third generation and beyond

[](http://en.wikipedia.org/wiki/File:Intel_8742_153056995.jpg)

[http://bits.wikimedia.org/static-1.23wmf3/skins/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:Intel_8742_153056995.jpg)

Intel [8742 eight-bit microcontroller IC](http://en.wikipedia.org/wiki/Integrated_circuit)

The explosion in the use of computers began with "third-generation" computers, making use of [Jack St. Clair Kilby](http://en.wikipedia.org/wiki/Jack_Kilby)'s[[92]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-92) and [Robert Noyce](http://en.wikipedia.org/wiki/Robert_Noyce)'s[[93]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-93) independent invention of the [integrated circuit](http://en.wikipedia.org/wiki/Integrated_circuit) (or microchip), which led to the invention of the [microprocessor](http://en.wikipedia.org/wiki/Microprocessor). While the subject of exactly which device was the first microprocessor is contentious, partly due to lack of agreement on the exact definition of the term "microprocessor", it is largely undisputed that the first single-chip microprocessor was the Intel 4004,[[94]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-94) designed and realized by [Ted Hoff](http://en.wikipedia.org/wiki/Marcian_Hoff), [Federico Faggin](http://en.wikipedia.org/wiki/Federico_Faggin), and Stanley Mazor at[Intel](http://en.wikipedia.org/wiki/Intel).[[95]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-95)

While the earliest microprocessor ICs literally contained only the processor, i.e. the central processing unit, of a computer, their progressive development naturally led to chips containing most or all of the internal electronic parts of a computer. The integrated circuit in the image on the right, for example, an [Intel](http://en.wikipedia.org/wiki/Intel) 8742, is an 8-bit [microcontroller](http://en.wikipedia.org/wiki/Microcontroller)that includes a [CPU](http://en.wikipedia.org/wiki/CPU) running at 12 MHz, 128 bytes of [RAM](http://en.wikipedia.org/wiki/RAM), 2048 bytes of [EPROM](http://en.wikipedia.org/wiki/EPROM), and [I/O](http://en.wikipedia.org/wiki/Input/output) in the same chip.

During the 1960s there was considerable overlap between second and third generation technologies.[[96]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-96) IBM implemented its [IBM Solid Logic Technology](http://en.wikipedia.org/wiki/IBM_Solid_Logic_Technology) modules in [hybrid circuits](http://en.wikipedia.org/wiki/Hybrid_circuit) for the IBM System/360 in 1964. As late as 1975, Sperry Univac continued the manufacture of second-generation machines such as the UNIVAC 494. The [Burroughs large systems](http://en.wikipedia.org/wiki/Burroughs_large_systems) such as the B5000 were [stack machines](http://en.wikipedia.org/wiki/Stack_machine), which allowed for simpler programming. These [pushdown automatons](http://en.wikipedia.org/wiki/Pushdown_automaton) were also implemented in minicomputers and microprocessors later, which influenced programming language design. Minicomputers served as low-cost computer centers for industry, business and universities.[[97]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-97) It became possible to simulate analog circuits with the *simulation program with integrated circuit emphasis*, or [SPICE](http://en.wikipedia.org/wiki/SPICE) (1971) on minicomputers, one of the programs for electronic design automation ([EDA](http://en.wikipedia.org/wiki/Category:Electronic_design_automation_software)). The microprocessor led to the development of the [microcomputer](http://en.wikipedia.org/wiki/Microcomputer), small, low-cost computers that could be owned by individuals and small businesses. Microcomputers, the first of which appeared in the 1970s, became ubiquitous in the 1980s and beyond.

In April 1975 at the Hannover Fair, [Olivetti](http://en.wikipedia.org/wiki/Olivetti) presented the [P6060](http://en.wikipedia.org/wiki/Olivetti_P6060), the world's first personal computer with built-in floppy disk: a central processing unit on two cards, code named PUCE1 and PUCE2, with [TTL](http://en.wikipedia.org/wiki/Transistor%E2%80%93transistor_logic) components. It had one or two 8" [floppy disk](http://en.wikipedia.org/wiki/Floppy_disk) drives, a 32-character [plasma display](http://en.wikipedia.org/wiki/Plasma_display), 80-column graphical [thermal printer](http://en.wikipedia.org/wiki/Thermal_printer), 48 Kbytes of [RAM](http://en.wikipedia.org/wiki/RAM), and [BASIC](http://en.wikipedia.org/wiki/BASIC) language. It weighed 40 kg (88 lb). It was in competition with a similar product by IBM that had an external floppy disk drive.

[MOS Technology KIM-1](http://en.wikipedia.org/wiki/KIM-1) and [Altair 8800](http://en.wikipedia.org/wiki/Altair_8800), were sold as kits for do-it-yourselfers, as was the [Apple I](http://en.wikipedia.org/wiki/Apple_I), soon afterward. The first Apple computer with graphic and sound capabilities came out well after the [Commodore PET](http://en.wikipedia.org/wiki/Commodore_PET). Computing has evolved with microcomputer architectures, with features added from their larger brethren, now dominant in most market segments.

Systems as complicated as computers require very high [reliability](http://en.wikipedia.org/wiki/Reliability_engineering). ENIAC remained on, in continuous operation from 1947 to 1955, for eight years before being shut down. Although a vacuum tube might fail, it would be replaced without bringing down the system. By the simple strategy of never shutting down ENIAC, the failures were dramatically reduced. The vacuum-tube SAGE air-defense computers became remarkably reliable – installed in pairs, one off-line, tubes likely to fail did so when the computer was intentionally run at reduced power to find them. [Hot-pluggable](http://en.wikipedia.org/wiki/Hot_plugging) hard disks, like the hot-pluggable vacuum tubes of yesteryear, continue the tradition of repair during continuous operation. Semiconductor memories routinely have no errors when they operate, although operating systems like Unix have employed memory tests on start-up to detect failing hardware. Today, the requirement of reliable performance is made even more stringent when [server farms](http://en.wikipedia.org/wiki/Server_farm) are the delivery platform.[[98]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-98) Google has managed this by using fault-tolerant software to recover from hardware failures, and is even working on the concept of replacing entire server farms on-the-fly, during a service event.[[99]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-99)[[100]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-100)

In the 21st century, [multi-core](http://en.wikipedia.org/wiki/Multi-core) CPUs became commercially available.[[101]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-101) [Content-addressable memory](http://en.wikipedia.org/wiki/Content-addressable_memory)(CAM)[[102]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-102) has become inexpensive enough to be used in networking, although no computer system has yet implemented hardware CAMs for use in programming languages. Currently, CAMs (or associative arrays) in software are programming-language-specific. Semiconductor memory cell arrays are very regular structures, and manufacturers prove their processes on them; this allows price reductions on memory products. During the 1980s, CMOS [logic gates](http://en.wikipedia.org/wiki/Logic_gates) developed into devices that could be made as fast as other circuit types; computer power consumption could therefore be decreased dramatically. Unlike the continuous current draw of a gate based on other logic types, a [CMOS](http://en.wikipedia.org/wiki/CMOS) gate only draws significant current during the 'transition' between logic states, except for leakage.

This has allowed computing to become a [commodity](http://en.wikipedia.org/wiki/Commodity) which is now ubiquitous, embedded in [many forms](http://en.wikipedia.org/wiki/Embedded_system), from greeting cards and [telephones](http://en.wikipedia.org/wiki/Mobile_phone) to [satellites](http://en.wikipedia.org/wiki/Satellite_communications#History). The [thermal design power](http://en.wikipedia.org/wiki/Thermal_design_power) which is dissipated during operation has become as essential as computing speed of operation. In 2006 servers consumed 1.5% of the total energy budget of the U.S.[[103]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-103) The energy consumption of computer data centers was expected to double to 3% of world consumption by 2011. The [SoC](http://en.wikipedia.org/wiki/System_on_a_chip) (system on a chip) has compressed even more of the [integrated circuitry](http://en.wikipedia.org/wiki/Integrated_circuit)into a single chip. Computing hardware and its software have even become a metaphor for the operation of the universe.[[104]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-104) Although [DNA-based computing](http://en.wikipedia.org/wiki/DNA_computing) and [quantum computing](http://en.wikipedia.org/wiki/Quantum_computing) are years or decades in the future, the infrastructure is being laid today, for example, with [DNA origami](http://en.wikipedia.org/wiki/DNA_origami) on photolithography[[105]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-105) and with quantum antennae for transferring information between ion traps.[[106]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-106) By 2011, researchers had [entangled](http://en.wikipedia.org/wiki/Qubit#Entanglement) [14](http://www.nanowerk.com/news/newsid=20823.php)[qubits](http://en.wikipedia.org/wiki/Qubit).[[107]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-107) Fast [digital circuits](http://en.wikipedia.org/wiki/Digital_circuit) (including those based on [Josephson junctions](http://en.wikipedia.org/wiki/Josephson_junction) and [rapid single flux quantum](http://en.wikipedia.org/wiki/Rapid_single_flux_quantum)technology) are becoming more nearly realizable with the discovery of [nanoscale superconductors](http://en.wikipedia.org/wiki/Nanoscale_superconductor).[[108]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-108)

Fiber-optic and photonic devices, which already have been used to transport data over long distances, are now entering the data center, side by side with CPU and semiconductor memory components. This allows the separation of RAM from CPU by optical interconnects.[[109]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-109) IBM has created an [integrated circuit](http://en.wikipedia.org/wiki/Integrated_circuit) with both electronic and optical (this is called *photonic*) information processing in one chip. This is denoted "CMOS-integrated nanophotonics" or (CINP).[[110]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-110) One benefit of optical interconnects is that motherboards which formerly required a certain kind of system on a chip (SoC) can now move formerly dedicated memory and network controllers off the motherboards, spreading the controllers out onto the rack. This allows standardization of backplane interconnects and motherboards for multiple types of SoCs, which allows more timely upgrades of CPUs.[[111]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-111)

An indication of the rapidity of development of this field can be inferred by the history of the seminal article.[[112]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-112)By the time that anyone had time to write anything down, it was obsolete. After 1945, others read John von Neumann's *First Draft of a Report on the EDVAC*, and immediately started implementing their own systems. To this day, the pace of development has continued, worldwide.[[113]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-113)[[114]](http://en.wikipedia.org/wiki/History_of_computing_hardware#cite_note-114)

**Computer network**or data network is a telecommunications network that allows computers to exchange data. In computer networks, networked computing devices (network nodes) pass data to each other along data connections. The connections (network links) between nodes are established using either cable media or wireless media. The best-known computer network is the Internet.

Network devices that originate, route and terminate the data are called network nodes.[1] Nodes can include hosts such as servers and personal computers, as well as networking hardware. Two devices are said to be networked when a device is able to exchange information with another device.

Computer networks support applications such as access to the World Wide Web, shared use of application and storage servers, printers, and fax machines, and use of email and instant messaging applications. Computer networks differ in the physical media used to transmit their signals, the communications protocols to organize network traffic, the network's size, topology and organizational intent.

**History**

In the late 1950s, early networks of communicating computers included the military radar system Semi-Automatic Ground Environment (SAGE).

In 1960, the commercial airline reservation system semi-automatic business research environment (SABRE) went online with two connected mainframes.

In 1962, J.C.R. Licklider developed a working group he called the "Intergalactic Computer Network", a precursor to the ARPANET, at the Advanced Research Projects Agency (ARPA).

In 1964, researchers at Dartmouth developed the Dartmouth Time Sharing System for distributed users of large computer systems. The same year, at Massachusetts Institute of Technology, a research group supported by General Electric and Bell Labs used a computer to route and manage telephone connections.

Throughout the 1960s, Leonard Kleinrock, Paul Baran and Donald Davies independently conceptualized and developed network systems which used packets to transfer information between computers over a network.

In 1965, Thomas Marill and Lawrence G. Roberts created the first wide area network (WAN). This was an immediate precursor to the ARPANET, of which Roberts became program manager.

Also in 1965, the first widely used telephone switch that implemented true computer control was introduced by Western Electric.

In 1969, the University of California at Los Angeles, the Stanford Research Institute, the University of California at Santa Barbara, and the University of Utah were connected as the beginning of the ARPANET network using 50 kbit/s circuits.[2]

In 1972, commercial services using X.25 were deployed, and later used as an underlying infrastructure for expanding TCP/IP networks.

In 1973, Robert Metcalfe wrote a formal memo at Xerox PARC describing Ethernet, a networking system that was based on the Aloha network, developed in the 1960s by Norman Abramson and colleagues at the University of Hawaii. In July 1976, Robert Metcalfe and David Boggs published their paper "Ethernet: Distributed Packet Switching for Local Computer Networks"[3] and collaborated on several patents received in 1977 and 1978. In 1979, Robert Metcalfe pursued making Ethernet an open standard.[4]

In 1976, John Murphy of Datapoint Corporation created ARCNET, a token-passing network first used to share storage devices.

In 1995, the transmission speed capacity for Ethernet was increased from 10 Mbit/s to 100 Mbit/s. By 1998, Ethernet supported transmission speeds of a Gigabit. The ability of Ethernet to scale easily (such as quickly adapting to support new fiber optic cable speeds) is a contributing factor to its continued use today.[4]

Today, computer networks are the core of modern communication. All modern aspects of the public switched telephone network (PSTN) are computer-controlled. Telephony increasingly runs over the Internet Protocol, although not necessarily the public Internet. The scope of communication has increased significantly in the past decade. This boom in communications would not have been possible without the progressively advancing computer network. Computer networks, and the technologies that make communication between networked computers possible, continue to drive computer hardware, software, and peripherals industries. The expansion of related industries is mirrored by growth in the numbers and types of people using networks, from the researcher to the home user.

**Properties**

Computer networking may be considered a branch of electrical engineering, telecommunications, computer science, information technology or computer engineering, since it relies upon the theoretical and practical application of the related disciplines.

A computer network has the following properties:

Facilitates interpersonal communications

People can communicate efficiently and easily via email, instant messaging, chat rooms, telephone, video telephone calls, and video conferencing.

Allows sharing of files, data, and other types of information

Authorized users may access information stored on other computers on the network. Providing access to information on shared storage devices is an important feature of many networks.

Allows sharing of network and computing resources

Users may access and use resources provided by devices on the network, such as printing a document on a shared network printer. Distributed computing uses computing resources across a network to accomplish tasks.

May be insecure

A computer network may be used by computer Hackers to deploy computer viruses or computer worms on devices connected to the network, or to prevent these devices from accessing the network (denial of service).

May interfere with other technologies

Power line communication strongly disturbs certain[5] forms of radio communication, e.g., amateur radio. It may also interfere with last mile access technologies such as ADSL and VDSL.

May be difficult to set up

A complex computer network may be difficult to set up. It may be costly to set up an effective computer network in a large organization.

**Network topology.**

The physical layout of a network is usually somewhat less important than the topology by which network nodes are connected. Most diagrams that are drawn to describe a physical network are therefore topological, rather than geographic. The symbols on these diagrams usually denote network links and network nodes.

**Network links.**

The communication media used to link devices to form a computer network include electrical cable (HomePNA, power line communication, G.hn), optical fiber (fiber-optic communication), and radio waves (wireless networking). In the OSI model, these are defined at layers 1 and 2 — the physical layer and the data link layer.

A widely adopted family of communication media used in local area network (LAN) technology is collectively known as Ethernet. The media and protocol standards that enable communication between networked devices over Ethernet are defined by IEEE 802.3. Ethernet transmit data over both copper and fiber cables. Wireless LAN standards (e.g. those defined by IEEE 802.11) use radio waves, or others use infrared signals as a transmission medium. Power line communication uses a building's power cabling to transmit data.

**Wired technologies.**

Fiber optic cables are used to transmit light from one computer/network node to another

The orders of the following wired technologies are, roughly, from slowest to fastest transmission speed.

Twisted pair wire is the most widely used medium for all telecommunication. Twisted-pair cabling consist of copper wires that are twisted into pairs. Ordinary telephone wires consist of two insulated copper wires twisted into pairs. Computer network cabling (wired Ethernet as defined by IEEE 802.3) consists of 4 pairs of copper cabling that can be utilized for both voice and data transmission. The use of two wires twisted together helps to reduce crosstalk and electromagnetic induction. The transmission speed ranges from 2 million bits per second to 10 billion bits per second. Twisted pair cabling comes in two forms: unshielded twisted pair (UTP) and shielded twisted-pair (STP). Each form comes in several category ratings, designed for use in various scenarios.

Coaxial cable is widely used for cable television systems, office buildings, and other work-sites for local area networks. The cables consist of copper or aluminum wire surrounded by an insulating layer (typically a flexible material with a high dielectric constant), which itself is surrounded by a conductive layer. The insulation helps minimize interference and distortion. Transmission speed ranges from 200 million bits per second to more than 500 million bits per second.

ITU-T G.hn technology uses existing home wiring (coaxial cable, phone lines and power lines) to create a high-speed (up to 1 Gigabit/s) local area network.

An optical fiber is a glass fiber. It uses pulses of light to transmit data. Some advantages of optical fibers over metal wires are less transmission loss, immunity from electromagnetic radiation, and very fast transmission speeds of up to trillions of bits per second. One can use different colors of lights to increase the number of messages being sent over a fiber optic cable.

**Wireless technologies.**

Computers are very often connected to networks using wireless links

**Terrestrial microwave** – Terrestrial microwave communication uses Earth-based transmitters and receivers resembling satellite dishes. Terrestrial microwaves are in the low-gigahertz range, which limits all communications to line-of-sight. Relay stations are spaced approximately 48 km (30 mi) apart.

**Communications satellites** – Satellites communicate via microwave radio waves, which are not deflected by the Earth's atmosphere. The satellites are stationed in space, typically in geosynchronous orbit 35,400 km (22,000 mi) above the equator. These Earth-orbiting systems are capable of receiving and relaying voice, data, and TV signals.

Cellular and PCS systems use several radio communications technologies. The systems divide the region covered into multiple geographic areas. Each area has a low-power transmitter or radio relay antenna device to relay calls from one area to the next area.

Radio and spread spectrum technologies – Wireless local area networks use a high-frequency radio technology similar to digital cellular and a low-frequency radio technology. Wireless LANs use spread spectrum technology to enable communication between multiple devices in a limited area. IEEE 802.11 defines a common flavor of open-standards wireless radio-wave technology known as Wifi.

Free-space optical communication uses visible or invisible light for communications. In most cases, line-of-sight propagation is used, which limits the physical positioning of communicating devices.

**Exotic technologies.**

There have been various attempts at transporting data over exotic media:

IP over Avian Carriers was a humorous April fool's Request for Comments, issued as RFC 1149. It was implemented in real life in 2001.[6]

Extending the Internet to interplanetary dimensions via radio waves.[7]

Both cases have a large round-trip delay time, which gives slow two-way communication, but doesn't prevent sending large amounts of information.

**Network nodes.**

Apart from the physical communications media described above, networks comprise additional basic hardware building blocks, such as network interface controller cards (NICs), repeaters, hubs, bridges, switches, routers, modems, and firewalls.

**Network interfaces.**

A network interface controller (NIC) is a hardware accessory that provides a computer with both a physical interface for accepting a network cable connector and the ability to process low-level network information.

In Ethernet networks, each network interface controller has a unique Media Access Control (MAC) address which is usually stored in the card's permanent memory. MAC address uniqueness is maintained and administered by the Institute of Electrical and Electronics Engineers (IEEE) in order to avoid address conflicts between devices on a network. The size of an Ethernet MAC address is six octets. The 3 most significant octets are reserved to identify card manufacturers. The card manufacturers, using only their assigned prefixes, uniquely assign the 3 least-significant octets of every Ethernet card they produce.

**Repeaters and hubs.**

A repeater is an electronic device that receives a network signal, cleans it of unnecessary noise, and regenerates it. The signal is retransmitted at a higher power level, or to the other side of an obstruction, so that the signal can cover longer distances without degradation. In most twisted pair Ethernet configurations, repeaters are required for cable that runs longer than 100 meters. A repeater with multiple ports is known as a hub. Repeaters work on the physical layer of the OSI model. Repeaters require a small amount of time to regenerate the signal. This can cause a propagation delay which can affect network performance. As a result, many network architectures limit the number of repeaters that can be used in a row, e.g., the Ethernet 5-4-3 rule.

Repeaters and hubs have been mostly obsoleted by modern switches.

**Bridges.**

A network bridge connects multiple network segments at the data link layer (layer 2) of the OSI model to form a single network. Bridges broadcast to all ports except the port on which the broadcast was received. However, bridges do not promiscuously copy traffic to all ports, as hubs do. Instead, bridges learn which MAC addresses are reachable through specific ports. Once the bridge associates a port with an address, it will send traffic for that address to that port only.

Bridges learn the association of ports and addresses by examining the source address of frames that it sees on various ports. Once a frame arrives through a port, the bridge assumes that the MAC address is associated with that port and stores its source address. The first time a bridge sees a previously unknown destination address, the bridge will forward the frame to all ports other than the one on which the frame arrived.

Bridges come in three basic types:

**Local bridges**: Directly connect LANs

Remote bridges: Can be used to create a wide area network (WAN) link between LANs. Remote bridges, where the connecting link is slower than the end networks, largely have been replaced with routers.

**Wireless bridges**: Can be used to join LANs or connect remote devices to LANs.

**Switches.**

A network switch is a device that forwards and filters OSI layer 2 datagrams between ports based on the MAC addresses in the packets.[8] A switch is distinct from a hub in that it only forwards the frames to the ports involved in the communication rather than all ports connected. A switch breaks the collision domain but represents itself as a broadcast domain. Switches make decisions about where to forward frames based on MAC addresses. A switch normally has numerous ports, facilitating a star topology for devices, and cascading additional switches. Multi-layer switches are capable of routing based on layer 3 addressing or additional logical levels. The term switch is often used loosely to include devices such as routers and bridges, as well as devices that may distribute traffic based on load or based on application content (e.g., a Web URL identifier).

**Routers.**

A typical home or small office router showing the ADSL telephone line and Ethernet network cable connections

A router is an internetworking device that forwards packets between networks by processing the routing information included in the packet or datagram (Internet protocol information from layer 3). The routing information is often processed in conjunction with the routing table (or forwarding table). A router uses its routing table to determine where to forward packets. (A destination in a routing table can include a "null" interface, also known as the "black hole" interface because data can go into it, however, no further processing is done for said data.)

**Modems.**

Modems (MOdulator-DEModulator) are used to connect network nodes via wire not originally designed for digital network traffic, or for wireless. To do this one or more frequencies are modulated by the digital signal to produce an analog signal that can be tailored to give the required properties for transmission. Modems are commonly used for telephone lines, using a Digital Subscriber Line technology.

**Firewalls.**

A firewall is a network device for controlling network security and access rules. Firewalls are typically configured to reject access requests from unrecognized sources while allowing actions from recognized ones. The vital role firewalls play in network security grows in parallel with the constant increase in cyber attacks.

**Network structure.**

Network topology is the layout or organizational hierarchy of interconnected nodes of a computer network. Different network topologies can affect throughput, but reliability is often more critical. With many technologies, such as bus networks, a single failure can cause the network to fail entirely. In general the more interconnections there are, the more robust the network is; but the more expensive it is to install.

**Common layouts.**

**Common network topologies**

Common layouts are:

**A bus network**: all nodes are connected to a common medium along this medium. This was the layout used in the original Ethernet, called 10BASE5 and 10BASE2.

**A star network**: all nodes are connected to a special central node. This is the typical layout found in a Wireless LAN, where each wireless client connects to the central Wireless access point.

**A ring network**: each node is connected to its left and right neighbour node, such that all nodes are connected and that each node can reach each other node by traversing nodes left- or rightwards. The Fiber Distributed Data Interface (FDDI) made use of such a topology.

**A mesh network**: each node is connected to an arbitrary number of neighbours in such a way that there is at least one traversal from any node to any other.

**A fully connected network**: each node is connected to every other node in the network.

**A tree network:** nodes are arranged hierarchically.

Note that the physical layout of the nodes in a network may not necessarily reflect the network topology. As an example, with FDDI, the network topology is a ring (actually two counter-rotating rings), but the physical topology is often a star, because all neighboring connections can be routed via a central physical location.

**Overlay network.**

An overlay network is a virtual computer network that is built on top of another network. Nodes in the overlay network are connected by virtual or logical links. Each link corresponds to a path, perhaps through many physical links, in the underlying network. The topology of the overlay network may (and often does) differ from that of the underlying one. For example, many peer-to-peer networks are overlay networks. They are organized as nodes of a virtual system of links that run on top of the Internet.[9]

Overlay networks have been around since the invention of networking when computer systems were connected over telephone lines using modems, before any data network existed.

The most striking example of an overlay network is the Internet itself. The Internet itself was initially built as an overlay on the telephone network.[9] Even today, at the network layer, each node can reach any other by a direct connection to the desired IP address, thereby creating a fully connected network. The underlying network, however, is composed of a mesh-like interconnect of sub-networks of varying topologies (and technologies). Address resolution and routing are the means that allow mapping of a fully connected IP overlay network to its underlying network.

Another example of an overlay network is a distributed hash table, which maps keys to nodes in the network. In this case, the underlying network is an IP network, and the overlay network is a table (actually a map) indexed by keys.

Overlay networks have also been proposed as a way to improve Internet routing, such as through quality of service guarantees to achieve higher-quality streaming media. Previous proposals such as IntServ, DiffServ, and IP Multicast have not seen wide acceptance largely because they require modification of all routers in the network.[citation needed] On the other hand, an overlay network can be incrementally deployed on end-hosts running the overlay protocol software, without cooperation from Internet service providers. The overlay network has no control over how packets are routed in the underlying network between two overlay nodes, but it can control, for example, the sequence of overlay nodes that a message traverses before it reaches its destination.

For example, Akamai Technologies manages an overlay network that provides reliable, efficient content delivery (a kind of multicast). Academic research includes end system multicast,[10] resilient routing and quality of service studies, among others.

**Communications protocols.**

The TCP/IP model or Internet layering scheme and its relation to some common protocols which are often layered on top of it.

A communications protocol is a set of rules for exchanging information over network links. In a protocol stack (also see the OSI model), each protocol leverages the services of the protocol below it. An important example of a protocol stack is HTTP running over TCP over IP over IEEE 802.11. (TCP and IP are members of the Internet Protocol Suite. IEEE 802.11 is a member of the Ethernet protocol suite.) This stack is used between the wireless router and the home user's personal computer when the user is surfing the web.

Communication protocols have various characteristics. They may be connection-oriented or connectionless, they may use circuit mode or packet switching, and they may use hierarchical addressing or flat addressing.

There are many communication protocols, a few of which are described below.

**Ethernet.**

Ethernet is a family of protocols used in LANs, described by a set of standards together called IEEE 802 published by the Institute of Electrical and Electronics Engineers. It has a flat addressing scheme. It operates mostly at levels 1 and 2 of the OSI model. For home users today, the most well-known member of this protocol family is IEEE 802.11, otherwise known as Wireless LAN (WLAN). The complete IEEE 802 protocol suite provides a diverse set of networking capabilities. For example, MAC bridging (IEEE 802.1D) deals with the routing of Ethernet packets using a Spanning Tree Protocol, IEEE 802.1Q describes VLANs, and IEEE 802.1X defines a port-based Network Access Control protocol, which forms the basis for the authentication mechanisms used in VLANs (but it is also found in WLANs) – it is what the home user sees when the user has to enter a "wireless access key".

**Internet Protocol Suite.**

The Internet Protocol Suite, also called TCP/IP, is the foundation of all modern internetworking. It offers connection-less as well as connection-oriented services over an inherently unreliable network traversed by datagram transmission at the Internet protocol (IP) level. At its core, the protocol suite defines the addressing, identification, and routing specifications for Internet Protocol Version 4 (IPv4) and for IPv6, the next generation of the protocol with a much enlarged addressing capability.

Synchronous optical networking (SONET) and Synchronous Digital Hierarchy (SDH) are standardized multiplexing protocols that transfer multiple digital bit streams over optical fiber using lasers. They were originally designed to transport circuit mode communications from a variety of different sources, primarily to support real-time, uncompressed, circuit-switched voice encoded in PCM(Pulse-Code Modulation) format. However, due to its protocol neutrality and transport-oriented features, SONET/SDH also was the obvious choice for transporting Asynchronous Transfer Mode (ATM) frames.

**Asynchronous Transfer Mode.**

Asynchronous Transfer Mode (ATM) is a switching technique for telecommunication networks. It uses asynchronous time-division multiplexing and encodes data into small, fixed-sized cells. This differs from other protocols such as the Internet Protocol Suite or Ethernet that use variable sized packets or frames. ATM has similarity with both circuit and packet switched networking. This makes it a good choice for a network that must handle both traditional high-throughput data traffic, and real-time, low-latency content such as voice and video. ATM uses a connection-oriented model in which a virtual circuit must be established between two endpoints before the actual data exchange begins.

While the role of ATM is diminishing in favor of next-generation networks, it still plays a role in the last mile, which is the connection between an Internet service provider and the home user. For an interesting write-up of the technologies involved, including the deep stacking of communications protocols used.

**Geographic scale.**

A network can be characterized by its physical capacity or its organizational purpose. Use of the network, including user authorization and access rights, differ accordingly.

**Personal area network**

A personal area network (PAN) is a computer network used for communication among computer and different information technological devices close to one person. Some examples of devices that are used in a PAN are personal computers, printers, fax machines, telephones, PDAs, scanners, and even video game consoles. A PAN may include wired and wireless devices. The reach of a PAN typically extends to 10 meters.[12] A wired PAN is usually constructed with USB and Firewire connections while technologies such as Bluetooth and infrared communication typically form a wireless PAN.

**Local area network**

A local area network (LAN) is a network that connects computers and devices in a limited geographical area such as a home, school, office building, or closely positioned group of buildings. Each computer or device on the network is a node. Wired LANs are most likely based on Ethernet technology. Newer standards such as ITU-T G.hn also provide a way to create a wired LAN using existing wiring, such as coaxial cables, telephone lines, and power lines.[13]

A LAN is depicted in the accompanying diagram. All interconnected devices use the network layer (layer 3) to handle multiple subnets (represented by different colors). Those inside the library have 10/100 Mbit/s Ethernet connections to the user device and a Gigabit Ethernet connection to the central router. They could be called Layer 3 switches, because they only have Ethernet interfaces and support the Internet Protocol. It might be more correct to call them access routers, where the router at the top is a distribution router that connects to the Internet and to the academic networks' customer access routers.

The defining characteristics of a LAN, in contrast to a wide area network (WAN), include higher data transfer rates, limited geographic range, and lack of reliance on leased lines to provide connectivity. Current Ethernet or other IEEE 802.3 LAN technologies operate at data transfer rates up to 10 Gbit/s. The IEEE investigates the standardization of 40 and 100 Gbit/s rates.[14] A LAN can be connected to a WAN using a router.

**Home area network**

A home area network (HAN) is a residential LAN which is used for communication between digital devices typically deployed in the home, usually a small number of personal computers and accessories, such as printers and mobile computing devices. An important function is the sharing of Internet access, often a broadband service through a cable TV or digital subscriber line (DSL) provider.

**Storage area network**

A storage area network (SAN) is a dedicated network that provides access to consolidated, block level data storage. SANs are primarily used to make storage devices, such as disk arrays, tape libraries, and optical jukeboxes, accessible to servers so that the devices appear like locally attached devices to the operating system. A SAN typically has its own network of storage devices that are generally not accessible through the local area network by other devices. The cost and complexity of SANs dropped in the early 2000s to levels allowing wider adoption across both enterprise and small to medium sized business environments.

**Campus area network**

A campus area network (CAN) is made up of an interconnection of LANs within a limited geographical area. The networking equipment (switches, routers) and transmission media (optical fiber, copper plant, Cat5 cabling, etc.) are almost entirely owned by the campus tenant / owner (an enterprise, university, government, etc.).

For example, a university campus network is likely to link a variety of campus buildings to connect academic colleges or departments, the library, and student residence halls.

**Backbone network**

A backbone network is part of a computer network infrastructure that provides a path for the exchange of information between different LANs or sub-networks. A backbone can tie together diverse networks within the same building, across different buildings, or over a wide area.

For example, a large company might implement a backbone network to connect departments that are located around the world. The equipment that ties together the departmental networks constitutes the network backbone. When designing a network backbone, network performance and network congestion are critical factors to take into account. Normally, the backbone network's capacity is greater than that of the individual networks connected to it.

Another example of a backbone network is the Internet backbone, which is the set of wide area networks (WANs) and core routers that tie together all networks connected to the Internet.

**Metropolitan area network**

A Metropolitan area network (MAN) is a large computer network that usually spans a city or a large campus.

**Wide area network**

A wide area network (WAN) is a computer network that covers a large geographic area such as a city, country, or spans even intercontinental distances. A WAN uses a communications channel that combines many types of media such as telephone lines, cables, and air waves. A WAN often makes use of transmission facilities provided by common carriers, such as telephone companies. WAN technologies generally function at the lower three layers of the OSI reference model: the physical layer, the data link layer, and the network layer.

**Enterprise private network**

An enterprise private network is a network built by a single organization to interconnect its office locations (e.g., production sites, head offices, remote offices, shops) in order to share computer resources.

**Virtual private network**

A virtual private network (VPN) is an overlay network in which some of the links between nodes are carried by open connections or virtual circuits in some larger network (e.g., the Internet) instead of by physical wires. The data link layer protocols of the virtual network are said to be tunneled through the larger network when this is the case. One common application is secure communications through the public Internet, but a VPN need not have explicit security features, such as authentication or content encryption. VPNs, for example, can be used to separate the traffic of different user communities over an underlying network with strong security features.

VPN may have best-effort performance, or may have a defined service level agreement (SLA) between the VPN customer and the VPN service provider. Generally, a VPN has a topology more complex than point-to-point.

**Global area network**

A global area network (GAN) is a network used for supporting mobile across an arbitrary number of wireless LANs, satellite coverage areas, etc. The key challenge in mobile communications is handing off user communications from one local coverage area to the next. In IEEE Project 802, this involves a succession of terrestrial wireless LANs.

**Organizational scope.**

Networks are typically managed by the organizations that own them. Private enterprise networks may use a combination of intranets and extranets. They may also provide network access to the Internet, which has no single owner and permits virtually unlimited global connectivity.

**Intranets.**

An intranet is a set of networks that are under the control of a single administrative entity. The intranet uses the IP protocol and IP-based tools such as web browsers and file transfer applications. The administrative entity limits use of the intranet to its authorized users. Most commonly, an intranet is the internal LAN of an organization. A large intranet will typically have at least one web server to provide users with organizational information.

**Extranet.**

An extranet is a network that is also under the administrative control of a single organization, but supports a limited connection to a specific external network. For example, an organization may provide access to some aspects of its intranet to share data with its business partners or customers. These other entities are not necessarily trusted from a security standpoint. Network connection to an extranet is often, but not always, implemented via WAN technology.

**Internetwork.**

An internetwork is the connection of multiple computer networks via a common routing technology using routers.

Partial map of the Internet based on the January 15, 2005 data found on opte.org. Each line is drawn between two nodes, representing two IP addresses. The length of the lines are indicative of the delay between those two nodes. This graph represents less than 30% of the Class C networks reachable.

The Internet is the largest example of an internetwork. It is a global system of interconnected governmental, academic, corporate, public, and private computer networks. It is based on the networking technologies of the Internet Protocol Suite. It is the successor of the Advanced Research Projects Agency Network (ARPANET) developed by DARPA of the United States Department of Defense. The Internet is also the communications backbone underlying the World Wide Web (WWW).

Participants in the Internet use a diverse array of methods of several hundred documented, and often standardized, protocols compatible with the Internet Protocol Suite and an addressing system (IP addresses) administered by the Internet Assigned Numbers Authority and address registries. Service providers and large enterprises exchange information about the reachability of their address spaces through the Border Gateway Protocol (BGP), forming a redundant worldwide mesh of transmission paths.

**Routing.**

Routing calculates good paths through a network for information to take. For example from node 1 to node 6 the best routes are likely to be 1-8-7-6 or 1-8-10-6, as this has the thickest routes.

Routing is the process of selecting paths in a network along which to send network traffic. Routing is performed for many kinds of networks, including circuit switching networks and packet switched networks.

In packet switched networks, routing directs packet forwarding (the transit of logically addressed network packets from their source toward their ultimate destination) through intermediate nodes. Intermediate nodes are typically network hardware devices such as routers, bridges, gateways, firewalls, or switches. General-purpose computers can also forward packets and perform routing, though they are not specialized hardware and may suffer from limited performance. The routing process usually directs forwarding on the basis of routing tables which maintain a record of the routes to various network destinations. Thus, constructing routing tables, which are held in the router's memory, is very important for efficient routing. Most routing algorithms use only one network path at a time. Multipath routing techniques enable the use of multiple alternative paths.

There are usually multiple routes that can be taken, and to choose between them, different elements can be considered to decide which routes get installed into the routing table, such as (sorted by priority):

Prefix-Length: where longer subnet masks are preferred (independent if it is within a routing protocol or over different routing protocol)

Metric: where a lower metric/cost is preferred (only valid within one and the same routing protocol)

Administrative distance: where a lower distance is preferred (only valid between different routing protocols)

Routing, in a more narrow sense of the term, is often contrasted with bridging in its assumption that network addresses are structured and that similar addresses imply proximity within the network. Structured addresses allow a single routing table entry to represent the route to a group of devices. In large networks, structured addressing (routing, in the narrow sense) outperforms unstructured addressing (bridging). Routing has become the dominant form of addressing on the Internet. Bridging is still widely used within localized environments.

**Network service.**

Network services are applications hosted by servers on a computer network, to provide some functionality for members or users of the network, or to help the network itself to operate.

The World Wide Web, E-mail,[16] printing and network file sharing are examples of well-known network services. Network services such as DNS (Domain Name System) give names for IP and MAC addresses (people remember names like “nm.lan” better than numbers like “210.121.67.18”),[17] and DHCP to ensure that the equipment on the network has a valid IP address.[18]

Services are usually based on a service protocol which defines the format and sequencing of messages between clients and servers of that network service.

**Network performance.**

**Quality of service.**

Depending on the installation requirements, network performance is usually measured by the quality of service of a telecommunications product. The parameters that affect this typically can include throughput, jitter, bit error rate and latency.

The following list gives examples of network performance measures for a circuit-switched network and one type of packet-switched network, viz. ATM:

Circuit-switched networks: In circuit switched networks, network performance is synonymous with the grade of service. The number of rejected calls is a measure of how well the network is performing under heavy traffic loads.[19] Other types of performance measures can include the level of noise and echo.

**ATM**: In an Asynchronous Transfer Mode (ATM) network, performance can be measured by line rate, quality of service (QoS), data throughput, connect time, stability, technology, modulation technique and modem enhancements.[20]

There are many ways to measure the performance of a network, as each network is different in nature and design. Performance can also be modelled instead of measured. For example, state transition diagrams are often used to model queuing performance in a circuit-switched network. These diagrams allow the network planner to analyze how the network will perform in each state, ensuring that the network will be optimally designed.

**Network congestion.**

Network congestion occurs when a link or node is carrying so much data that its quality of service deteriorates. Typical effects include queueing delay, packet loss or the blocking of new connections. A consequence of these latter two is that incremental increases in offered load lead either only to small increase in network throughput, or to an actual reduction in network throughput.

Network protocols which use aggressive retransmissions to compensate for packet loss tend to keep systems in a state of network congestion even after the initial load has been reduced to a level which would not normally have induced network congestion. Thus, networks using these protocols can exhibit two stable states under the same level of load. The stable state with low throughput is known as congestive collapse.

Modern networks use congestion control and congestion avoidance techniques to try to avoid congestion collapse. These include: exponential backoff in protocols such as 802.11's CSMA/CA and the original Ethernet, window reduction in TCP, and fair queueing in devices such as routers. Another method to avoid the negative effects of network congestion is implementing priority schemes, so that some packets are transmitted with higher priority than others. Priority schemes do not solve network congestion by themselves, but they help to alleviate the effects of congestion for some services. An example of this is 802.1p. A third method to avoid network congestion is the explicit allocation of network resources to specific flows. One example of this is the use of Contention-Free Transmission Opportunities (CFTXOPs) in the ITU-T G.hn standard, which provides high-speed (up to 1 Gbit/s) Local area networking over existing home wires (power lines, phone lines and coaxial cables).

**Network security.**

Network security consists of the provisions and policies adopted by the network administrator to prevent and monitor unauthorized access, misuse, modification, or denial of the computer network and its network-accessible resources.[22] Network security is the authorization of access to data in a network, which is controlled by the network administrator. Users are assigned an ID and password that allows them access to information and programs within their authority. Network security is used on a variety of computer networks, both public and private, to secure daily transactions and communications among businesses, government agencies and individuals.

**Network resilience.**

Network resilience is "the ability to provide and maintain an acceptable level of service in the face of faults and challenges to normal operation.”

**Views of networks.**

Users and network administrators typically have different views of their networks. Users can share printers and some servers from a workgroup, which usually means they are in the same geographic location and are on the same LAN, whereas a Network Administrator is responsible to keep that network up and running. A community of interest has less of a connection of being in a local area, and should be thought of as a set of arbitrarily located users who share a set of servers, and possibly also communicate via peer-to-peer technologies.

Network administrators can see networks from both physical and logical perspectives. The physical perspective involves geographic locations, physical cabling, and the network elements (e.g., routers, bridges and application layer gateways) that interconnect the physical media. Logical networks, called, in the TCP/IP architecture, subnets, map onto one or more physical media. For example, a common practice in a campus of buildings is to make a set of LAN cables in each building appear to be a common subnet, using virtual LAN (VLAN) technology.

Both users and administrators will be aware, to varying extents, of the trust and scope characteristics of a network. Again using TCP/IP architectural terminology, an intranet is a community of interest under private administration usually by an enterprise, and is only accessible by authorized users (e.g. employees).[24] Intranets do not have to be connected to the Internet, but generally have a limited connection. An extranet is an extension of an intranet that allows secure communications to users outside of the intranet (e.g. business partners, customers).[24]

Unofficially, the Internet is the set of users, enterprises, and content providers that are interconnected by Internet Service Providers (ISP). From an engineering viewpoint, the Internet is the set of subnets, and aggregates of subnets, which share the registered IP address space and exchange information about the reachability of those IP addresses using the Border Gateway Protocol. Typically, the human-readable names of servers are translated to IP addresses, transparently to users, via the directory function of the Domain Name System (DNS).

Over the Internet, there can be business-to-business (B2B), business-to-consumer (B2C) and consumer-to-consumer (C2C) communications. When money or sensitive information is exchanged, the communications are apt to be protected by some form of communications security mechanism. Intranets and extranets can be securely superimposed onto the Internet, without any access by general Internet users and administrators, using secure Virtual Private Network (VPN) technology.

=====================================================================================

**The Open Systems Interconnection (OSI)**model (ISO/IEC 7498-1) is a conceptual model that characterizes and standardizes the internal functions of a communication system by partitioning it into abstraction layers. The model is a product of the Open Systems Interconnection project at the International Organization for Standardization (ISO).

The model groups similar communication functions into one of seven logical layers. A layer serves the layer above it and is served by the layer below it. For example, a layer that provides error-free communications across a network provides the path needed by applications above it, while it calls the next lower layer to send and receive packets that make up the contents of that path. Two instances at one layer are connected by a horizontal connection on that layer.

History

Work on a layered model of network architecture was started and the[International Organization for Standardization](http://en.wikipedia.org/wiki/International_Organization_for_Standardization) (ISO) began to develop its OSI framework architecture. OSI had two major components: an *abstract model* of networking, called the Basic Reference Model or seven-layer model, and a set of specific protocols.

The concept of a seven-layer model was provided by the work of [Charles Bachman](http://en.wikipedia.org/wiki/Charles_Bachman), Honeywell Information Services. Various aspects of OSI design evolved from experiences with the [ARPANET](http://en.wikipedia.org/wiki/ARPANET), the fledgling Internet, NPLNET, EIN, [CYCLADES](http://en.wikipedia.org/wiki/CYCLADES) network and the work in IFIP WG6.1. The new design was documented in ISO 7498 and its various addenda. In this model, a networking system was divided into layers. Within each layer, one or more entities implement its functionality. Each entity interacted directly only with the layer immediately beneath it, and provided facilities for use by the layer above it.

Protocols enabled an entity in one host to interact with a corresponding entity at the same layer in another host. Service definitions abstractly described the functionality provided to an (N)-layer by an (N-1) layer, where N was one of the seven layers of protocols operating in the local host.

The OSI standards documents are available from the ITU-T as the X.200-series of recommendations.[[1]](http://en.wikipedia.org/wiki/Osi_model#cite_note-1) Some of the protocol specifications were also available as part of the ITU-T X series. The equivalent ISO and ISO/IEC standards for the OSI model were available from ISO, but only some of them without fees.[[2]](http://en.wikipedia.org/wiki/Osi_model#cite_note-2)

Description of OSI layers

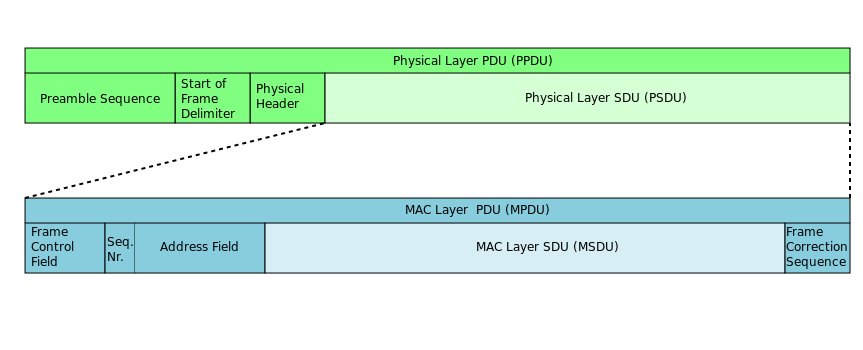
According to recommendation X.200, there are seven layers, labelled 1 to 7, with layer 1 at the bottom. Each layer is generically known as an N layer. An "N+1 entity" (at layer N+1) requests services from an "N entity" (at layer N).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OSI Model** | | | | |
|  | **Data unit** | **Layer** | **Function** |  |
| **Host layers** | [Data](http://en.wikipedia.org/wiki/Data_(computing)) | 7. [Application](http://en.wikipedia.org/wiki/Application_layer) | Network process to application |  |
| 6. [Presentation](http://en.wikipedia.org/wiki/Presentation_layer) | Data representation, encryption and decryption, convert machine dependent data to machine independent data |  |
| 5. [Session](http://en.wikipedia.org/wiki/Session_layer) | Interhost communication, managing sessions between applications |  |
| [Segments](http://en.wikipedia.org/wiki/Data_segment) | 4. [Transport](http://en.wikipedia.org/wiki/Transport_layer) | Reliable delivery of packets between points on a network. |  |
| **Media layers** | [Packet](http://en.wikipedia.org/wiki/Network_packet)/[Datagram](http://en.wikipedia.org/wiki/Datagram) | 3. [Network](http://en.wikipedia.org/wiki/Network_layer) | Addressing, routing and (not necessarily reliable) delivery of datagrams between points on a network. |  |
| [Bit](http://en.wikipedia.org/wiki/Bit)/[Frame](http://en.wikipedia.org/wiki/Frame_(networking)) | 2. [Data link](http://en.wikipedia.org/wiki/Data_link_layer) | A reliable direct point-to-point data connection. |  |
| [Bit](http://en.wikipedia.org/wiki/Bit) | 1. [Physical](http://en.wikipedia.org/wiki/Physical_layer) | A (not necessarily reliable) direct point-to-point data connection. |  |

At each level, two entities (N-entity peers) interact by means of the N protocol by transmitting [protocol data units](http://en.wikipedia.org/wiki/Protocol_data_unit)(PDU).

A [service data unit](http://en.wikipedia.org/wiki/Service_data_unit) (SDU) is a specific unit of data that has been passed down from an OSI layer to a lower layer, and which the lower layer has not yet encapsulated into a protocol data unit (PDU). An SDU is a set of data that is sent by a user of the services of a given layer, and is transmitted semantically unchanged to a peer service user.

The SDU at a layer N is the PDU of layer N+1. In effect the SDU is the 'payload' of a given PDU. That is, the process of changing an SDU to a PDU, consists of an encapsulation process, performed by the lower layer. All the data contained in the SDU becomes encapsulated within the PDU. The layer N-1 adds headers or footers, or both, to the SDU, transforming it into a PDU of layer N-1. The added headers or footers are part of the process used to make it possible to get data from a source to a destination.

[](http://en.wikipedia.org/wiki/File:Pdu_and_sdu.svg)

Some orthogonal aspects, such as management and security, involve every layer.

[Security services](http://en.wikipedia.org/wiki/Security_service_(telecommunication)) are not related to a specific layer: they can be related by a number of layers, as defined by [ITU-T](http://en.wikipedia.org/wiki/ITU-T) X.800 Recommendation.[[3]](http://en.wikipedia.org/wiki/Osi_model#cite_note-x800-3)

These services are aimed to improve the [CIA triad](http://en.wikipedia.org/wiki/Information_Security) ([confidentiality](http://en.wikipedia.org/wiki/Confidentiality), [integrity](http://en.wikipedia.org/wiki/Integrity), and [availability](http://en.wikipedia.org/wiki/Availability)) of transmitted data. In practice, the availability of communication service is determined by the interaction between [network design](http://en.wikipedia.org/wiki/Network_design)and [network management](http://en.wikipedia.org/wiki/Network_management) protocols. Appropriate choices for both of these are needed to protect against [denial of service](http://en.wikipedia.org/wiki/Denial_of_service).[[*citation needed*](http://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

**Layer 1: physical layer**

The [physical layer](http://en.wikipedia.org/wiki/Physical_layer) has the following major functions:

* it defines the [electrical](http://en.wikipedia.org/wiki/Electrical) and physical specifications of the data connection. It defines the relationship between a device and a physical [transmission medium](http://en.wikipedia.org/wiki/Transmission_medium) (e.g. a copper or [fiber optical cable](http://en.wikipedia.org/wiki/Optical_fiber_cable)). This includes the layout of[pins](http://en.wikipedia.org/wiki/Lead_(electronics)), [voltages](http://en.wikipedia.org/wiki/Voltage), line [impedance](http://en.wikipedia.org/wiki/Characteristic_impedance), [cable](http://en.wikipedia.org/wiki/Cable) [specifications](http://en.wikipedia.org/wiki/Specification), signal timing, [hubs](http://en.wikipedia.org/wiki/Network_hub), [repeaters](http://en.wikipedia.org/wiki/Repeater), [network adapters](http://en.wikipedia.org/wiki/Network_card), [host bus adapters](http://en.wikipedia.org/wiki/Host_adapter) (HBA used in [storage area networks](http://en.wikipedia.org/wiki/Storage_area_network)) and more.
* it defines the protocol to establish and terminate a [connection](http://en.wikipedia.org/wiki/Electrical_connector) between two directly connected nodes over a[communications](http://en.wikipedia.org/wiki/Communication) [medium](http://en.wikipedia.org/wiki/Transmission_medium).
* it may define the protocol for [flow control](http://en.wikipedia.org/wiki/Flow_control_(data)).
* it defines a protocol for the provision of a (not necessarily reliable) connection between two directly connected nodes, and the [Modulation](http://en.wikipedia.org/wiki/Modulation) or conversion between the representation of [digital data](http://en.wikipedia.org/wiki/Digital_data) in user equipment and the corresponding signals transmitted over the physical communications [channel](http://en.wikipedia.org/wiki/Channel_(communications)). This channel can involve physical cabling (such as copper and [optical fiber](http://en.wikipedia.org/wiki/Optical_fiber)) or a wireless [radio link](http://en.wikipedia.org/wiki/Electromagnetic_wave#Radio_waves).

The physical layer of [Parallel SCSI](http://en.wikipedia.org/wiki/Parallel_SCSI) operates in this layer, as do the physical layers of Ethernet and other local-area networks, such as [token ring](http://en.wikipedia.org/wiki/IBM_token_ring), [FDDI](http://en.wikipedia.org/wiki/Fiber_distributed_data_interface), [ITU-T](http://en.wikipedia.org/wiki/ITU-T) [G.hn](http://en.wikipedia.org/wiki/G.hn), and [IEEE 802.11](http://en.wikipedia.org/wiki/IEEE_802.11), as well as personal area networks such as [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth) and [IEEE 802.15.4](http://en.wikipedia.org/wiki/IEEE_802.15#Task_group_4_.28Low_Rate_WPAN.29).

**Layer 2: data link layer**

The [data link layer](http://en.wikipedia.org/wiki/Data_link_layer) provides a reliable link between two directly connected nodes, by detecting and possibly correcting errors that may occur in the physical layer.

[Point-to-Point Protocol](http://en.wikipedia.org/wiki/Point-to-Point_Protocol) (PPP) is an example of a data link layer in the TCP/IP protocol stack.

The [ITU-T](http://en.wikipedia.org/wiki/ITU-T) [G.hn](http://en.wikipedia.org/wiki/G.hn) standard, which provides high-speed local area networking over existing wires (power lines, phone lines and coaxial cables), includes a complete [data link layer](http://en.wikipedia.org/wiki/Data_link_layer) which provides both error correction and flow control by means of a [selective repeat](http://en.wikipedia.org/wiki/Selective_repeat) [Sliding Window Protocol](http://en.wikipedia.org/wiki/Sliding_Window_Protocol).

**Layer 3: network layer**

The [network layer](http://en.wikipedia.org/wiki/Network_layer) provides the functional and procedural means of transferring variable length [data](http://en.wikipedia.org/wiki/Data) sequences (called datagrams) from one node to another connected to the same *network.* A network is a medium to which many nodes can be connected, on which every node has an *address* and which permits nodes connected to it to transfer messages to other nodes connected to it by merely providing the content of a message and the address of the destination node and letting the network find the way to deliver ("route") the message to the destination node. In addition to message [routing](http://en.wikipedia.org/wiki/Routing), the network may (or may not) implement message delivery by splitting the message into several fragments, delivering each fragment by a separate route and reassembling the fragments, report delivery errors, etc.

Datagram delivery at the network layer is *not* guaranteed to be *reliable*.

A number of layer-management protocols, a function defined in the Management Annex, ISO 7498/4, belong to the network layer. These include routing protocols, multicast group management, network-layer information and error, and network-layer address assignment. It is the function of the payload that makes these belong to the network layer, not the protocol that carries them.

**Layer 4: transport layer**

The [transport layer](http://en.wikipedia.org/wiki/Transport_layer) provides the *reliable* sending of data packets between nodes (with addresses) located on a network, providing reliable data transfer services to the upper layers.

An example of a transport layer protocol in the standard Internet protocol stack is TCP, usually built on top of the IP protocol.

The transport layer controls the reliability of a given link through flow control, [segmentation/desegmentation](http://en.wikipedia.org/wiki/Packet_segmentation), and error control. Some protocols are state- and connection-oriented. This means that the transport layer can keep track of the segments and retransmit those that fail. The transport layer also provides the acknowledgement of the successful data transmission and sends the next data if no errors occurred. The transport layer creates packets out of the message received from the application layer. Packetizing is a process of dividing the long message into smaller messages.

OSI defines five classes of connection-mode transport protocols ranging from class 0 (which is also known as TP0 and provides the least features) to class 4 (TP4, designed for less reliable networks, similar to the Internet). Class 0 contains no error recovery, and was designed for use on network layers that provide error-free connections. Class 4 is closest to TCP, although TCP contains functions, such as the graceful close, which OSI assigns to the session layer. Also, all OSI TP connection-mode [protocol](http://en.wikipedia.org/wiki/Communications_protocol) classes provide expedited data and preservation of record boundaries. Detailed characteristics of TP0-4 classes are shown in the following table:[[4]](http://en.wikipedia.org/wiki/Osi_model#cite_note-4)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Feature Name** | **TP0** | **TP1** | **TP2** | **TP3** | **TP4** |
| Connection oriented network | Yes | Yes | Yes | Yes | Yes |
| Connectionless network | No | No | No | No | Yes |
| Concatenation and separation | No | Yes | Yes | Yes | Yes |
| Segmentation and reassembly | Yes | Yes | Yes | Yes | Yes |
| Error Recovery | No | Yes | Yes | Yes | Yes |
| Reinitiate connection (if an excessive number of [PDUs](http://en.wikipedia.org/wiki/Protocol_data_unit) are unacknowledged) | No | Yes | No | Yes | No |
| Multiplexing and demultiplexing over a single [virtual circuit](http://en.wikipedia.org/wiki/Virtual_circuit) | No | No | Yes | Yes | Yes |
| Explicit flow control | No | No | Yes | Yes | Yes |
| Retransmission on timeout | No | No | No | No | Yes |
| Reliable Transport Service | No | Yes | No | Yes | Yes |

An easy way to visualize the transport layer is to compare it with a Post Office, which deals with the dispatch and classification of mail and parcels sent. Do remember, however, that a post office manages the outer envelope of mail. Higher layers may have the equivalent of double envelopes, such as cryptographic presentation services that can be read by the addressee only. Roughly speaking, [tunneling protocols](http://en.wikipedia.org/wiki/Tunneling_protocol) operate at the transport layer, such as carrying non-IP protocols such as [IBM](http://en.wikipedia.org/wiki/IBM)'s [SNA](http://en.wikipedia.org/wiki/IBM_Systems_Network_Architecture) or [Novell](http://en.wikipedia.org/wiki/Novell)'s [IPX](http://en.wikipedia.org/wiki/Internetwork_Packet_Exchange) over an IP network, or end-to-end encryption with [IPsec](http://en.wikipedia.org/wiki/IPsec). While [Generic Routing Encapsulation](http://en.wikipedia.org/wiki/Generic_Routing_Encapsulation) (GRE) might seem to be a network-layer protocol, if the encapsulation of the payload takes place only at endpoint, GRE becomes closer to a transport protocol that uses IP headers but contains complete frames or packets to deliver to an endpoint. [L2TP](http://en.wikipedia.org/wiki/Layer_2_Tunneling_Protocol) carries [PPP](http://en.wikipedia.org/wiki/Point-to-Point_Protocol) frames inside transport packet.

Although not developed under the OSI Reference Model and not strictly conforming to the OSI definition of the transport layer, the [Transmission Control Protocol](http://en.wikipedia.org/wiki/Transmission_Control_Protocol) (TCP) and the [User Datagram Protocol](http://en.wikipedia.org/wiki/User_Datagram_Protocol) (UDP) of the Internet Protocol Suite are commonly categorized as layer-4 protocols within OSI.

**Layer 5: session layer**

The [session layer](http://en.wikipedia.org/wiki/Session_layer) controls the dialogues (connections) between computers. It establishes, manages and terminates the connections between the local and remote application. It provides for [full-duplex](http://en.wikipedia.org/wiki/Duplex_(telecommunications)), [half-duplex](http://en.wikipedia.org/wiki/Half-duplex), or[simplex](http://en.wikipedia.org/wiki/Simplex_communication) operation, and establishes checkpointing, adjournment, termination, and restart procedures. The OSI model made this layer responsible for graceful close of sessions, which is a property of the [Transmission Control Protocol](http://en.wikipedia.org/wiki/Transmission_Control_Protocol), and also for session checkpointing and recovery, which is not usually used in the Internet Protocol Suite. The session layer is commonly implemented explicitly in application environments that use[remote procedure calls](http://en.wikipedia.org/wiki/Remote_procedure_call).

**Layer 6: presentation layer**

The [presentation layer](http://en.wikipedia.org/wiki/Presentation_layer) establishes context between application-layer entities, in which the higher-layer entities may use different syntax and semantics if the presentation service provides a mapping between them. If a mapping is available, presentation service data units are encapsulated into session protocol data units, and passed down the stack.

This layer provides independence from data representation (e.g., [encryption](http://en.wikipedia.org/wiki/Encryption)) by translating between application and network formats. The presentation layer transforms data into the form that the application accepts. This layer formats and encrypts data to be sent across a network. It is sometimes called the syntax layer.[[5]](http://en.wikipedia.org/wiki/Osi_model#cite_note-5)

The original presentation structure used the [Basic Encoding Rules](http://en.wikipedia.org/wiki/Basic_Encoding_Rules) of [Abstract Syntax Notation One](http://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One) (ASN.1), with capabilities such as converting an [EBCDIC](http://en.wikipedia.org/wiki/EBCDIC)-coded text [file](http://en.wikipedia.org/wiki/Computer_file) to an [ASCII](http://en.wikipedia.org/wiki/ASCII)-coded file, or [serialization](http://en.wikipedia.org/wiki/Serialization) of [objects](http://en.wikipedia.org/wiki/Object_(computer_science)) and other [data structures](http://en.wikipedia.org/wiki/Data_structure) from and to [XML](http://en.wikipedia.org/wiki/XML).

**Layer 7: application layer**

The [application layer](http://en.wikipedia.org/wiki/Application_layer) is the OSI layer closest to the end user, which means that both the OSI application layer and the user interact directly with the software application. This layer interacts with software applications that implement a communicating component. Such application programs fall outside the scope of the OSI model. Application-layer functions typically include identifying communication partners, determining resource availability, and synchronizing communication. When identifying communication partners, the application layer determines the identity and availability of communication partners for an application with data to transmit. When determining resource availability, the application layer must decide whether sufficient network or the requested communication exists. In synchronizing communication, all communication between applications requires cooperation that is managed by the application layer. Some examples of application-layer implementations also include:

* On OSI stack:
  + [FTAM](http://en.wikipedia.org/wiki/FTAM) File Transfer and Access Management Protocol
  + [X.400](http://en.wikipedia.org/wiki/X.400) Mail
  + [Common Management Information Protocol](http://en.wikipedia.org/wiki/Common_Management_Information_Protocol) (CMIP)
* On TCP/IP stack:
  + [Hypertext Transfer Protocol](http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (HTTP),
  + [File Transfer Protocol](http://en.wikipedia.org/wiki/File_Transfer_Protocol) (FTP),
  + [Simple Mail Transfer Protocol](http://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol) (SMTP)
  + [Simple Network Management Protocol](http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol) (SNMP).

**Ethernet** /ˈiːθərnɛt/ is a family of computer networking technologies for local area networks (LANs). Ethernet was commercially introduced in 1980 and standardized in 1985 as IEEE 802.3. Ethernet has largely replaced competing wired LAN technologies such as token ring, FDDI, and ARCNET.

The Ethernet standards comprise several wiring and signaling variants of the OSI physical layer in use with Ethernet. The original 10BASE5 Ethernet used coaxial cable as a shared medium. Later the coaxial cables were replaced with twisted pair and fiber optic links in conjunction with hubs or switches. Data rates were periodically increased from the original 10 megabits per second to 100 gigabits per second.

Systems communicating over Ethernet divide a stream of data into shorter pieces called frames. Each frame contains source and destination addresses and error-checking data so that damaged data can be detected and re-transmitted. As per the OSI model Ethernet provides services up to and including the data link layer.

Since its commercial release, Ethernet has retained a good degree of compatibility. Features such as the 48-bit MAC address and Ethernet frame format have influenced other networking protocols.

**History**

An 8P8C modular connector (often called RJ45) commonly used on Cat 5 cables in Ethernet networks

Ethernet was developed at Xerox PARC between 1973 and 1974.[1][2] It was inspired by ALOHAnet, which Robert Metcalfe had studied as part of his PhD dissertation.[3] The idea was first documented in a memo that Metcalfe wrote on May 22, 1973, where he named it after the disproven luminiferous ether as an "omnipresent, completely-passive medium for the propagation of electromagnetic waves".[1][4][5] In 1975, Xerox filed a patent application listing Metcalfe, David Boggs, Chuck Thacker, and Butler Lampson as inventors.[6] In 1976, after the system was deployed at PARC, Metcalfe and Boggs published a seminal paper.[7][note 1]

Metcalfe left Xerox in June 1979 to form 3Com.[1][9] He convinced Digital Equipment Corporation (DEC), Intel, and Xerox to work together to promote Ethernet as a standard. The so-called "DIX" standard, for "Digital/Intel/Xerox", specified 10 Mbit/s Ethernet, with 48-bit destination and source addresses and a global 16-bit Ethertype-type field. It was published on September 30, 1980 as "The Ethernet, A Local Area Network. Data Link Layer and Physical Layer Specifications".[10] Version 2 was published in November, 1982[11] and defines what has become known as Ethernet II. Formal standardization efforts proceeded at the same time.

Ethernet initially competed with two largely proprietary systems, Token Ring and Token Bus. Because Ethernet was able to adapt to market realities and shift to inexpensive and ubiquitous twisted pair wiring, these proprietary protocols soon found themselves competing in a market inundated by Ethernet products, and, by the end of the 1980s, Ethernet was clearly the dominant network technology.[1] In the process, 3Com became a major company. 3Com shipped its first 10 Mbit/s Ethernet 3C100 transceiver in March 1981, and that year started selling adapters for PDP-11s and VAXes, as well as Multibus-based Intel and Sun Microsystems computers.[12]:9 This was followed quickly by DEC's Unibus to Ethernet adapter, which DEC sold and used internally to build its own corporate network, which reached over 10,000 nodes by 1986, making it one of the largest computer networks in the world at that time.[13] An Ethernet adapter card for the IBM PC was released in 1982, and, by 1985, 3Com had sold 100,000.[9]

Since then, Ethernet technology has evolved to meet new bandwidth and market requirements.[14] In addition to computers, Ethernet is now used to interconnect appliances and other personal devices.[1] It is used in industrial applications and is quickly replacing legacy data transmission systems in the world's telecommunications networks.[15] By 2010, the market for Ethernet equipment amounted to over $16 billion per year.[16]

**Standardization**

In February 1980, the Institute of Electrical and Electronics Engineers (IEEE) started project 802 to standardize local area networks (LAN).[9][17] The "DIX-group" with Gary Robinson (DEC), Phil Arst (Intel), and Bob Printis (Xerox) submitted the so-called "Blue Book" CSMA/CD specification as a candidate for the LAN specification.[10] In addition to CSMA/CD, Token Ring (supported by IBM) and Token Bus (selected and henceforward supported by General Motors) were also considered as candidates for a LAN standard. Competing proposals and broad interest in the initiative led to strong disagreement over which technology to standardize. In December 1980, the group was split into three subgroups, and standardization proceeded separately for each proposal.[9]

Delays in the standards process put at risk the market introduction of the Xerox Star workstation and 3Com's Ethernet LAN products. With such business implications in mind, David Liddle (General Manager, Xerox Office Systems) and Metcalfe (3Com) strongly supported a proposal of Fritz Röscheisen (Siemens Private Networks) for an alliance in the emerging office communication market, including Siemens' support for the international standardization of Ethernet (April 10, 1981). Ingrid Fromm, Siemens' representative to IEEE 802, quickly achieved broader support for Ethernet beyond IEEE by the establishment of a competing Task Group "Local Networks" within the European standards body ECMA TC24. As early as March 1982 ECMA TC24 with its corporate members reached agreement on a standard for CSMA/CD based on the IEEE 802 draft.[12]:8 Because the DIX proposal was most technically complete and because of the speedy action taken by ECMA which decisively contributed to the conciliation of opinions within IEEE, the IEEE 802.3 CSMA/CD standard was approved in December 1982.[9] IEEE published the 802.3 standard as a draft in 1983 and as a standard in 1985.[18]

Approval of Ethernet on the international level was achieved by a similar, cross-partisan action with Fromm as liaison officer working to integrate International Electrotechnical Commission, TC83 and International Organization for Standardization (ISO) TC97SC6, and the ISO/IEEE 802/3 standard was approved in 1984.

**Evolution**

Ethernet evolved to include higher bandwidth, improved media access control methods, and different physical media. The coaxial cable was replaced with point-to-point links connected by Ethernet repeaters or switches to reduce installation costs, increase reliability, and improve management and troubleshooting. Many variants of Ethernet remain in common use.

Ethernet stations communicate by sending each other data packets: blocks of data individually sent and delivered. As with other IEEE 802 LANs, each Ethernet station is given a 48-bit MAC address. The MAC addresses are used to specify both the destination and the source of each data packet. Ethernet establishes link level connections, which can be defined using both the destination and source addresses. On reception of a transmission, the receiver uses the destination address to determine whether the transmission is relevant to the station or should be ignored. Network interfaces normally do not accept packets addressed to other Ethernet stations. Adapters come programmed with a globally unique address.[note 2] An EtherType field in each frame is used by the operating system on the receiving station to select the appropriate protocol module (e.g., an Internet Protocol version such as IPv4). Ethernet frames are said to be self-identifying, because of the frame type. Self-identifying frames make it possible to intermix multiple protocols on the same physical network and allow a single computer to use multiple protocols together.[19] Despite the evolution of Ethernet technology, all generations of Ethernet (excluding early experimental versions) use the same frame formats[20] (and hence the same interface for higher layers), and can be readily interconnected through bridging.

Due to the ubiquity of Ethernet, the ever-decreasing cost of the hardware needed to support it, and the reduced panel space needed by twisted pair Ethernet, most manufacturers now build Ethernet interfaces directly into PC motherboards, eliminating the need for installation of a separate network card.[21]

Ethernet was originally based on the idea of computers communicating over a shared coaxial cable acting as a broadcast transmission medium. The methods used were similar to those used in radio systems,[note 3] with the common cable providing the communication channel likened to the Luminiferous aether in 19th century physics, and it was from this reference that the name "Ethernet" was derived.[22]

Original Ethernet's shared coaxial cable (the shared medium) traversed a building or campus to every attached machine. A scheme known as carrier sense multiple access with collision detection (CSMA/CD) governed the way the computers shared the channel. This scheme was simpler than the competing token ring or token bus technologies.[note 4] Computers were connected to an Attachment Unit Interface (AUI) transceiver, which was in turn connected to the cable (later with thin Ethernet the transceiver was integrated into the network adapter). While a simple passive wire was highly reliable for small networks, it was not reliable for large extended networks, where damage to the wire in a single place, or a single bad connector, could make the whole Ethernet segment unusable.[note 5]

Through the first half of the 1980s, Ethernet's 10BASE5 implementation used a coaxial cable 0.375 inches (9.5 mm) in diameter, later called "thick Ethernet" or "thicknet". Its successor, 10BASE2, called "thin Ethernet" or "thinnet", used a cable similar to cable television cable of the era. The emphasis was on making installation of the cable easier and less costly.

Since all communications happen on the same wire, any information sent by one computer is received by all, even if that information is intended for just one destination.[note 6] The network interface card interrupts the CPU only when applicable packets are received: The card ignores information not addressed to it.[note 7] Use of a single cable also means that the bandwidth is shared, such that, for example, available bandwidth to each device is halved when two stations are simultaneously active.

Collisions happen when two stations attempt to transmit at the same time. They corrupt transmitted data and require stations to retransmit. The lost data and retransmissions reduce throughput. In the worst case where multiple active hosts connected with maximum allowed cable length attempt to transmit many short frames, excessive collisions can reduce throughput dramatically. However, a Xerox report in 1980 studied performance of an existing Ethernet installation under both normal and artificially generated heavy load. The report claims that 98% throughput on the LAN was observed.[23] This is in contrast with token passing LANs (token ring, token bus), all of which suffer throughput degradation as each new node comes into the LAN, due to token waits. This report was controversial, as modeling showed that collision-based networks theoretically became unstable under loads as low as 37% of nominal capacity. Many early researchers failed to understand these results. Performance on real networks is significantly better.[24]

In a modern Ethernet, the stations do not all share one channel through a shared cable or a simple repeater hub; instead, each station communicates with a switch, which in turn forwards that traffic to the destination station. In this topology, collisions are only possible if station and switch attempt to communicate with each other at the same time, and collisions are limited to this link. Furthermore, the 10BASE-T standard introduced a full duplex mode of operation which has become extremely common. In full duplex, switch and station can communicate with each other simultaneously, and therefore modern Ethernets are completely collision-free.

**Repeaters and hubs**

A 1990s network interface card supporting both coaxial cable-based 10BASE2 (BNC connector, left) and twisted pair-based 10BASE-T (8P8C connector, right)

Main article: Ethernet hub

For signal degradation and timing reasons, coaxial Ethernet segments had a restricted size. Somewhat larger networks could be built by using an Ethernet repeater. Early repeaters had only two ports, allowing, at most, a doubling of network size. Once repeaters with more than two ports became available, it was possible to wire the network in a star topology. Early experiments with star topologies (called "Fibernet") using optical fiber were published by 1978.[25]

Shared cable Ethernet was always hard to install in offices because its bus topology was in conflict with the star topology cable plans designed into buildings for telephony. Modifying Ethernet to conform to twisted pair telephone wiring already installed in commercial buildings provided another opportunity to lower costs, expand the installed base, and leverage building design, and, thus, twisted-pair Ethernet was the next logical development in the mid-1980s.

Ethernet on unshielded twisted-pair cables (UTP) began with StarLAN at 1 Mbit/s in the mid-1980s. In 1987 SynOptics introduced the first twisted-pair Ethernet at 10 Mbit/s in a star-wired cabling topology with a central hub, later called LattisNet.[9][26][27] These evolved into 10BASE-T, which was designed for point-to-point links only, and all termination was built into the device. This changed repeaters from a specialist device used at the center of large networks to a device that every twisted pair-based network with more than two machines had to use. The tree structure that resulted from this made Ethernet networks easier to maintain by preventing most faults with one peer or its associated cable from affecting other devices on the network.

Despite the physical star topology and the presence of separate transmit and receive channels in the twisted pair and fiber media, repeater based Ethernet networks still use half-duplex and CSMA/CD, with only minimal activity by the repeater, primarily the Collision Enforcement signal, in dealing with packet collisions. Every packet is sent to every port on the repeater, so bandwidth and security problems are not addressed. The total throughput of the repeater is limited to that of a single link, and all links must operate at the same speed.

**Bridging and switching**

While repeaters could isolate some aspects of Ethernet segments, such as cable breakages, they still forwarded all traffic to all Ethernet devices. This created practical limits on how many machines could communicate on an Ethernet network. The entire network was one collision domain, and all hosts had to be able to detect collisions anywhere on the network. This limited the number of repeaters between the farthest nodes. Segments joined by repeaters had to all operate at the same speed, making phased-in upgrades impossible.

To alleviate these problems, bridging was created to communicate at the data link layer while isolating the physical layer. With bridging, only well-formed Ethernet packets are forwarded from one Ethernet segment to another; collisions and packet errors are isolated. At initial startup, Ethernet bridges (and switches) work somewhat like Ethernet repeaters, passing all traffic between segments. By observing the source addresses of incoming frames, the bridge then builds an address table associating addresses to segments. Once an address is learned, the bridge forwards network traffic destined for that address only to the associated segment, improving overall performance. Broadcast traffic is still forwarded to all network segments. Bridges also overcame the limits on total segments between two hosts and allowed the mixing of speeds, both of which are critical to deployment of Fast Ethernet.

In 1989, the networking company Kalpana introduced their EtherSwitch, the first Ethernet switch.[note 8] This worked somewhat differently from an Ethernet bridge, where only the header of the incoming packet would be examined before it was either dropped or forwarded to another segment. This greatly reduced the forwarding latency and the processing load on the network device. One drawback of this cut-through switching method was that packets that had been corrupted would still be propagated through the network, so a jabbering station could continue to disrupt the entire network. The eventual remedy for this was a return to the original store and forward approach of bridging, where the packet would be read into a buffer on the switch in its entirety, verified against its checksum and then forwarded, but using more powerful application-specific integrated circuits. Hence, the bridging is then done in hardware, allowing packets to be forwarded at full wire speed.

When a twisted pair or fiber link segment is used and neither end is connected to a repeater, full-duplex Ethernet becomes possible over that segment. In full-duplex mode, both devices can transmit and receive to and from each other at the same time, and there is no collision domain. This doubles the aggregate bandwidth of the link and is sometimes advertised as double the link speed (e.g., 200 Mbit/s).[note 9] The elimination of the collision domain for these connections also means that all the link's bandwidth can be used by the two devices on that segment and that segment length is not limited by the need for correct collision detection.

Since packets are typically delivered only to the port they are intended for, traffic on a switched Ethernet is less public than on shared-medium Ethernet. Despite this, switched Ethernet should still be regarded as an insecure network technology, because it is easy to subvert switched Ethernet systems by means such as ARP spoofing and MAC flooding.

The bandwidth advantages, the improved isolation of devices from each other, the ability to easily mix different speeds of devices and the elimination of the chaining limits inherent in non-switched Ethernet have made switched Ethernet the dominant network technology.

**Advanced networking**

Simple switched Ethernet networks, while a great improvement over repeater-based Ethernet, suffer from single points of failure, attacks that trick switches or hosts into sending data to a machine even if it is not intended for it, scalability and security issues with regard to broadcast radiation and multicast traffic, and bandwidth choke points where a lot of traffic is forced down a single link.[citation needed]

Advanced networking features in switches and routers combat these issues through means including spanning-tree protocol to maintain the active links of the network as a tree while allowing physical loops for redundancy, port security and protection features such as MAC lock down and broadcast radiation filtering, virtual LANs to keep different classes of users separate while using the same physical infrastructure, multilayer switching to route between different classes and link aggregation to add bandwidth to overloaded links and to provide some measure of redundancy.

IEEE 802.1aq (shortest path bridging) includes the use of the link-state routing protocol IS-IS to allow larger networks with shortest path routes between devices. In 2012 it was stated by David Allan and Nigel Bragg, in 802.1aq Shortest Path Bridging Design and Evolution: The Architect's Perspective that shortest path bridging is one of the most significant enhancements in Ethernet's history.

**Varieties of Ethernet**

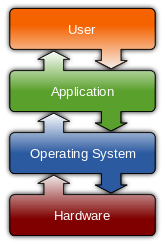
The Ethernet physical layer evolved over a considerable time span and encompasses coaxial, twisted pair and fiber optic physical media interfaces and speeds from 10 Mbit to 100 Gbit. The most common forms used are 10BASE-T, 100BASE-TX, and 1000BASE-T. All three utilize twisted pair cables and 8P8C modular connectors. They run at 10 Mbit/s, 100 Mbit/s, and 1 Gbit/s, respectively. Fiber optic variants of Ethernet offer high performance, electrical isolation and distance (tens of kilometers with some versions). In general, network protocol stack software will work similarly on all varieties.

**Autonegotiation**

Autonegotiation is the procedure by which two connected devices choose common transmission parameters, e.g. speed and duplex mode. Autonegotiation was an optional feature on first introduction of 100BASE-TX, while it is also backward compatible with 10BASE-T. Autonegotiation is mandatory for 1000BASE-T.

=========================================================================

**Operating system**(OS) is a collection of software that manages computer hardware resources and provides common services for computer programs. The operating system is an essential component of the system software in a computer system. Application programs usually require an operating system to function.

Time-sharing operating systems schedule tasks for efficient use of the system and may also include accounting software for cost allocation of processor time, mass storage, printing, and other resources.

For hardware functions such as input and output and memory allocation, the operating system acts as an intermediary between programs and the computer hardware,[1][2] although the application code is usually executed directly by the hardware and will frequently make a system call to an OS function or be interrupted by it. Operating systems can be found on almost any device that contains a computer—from cellular phones and video game consoles to supercomputers and web servers.

Examples of popular modern operating systems include Android, BSD, iOS, Linux, OS X, QNX, Microsoft Windows,[3] Windows Phone, and IBM z/OS. All these, except Windows, Windows Phone and z/OS, share roots in UNIX.

**Types of operating systems**

**Real-time**

A real-time operating system is a multitasking operating system that aims at executing real-time applications. Real-time operating systems often use specialized scheduling algorithms so that they can achieve a deterministic nature of behavior. The main objective of real-time operating systems is their quick and predictable response to events. They have an event-driven or time-sharing design and often aspects of both. An event-driven system switches between tasks based on their priorities or external events while time-sharing operating systems switch tasks based on clock interrupts.

**Multi-user**

A multi-user operating system allows multiple users to access a computer system at the same time. Time-sharing systems and Internet servers can be classified as multi-user systems as they enable multiple-user access to a computer through the sharing of time. Single-user operating systems have only one user but may allow multiple programs to run at the same time.

**Multi-tasking vs. single-tasking**

A multi-tasking operating system allows more than one program to be running at the same time, from the point of view of human time scales. A single-tasking system has only one running program. Multi-tasking can be of two types: pre-emptive and co-operative. In pre-emptive multitasking, the operating system slices the CPU time and dedicates one slot to each of the programs. Unix-like operating systems such as Solaris and Linux support pre-emptive multitasking, as does AmigaOS. Cooperative multitasking is achieved by relying on each process to give time to the other processes in a defined manner. 16-bit versions of Microsoft Windows used cooperative multi-tasking. 32-bit versions of both Windows NT and Win9x, used pre-emptive multi-tasking. Mac OS prior to OS X used to support cooperative multitasking.

**Distributed**

A distributed operating system manages a group of independent computers and makes them appear to be a single computer. The development of networked computers that could be linked and communicate with each other gave rise to distributed computing. Distributed computations are carried out on more than one machine. When computers in a group work in cooperation, they make a distributed system.

**Embedded**

Embedded operating systems are designed to be used in embedded computer systems. They are designed to operate on small machines like PDAs with less autonomy. They are able to operate with a limited number of resources. They are very compact and extremely efficient by design. Windows CE and Minix 3 are some examples of embedded operating systems.

**History**

Early computers were built to perform a series of single tasks, like a calculator. Basic operating system features were developed in the 1950s, such as resident monitor functions that could automatically run different programs in succession to speed up processing. Operating systems did not exist in their modern and more complex forms until the early 1960s.[4] Hardware features were added, that enabled use of runtime libraries, interrupts, and parallel processing. When personal computers became popular in the 1980s, operating systems were made for them similar in concept to those used on larger computers.

In the 1940s, the earliest electronic digital systems had no operating systems. Electronic systems of this time were programmed on rows of mechanical switches or by jumper wires on plug boards. These were special-purpose systems that, for example, generated ballistics tables for the military or controlled the printing of payroll checks from data on punched paper cards. After programmable general purpose computers were invented, machine languages (consisting of strings of the binary digits 0 and 1 on punched paper tape) were introduced that sped up the programming process (Stern, 1981).

In the early 1950s, a computer could execute only one program at a time. Each user had sole use of the computer for a limited period of time and would arrive at a scheduled time with program and data on punched paper cards and/or punched tape. The program would be loaded into the machine, and the machine would be set to work until the program completed or crashed. Programs could generally be debugged via a front panel using toggle switches and panel lights. It is said that Alan Turing was a master of this on the early Manchester Mark 1 machine, and he was already deriving the primitive conception of an operating system from the principles of the Universal Turing machine.[4]

Later machines came with libraries of programs, which would be linked to a user's program to assist in operations such as input and output and generating computer code from human-readable symbolic code. This was the genesis of the modern-day operating system. However, machines still ran a single job at a time. At Cambridge University in England the job queue was at one time a washing line from which tapes were hung with different colored clothes-pegs to indicate job-priority.[citation needed]

**Mainframes**

Through the 1950s, many major features were pioneered in the field of operating systems, including batch processing, input/output interrupt, buffering, multitasking, spooling, runtime libraries, link-loading, and programs for sorting records in files. These features were included or not included in application software at the option of application programmers, rather than in a separate operating system used by all applications. In 1959 the SHARE Operating System was released as an integrated utility for the IBM 704, and later in the 709 and 7090 mainframes, although it was quickly supplanted by IBSYS/IBJOB on the 709, 7090 and 7094.

During the 1960s, IBM's OS/360 introduced the concept of a single OS spanning an entire product line, which was crucial for the success of the System/360 machines. IBM's current mainframe operating systems are distant descendants of this original system and applications written for OS/360 can still be run on modern machines.[citation needed]

OS/360 also pioneered the concept that the operating system keeps track of all of the system resources that are used, including program and data space allocation in main memory and file space in secondary storage, and file locking during update. When the process is terminated for any reason, all of these resources are re-claimed by the operating system.

The alternative CP-67 system for the S/360-67 started a whole line of IBM operating systems focused on the concept of virtual machines. Other operating systems used on IBM S/360 series mainframes included systems developed by IBM: COS/360 (Compatibility Operating System), DOS/360 (Disk Operating System), TSS/360 (Time Sharing System), TOS/360 (Tape Operating System), BOS/360 (Basic Operating System), and ACP (Airline Control Program), as well as a few non-IBM systems: MTS (Michigan Terminal System), MUSIC (Multi-User System for Interactive Computing), and ORVYL (Stanford Timesharing System).

Control Data Corporation developed the SCOPE operating system in the 1960s, for batch processing. In cooperation with the University of Minnesota, the Kronos and later the NOS operating systems were developed during the 1970s, which supported simultaneous batch and timesharing use. Like many commercial timesharing systems, its interface was an extension of the Dartmouth BASIC operating systems, one of the pioneering efforts in timesharing and programming languages. In the late 1970s, Control Data and the University of Illinois developed the PLATO operating system, which used plasma panel displays and long-distance time sharing networks. Plato was remarkably innovative for its time, featuring real-time chat, and multi-user graphical games.

In 1961, Burroughs Corporation introduced the B5000 with the MCP, (Master Control Program) operating system. The B5000 was a stack machine designed to exclusively support high-level languages with no machine language or assembler, and indeed the MCP was the first OS to be written exclusively in a high-level language – ESPOL, a dialect of ALGOL. MCP also introduced many other ground-breaking innovations, such as being the first commercial implementation of virtual memory. During development of the AS400, IBM made an approach to Burroughs to licence MCP to run on the AS400 hardware. This proposal was declined by Burroughs management to protect its existing hardware production. MCP is still in use today in the Unisys ClearPath/MCP line of computers.

UNIVAC, the first commercial computer manufacturer, produced a series of EXEC operating systems. Like all early main-frame systems, this batch-oriented system managed magnetic drums, disks, card readers and line printers. In the 1970s, UNIVAC produced the Real-Time Basic (RTB) system to support large-scale time sharing, also patterned after the Dartmouth BC system.

General Electric and MIT developed General Electric Comprehensive Operating Supervisor (GECOS), which introduced the concept of ringed security privilege levels. After acquisition by Honeywell it was renamed General Comprehensive Operating System (GCOS).

Digital Equipment Corporation developed many operating systems for its various computer lines, including TOPS-10 and TOPS-20 time sharing systems for the 36-bit PDP-10 class systems. Prior to the widespread use of UNIX, TOPS-10 was a particularly popular system in universities, and in the early ARPANET community.

From the late 1960s through the late 1970s, several hardware capabilities evolved that allowed similar or ported software to run on more than one system. Early systems had utilized microprogramming to implement features on their systems in order to permit different underlying computer architectures to appear to be the same as others in a series. In fact, most 360s after the 360/40 (except the 360/165 and 360/168) were microprogrammed implementations.

The enormous investment in software for these systems made since the 1960s caused most of the original computer manufacturers to continue to develop compatible operating systems along with the hardware. Notable supported mainframe operating systems include:

Burroughs MCP – B5000, 1961 to Unisys Clearpath/MCP, present.

IBM OS/360 – IBM System/360, 1966 to IBM z/OS, present.

IBM CP-67 – IBM System/360, 1967 to IBM z/VM, present.

UNIVAC EXEC 8 – UNIVAC 1108, 1967, to OS 2200 Unisys Clearpath Dorado, present.

**Microcomputers**

Mac OS by Apple Computer became the first widespread OS to feature a graphical user interface. Many of its features such as windows and icons would later become commonplace in GUIs.

The first microcomputers did not have the capacity or need for the elaborate operating systems that had been developed for mainframes and minis; minimalistic operating systems were developed, often loaded from ROM and known as monitors. One notable early disk operating system was CP/M, which was supported on many early microcomputers and was closely imitated by Microsoft's MS-DOS, which became wildly popular as the operating system chosen for the IBM PC (IBM's version of it was called IBM DOS or PC DOS). In the '80s, Apple Computer Inc. (now Apple Inc.) abandoned its popular Apple II series of microcomputers to introduce the Apple Macintosh computer with an innovative Graphical User Interface (GUI) to the Mac OS.

The introduction of the Intel 80386 CPU chip with 32-bit architecture and paging capabilities, provided personal computers with the ability to run multitasking operating systems like those of earlier minicomputers and mainframes. Microsoft responded to this progress by hiring Dave Cutler, who had developed the VMS operating system for Digital Equipment Corporation. He would lead the development of the Windows NT operating system, which continues to serve as the basis for Microsoft's operating systems line. Steve Jobs, a co-founder of Apple Inc., started NeXT Computer Inc., which developed the NEXTSTEP operating system. NEXTSTEP would later be acquired by Apple Inc. and used, along with code from FreeBSD as the core of Mac OS X.

The GNU Project was started by activist and programmer Richard Stallman with the goal of creating a complete free software replacement to the proprietary UNIX operating system. While the project was highly successful in duplicating the functionality of various parts of UNIX, development of the GNU Hurd kernel proved to be unproductive. In 1991, Finnish computer science student Linus Torvalds, with cooperation from volunteers collaborating over the Internet, released the first version of the Linux kernel. It was soon merged with the GNU user space components and system software to form a complete operating system. Since then, the combination of the two major components has usually been referred to as simply "Linux" by the software industry, a naming convention that Stallman and the Free Software Foundation remain opposed to, preferring the name GNU/Linux. The Berkeley Software Distribution, known as BSD, is the UNIX derivative distributed by the University of California, Berkeley, starting in the 1970s. Freely distributed and ported to many minicomputers, it eventually also gained a following for use on PCs, mainly as FreeBSD, NetBSD and OpenBSD.

**Examples of operating systems**

**UNIX and UNIX-like operating systems**

**Evolution of Unix systems**

Unix was originally written in assembly language.[5] Ken Thompson wrote B, mainly based on BCPL, based on his experience in the MULTICS project. B was replaced by C, and Unix, rewritten in C, developed into a large, complex family of inter-related operating systems which have been influential in every modern operating system (see History).

The UNIX-like family is a diverse group of operating systems, with several major sub-categories including System V, BSD, and Linux. The name "UNIX" is a trademark of The Open Group which licenses it for use with any operating system that has been shown to conform to their definitions. "UNIX-like" is commonly used to refer to the large set of operating systems which resemble the original UNIX.

Unix-like systems run on a wide variety of computer architectures. They are used heavily for servers in business, as well as workstations in academic and engineering environments. Free UNIX variants, such as Linux and BSD, are popular in these areas.

Four operating systems are certified by the The Open Group (holder of the Unix trademark) as Unix. HP's HP-UX and IBM's AIX are both descendants of the original System V Unix and are designed to run only on their respective vendor's hardware. In contrast, Sun Microsystems's Solaris Operating System can run on multiple types of hardware, including x86 and Sparc servers, and PCs. Apple's OS X, a replacement for Apple's earlier (non-Unix) Mac OS, is a hybrid kernel-based BSD variant derived from NeXTSTEP, Mach, and FreeBSD.

Unix interoperability was sought by establishing the POSIX standard. The POSIX standard can be applied to any operating system, although it was originally created for various Unix variants.

**BSD and its descendants**

A subgroup of the Unix family is the Berkeley Software Distribution family, which includes FreeBSD, NetBSD, and OpenBSD. These operating systems are most commonly found on webservers, although they can also function as a personal computer OS. The Internet owes much of its existence to BSD, as many of the protocols now commonly used by computers to connect, send and receive data over a network were widely implemented and refined in BSD. The world wide web was also first demonstrated on a number of computers running an OS based on BSD called NextStep.

BSD has its roots in Unix. In 1974, University of California, Berkeley installed its first Unix system. Over time, students and staff in the computer science department there began adding new programs to make things easier, such as text editors. When Berkely received new VAX computers in 1978 with Unix installed, the school's undergraduates modified Unix even more in order to take advantage of the computer's hardware possibilities. The Defense Advanced Research Projects Agency of the US Department of Defense took interest, and decided to fund the project. Many schools, corporations, and government organizations took notice and started to use Berkeley's version of Unix instead of the official one distributed by AT&T.

Steve Jobs, upon leaving Apple Inc. in 1985, formed NeXT Inc., a company that manufactured high-end computers running on a variation of BSD called NeXTSTEP. One of these computers was used by Tim Berners-Lee as the first webserver to create the World Wide Web.

Developers like Keith Bostic encouraged the project to replace any non-free code that originated with Bell Labs. Once this was done, however, AT&T sued. Eventually, after two years of legal disputes, the BSD project came out ahead and spawned a number of free derivatives, such as FreeBSD and NetBSD.

**OS X**

OS X (formerly "Mac OS X") is a line of open core graphical operating systems developed, marketed, and sold by Apple Inc., the latest of which is pre-loaded on all currently shipping Macintosh computers. OS X is the successor to the original Mac OS, which had been Apple's primary operating system since 1984. Unlike its predecessor, OS X is a UNIX operating system built on technology that had been developed at NeXT through the second half of the 1980s and up until Apple purchased the company in early 1997. The operating system was first released in 1999 as Mac OS X Server 1.0, with a desktop-oriented version (Mac OS X v10.0 "Cheetah") following in March 2001. Since then, six more distinct "client" and "server" editions of OS X have been released, until the two were merged in OS X 10.7 "Lion". The most recent version is OS X 10.9 "Mavericks", which was announced on June 10, 2013, and released on October 22, 2013. Releases of OS X v10.0 through v10.8 are named after big cats. Starting with v10.9, "Mavericks", OS X versions are named after inspirational places in California.[6]

Prior to its merging with OS X, the server edition – OS X Server – was architecturally identical to its desktop counterpart and usually ran on Apple's line of Macintosh server hardware. OS X Server included work group management and administration software tools that provide simplified access to key network services, including a mail transfer agent, a Samba server, an LDAP server, a domain name server, and others. With Mac OS X v10.7 Lion, all server aspects of Mac OS X Server have been integrated into the client version and the product re-branded as "OS X" (dropping "Mac" from the name). The server tools are now offered as an application.

**Linux and GNU**

Linux (or GNU/Linux) is a Unix-like operating system that was developed without any actual Unix code, unlike BSD and its variants. Linux can be used on a wide range of devices from supercomputers to wristwatches. The Linux kernel is released under an open source license, so anyone can read and modify its code. It has been modified to run on a large variety of electronics. Although estimates suggest that Linux is used on 1.82% of all personal computers,[8][9] it has been widely adopted for use in servers[10] and embedded systems[11] (such as cell phones). Linux has superseded Unix in most places[which?], and is used on the 10 most powerful supercomputers in the world.[12] The Linux kernel is used in some popular distributions, such as Red Hat, Debian, Ubuntu, Linux Mint and Google's Android.

The GNU project is a mass collaboration of programmers who seek to create a completely free and open operating system that was similar to Unix but with completely original code. It was started in 1983 by Richard Stallman, and is responsible for many of the parts of most Linux variants. Thousands of pieces of software for virtually every operating system are licensed under the GNU General Public License. Meanwhile, the Linux kernel began as a side project of Linus Torvalds, a university student from Finland. In 1991, Torvalds began work on it, and posted information about his project on a newsgroup for computer students and programmers. He received a wave of support and volunteers who ended up creating a full-fledged kernel. Programmers from GNU took notice, and members of both projects worked to integrate the finished GNU parts with the Linux kernel in order to create a full-fledged operating system.

**Google Chromium OS**

Chromium is an operating system based on the Linux kernel and designed by Google. Since Chromium OS targets computer users who spend most of their time on the Internet, it is mainly a web browser with limited ability to run local applications, though it has a built-in file manager and media player. Instead, it relies on Internet applications (or Web apps) used in the web browser to accomplish tasks such as word processing.[13]

**Microsoft Windows**

Microsoft Windows is a family of proprietary operating systems designed by Microsoft Corporation and primarily targeted to Intel architecture based computers, with an estimated 88.9 percent total usage share on Web connected computers.[9][14][15][16] The newest version is Windows 8 for workstations and Windows Server 2012 for servers. Windows 7 recently overtook Windows XP as most used OS.[17][18][19]

Microsoft Windows originated in 1985 as an operating environment running on top of MS-DOS, which was the standard operating system shipped on most Intel architecture personal computers at the time. In 1995, Windows 95 was released which only used MS-DOS as a bootstrap. For backwards compatibility, Win9x could run real-mode MS-DOS[20][21] and 16 bits Windows 3.x[22] drivers. Windows ME, released in 2000, was the last version in the Win9x family. Later versions have all been based on the Windows NT kernel. Current versions of Windows run on IA-32 and x86-64 microprocessors, although Windows 8 will support ARM architecture.[23] In the past, Windows NT supported non-Intel architectures.

Server editions of Windows are widely used. In recent years, Microsoft has expended significant capital in an effort to promote the use of Windows as a server operating system. However, Windows' usage on servers is not as widespread as on personal computers, as Windows competes against Linux and BSD for server market share.[24][25]

**Other**

There have been many operating systems that were significant in their day but are no longer so, such as AmigaOS; OS/2 from IBM and Microsoft; Mac OS, the non-Unix precursor to Apple's Mac OS X; BeOS; XTS-300; RISC OS; MorphOS and FreeMint. Some are still used in niche markets and continue to be developed as minority platforms for enthusiast communities and specialist applications. OpenVMS formerly from DEC, is still under active development by Hewlett-Packard. Yet other operating systems are used almost exclusively in academia, for operating systems education or to do research on operating system concepts. A typical example of a system that fulfills both roles is MINIX, while for example Singularity is used purely for research.

Other operating systems have failed to win significant market share, but have introduced innovations that have influenced mainstream operating systems, not least Bell Labs' Plan 9.

**Components**

The components of an operating system all exist in order to make the different parts of a computer work together. All user software needs to go through the operating system in order to use any of the hardware, whether it be as simple as a mouse or keyboard or as complex as an Internet component.

**Kernel**

A kernel connects the application software to the hardware of a computer.

With the aid of the firmware and device drivers, the kernel provides the most basic level of control over all of the computer's hardware devices. It manages memory access for programs in the RAM, it determines which programs get access to which hardware resources, it sets up or resets the CPU's operating states for optimal operation at all times, and it organizes the data for long-term non-volatile storage with file systems on such media as disks, tapes, flash memory, etc.

**Program execution**

The operating system provides an interface between an application program and the computer hardware, so that an application program can interact with the hardware only by obeying rules and procedures programmed into the operating system. The operating system is also a set of services which simplify development and execution of application programs. Executing an application program involves the creation of a process by the operating system kernel which assigns memory space and other resources, establishes a priority for the process in multi-tasking systems, loads program binary code into memory, and initiates execution of the application program which then interacts with the user and with hardware devices.

**Interrupts**

Interrupts are central to operating systems, as they provide an efficient way for the operating system to interact with and react to its environment. The alternative — having the operating system "watch" the various sources of input for events (polling) that require action — can be found in older systems with very small stacks (50 or 60 bytes) but are unusual in modern systems with large stacks. Interrupt-based programming is directly supported by most modern CPUs. Interrupts provide a computer with a way of automatically saving local register contexts, and running specific code in response to events. Even very basic computers support hardware interrupts, and allow the programmer to specify code which may be run when that event takes place.

When an interrupt is received, the computer's hardware automatically suspends whatever program is currently running, saves its status, and runs computer code previously associated with the interrupt; this is analogous to placing a bookmark in a book in response to a phone call. In modern operating systems, interrupts are handled by the operating system's kernel. Interrupts may come from either the computer's hardware or from the running program.

When a hardware device triggers an interrupt, the operating system's kernel decides how to deal with this event, generally by running some processing code. The amount of code being run depends on the priority of the interrupt (for example: a person usually responds to a smoke detector alarm before answering the phone). The processing of hardware interrupts is a task that is usually delegated to software called device driver, which may be either part of the operating system's kernel, part of another program, or both. Device drivers may then relay information to a running program by various means.

A program may also trigger an interrupt to the operating system. If a program wishes to access hardware for example, it may interrupt the operating system's kernel, which causes control to be passed back to the kernel. The kernel will then process the request. If a program wishes additional resources (or wishes to shed resources) such as memory, it will trigger an interrupt to get the kernel's attention.

**Modes**

Privilege rings for the x86 available in protected mode. Operating systems determine which processes run in each mode.

Modern CPUs support multiple modes of operation. CPUs with this capability use at least two modes: protected mode and supervisor mode. The supervisor mode is used by the operating system's kernel for low level tasks that need unrestricted access to hardware, such as controlling how memory is written and erased, and communication with devices like graphics cards. Protected mode, in contrast, is used for almost everything else. Applications operate within protected mode, and can only use hardware by communicating with the kernel, which controls everything in supervisor mode. CPUs might have other modes similar to protected mode as well, such as the virtual modes in order to emulate older processor types, such as 16-bit processors on a 32-bit one, or 32-bit processors on a 64-bit one.

When a computer first starts up, it is automatically running in supervisor mode. The first few programs to run on the computer, being the BIOS or EFI, bootloader, and the operating system have unlimited access to hardware – and this is required because, by definition, initializing a protected environment can only be done outside of one. However, when the operating system passes control to another program, it can place the CPU into protected mode.

In protected mode, programs may have access to a more limited set of the CPU's instructions. A user program may leave protected mode only by triggering an interrupt, causing control to be passed back to the kernel. In this way the operating system can maintain exclusive control over things like access to hardware and memory.

The term "protected mode resource" generally refers to one or more CPU registers, which contain information that the running program isn't allowed to alter. Attempts to alter these resources generally causes a switch to supervisor mode, where the operating system can deal with the illegal operation the program was attempting (for example, by killing the program).

**Memory management**

Among other things, a multiprogramming operating system kernel must be responsible for managing all system memory which is currently in use by programs. This ensures that a program does not interfere with memory already in use by another program. Since programs time share, each program must have independent access to memory.

Cooperative memory management, used by many early operating systems, assumes that all programs make voluntary use of the kernel's memory manager, and do not exceed their allocated memory. This system of memory management is almost never seen any more, since programs often contain bugs which can cause them to exceed their allocated memory. If a program fails, it may cause memory used by one or more other programs to be affected or overwritten. Malicious programs or viruses may purposefully alter another program's memory, or may affect the operation of the operating system itself. With cooperative memory management, it takes only one misbehaved program to crash the system.

Memory protection enables the kernel to limit a process' access to the computer's memory. Various methods of memory protection exist, including memory segmentation and paging. All methods require some level of hardware support (such as the 80286 MMU), which doesn't exist in all computers.

In both segmentation and paging, certain protected mode registers specify to the CPU what memory address it should allow a running program to access. Attempts to access other addresses will trigger an interrupt which will cause the CPU to re-enter supervisor mode, placing the kernel in charge. This is called a segmentation violation or Seg-V for short, and since it is both difficult to assign a meaningful result to such an operation, and because it is usually a sign of a misbehaving program, the kernel will generally resort to terminating the offending program, and will report the error.

Windows 3.1-Me had some level of memory protection, but programs could easily circumvent the need to use it. A general protection fault would be produced, indicating a segmentation violation had occurred; however, the system would often crash anyway.

**Virtual memory**

Many operating systems can "trick" programs into using memory scattered around the hard disk and RAM as if it is one continuous chunk of memory, called virtual memory.

The use of virtual memory addressing (such as paging or segmentation) means that the kernel can choose what memory each program may use at any given time, allowing the operating system to use the same memory locations for multiple tasks.

If a program tries to access memory that isn't in its current range of accessible memory, but nonetheless has been allocated to it, the kernel will be interrupted in the same way as it would if the program were to exceed its allocated memory. (See section on memory management.) Under UNIX this kind of interrupt is referred to as a page fault.

When the kernel detects a page fault it will generally adjust the virtual memory range of the program which triggered it, granting it access to the memory requested. This gives the kernel discretionary power over where a particular application's memory is stored, or even whether or not it has actually been allocated yet.

In modern operating systems, memory which is accessed less frequently can be temporarily stored on disk or other media to make that space available for use by other programs. This is called swapping, as an area of memory can be used by multiple programs, and what that memory area contains can be swapped or exchanged on demand.

"Virtual memory" provides the programmer or the user with the perception that there is a much larger amount of RAM in the computer than is really there.

**Multitasking**

Multitasking refers to the running of multiple independent computer programs on the same computer; giving the appearance that it is performing the tasks at the same time. Since most computers can do at most one or two things at one time, this is generally done via time-sharing, which means that each program uses a share of the computer's time to execute.

An operating system kernel contains a piece of software called a scheduler which determines how much time each program will spend executing, and in which order execution control should be passed to programs. Control is passed to a process by the kernel, which allows the program access to the CPU and memory. Later, control is returned to the kernel through some mechanism, so that another program may be allowed to use the CPU. This so-called passing of control between the kernel and applications is called a context switch.

An early model which governed the allocation of time to programs was called cooperative multitasking. In this model, when control is passed to a program by the kernel, it may execute for as long as it wants before explicitly returning control to the kernel. This means that a malicious or malfunctioning program may not only prevent any other programs from using the CPU, but it can hang the entire system if it enters an infinite loop.

Modern operating systems extend the concepts of application preemption to device drivers and kernel code, so that the operating system has preemptive control over internal run-times as well.

The philosophy governing preemptive multitasking is that of ensuring that all programs are given regular time on the CPU. This implies that all programs must be limited in how much time they are allowed to spend on the CPU without being interrupted. To accomplish this, modern operating system kernels make use of a timed interrupt. A protected mode timer is set by the kernel which triggers a return to supervisor mode after the specified time has elapsed. (See above sections on Interrupts and Dual Mode Operation.)

On many single user operating systems cooperative multitasking is perfectly adequate, as home computers generally run a small number of well tested programs. The AmigaOS is an exception, having pre-emptive multitasking from its very first version. Windows NT was the first version of Microsoft Windows which enforced preemptive multitasking, but it didn't reach the home user market until Windows XP (since Windows NT was targeted at professionals).

**Disk access and file systems**

Filesystems allow users and programs to organize and sort files on a computer, often through the use of directories (or "folders")

Access to data stored on disks is a central feature of all operating systems. Computers store data on disks using files, which are structured in specific ways in order to allow for faster access, higher reliability, and to make better use out of the drive's available space. The specific way in which files are stored on a disk is called a file system, and enables files to have names and attributes. It also allows them to be stored in a hierarchy of directories or folders arranged in a directory tree.

Early operating systems generally supported a single type of disk drive and only one kind of file system. Early file systems were limited in their capacity, speed, and in the kinds of file names and directory structures they could use. These limitations often reflected limitations in the operating systems they were designed for, making it very difficult for an operating system to support more than one file system.

While many simpler operating systems support a limited range of options for accessing storage systems, operating systems like UNIX and Linux support a technology known as a virtual file system or VFS. An operating system such as UNIX supports a wide array of storage devices, regardless of their design or file systems, allowing them to be accessed through a common application programming interface (API). This makes it unnecessary for programs to have any knowledge about the device they are accessing. A VFS allows the operating system to provide programs with access to an unlimited number of devices with an infinite variety of file systems installed on them, through the use of specific device drivers and file system drivers.

A connected storage device, such as a hard drive, is accessed through a device driver. The device driver understands the specific language of the drive and is able to translate that language into a standard language used by the operating system to access all disk drives. On UNIX, this is the language of block devices.

When the kernel has an appropriate device driver in place, it can then access the contents of the disk drive in raw format, which may contain one or more file systems. A file system driver is used to translate the commands used to access each specific file system into a standard set of commands that the operating system can use to talk to all file systems. Programs can then deal with these file systems on the basis of filenames, and directories/folders, contained within a hierarchical structure. They can create, delete, open, and close files, as well as gather various information about them, including access permissions, size, free space, and creation and modification dates.

Various differences between file systems make supporting all file systems difficult. Allowed characters in file names, case sensitivity, and the presence of various kinds of file attributes makes the implementation of a single interface for every file system a daunting task. Operating systems tend to recommend using (and so support natively) file systems specifically designed for them; for example, NTFS in Windows and ext3 and ReiserFS in Linux. However, in practice, third party drives are usually available to give support for the most widely used file systems in most general-purpose operating systems (for example, NTFS is available in Linux through NTFS-3g, and ext2/3 and ReiserFS are available in Windows through third-party software).

Support for file systems is highly varied among modern operating systems, although there are several common file systems which almost all operating systems include support and drivers for. Operating systems vary on file system support and on the disk formats they may be installed on. Under Windows, each file system is usually limited in application to certain media; for example, CDs must use ISO 9660 or UDF, and as of Windows Vista, NTFS is the only file system which the operating system can be installed on. It is possible to install Linux onto many types of file systems. Unlike other operating systems, Linux and UNIX allow any file system to be used regardless of the media it is stored in, whether it is a hard drive, a disc (CD,DVD...), a USB flash drive, or even contained within a file located on another file system.

**Device drivers**

A device driver is a specific type of computer software developed to allow interaction with hardware devices. Typically this constitutes an interface for communicating with the device, through the specific computer bus or communications subsystem that the hardware is connected to, providing commands to and/or receiving data from the device, and on the other end, the requisite interfaces to the operating system and software applications. It is a specialized hardware-dependent computer program which is also operating system specific that enables another program, typically an operating system or applications software package or computer program running under the operating system kernel, to interact transparently with a hardware device, and usually provides the requisite interrupt handling necessary for any necessary asynchronous time-dependent hardware interfacing needs.

The key design goal of device drivers is abstraction. Every model of hardware (even within the same class of device) is different. Newer models also are released by manufacturers that provide more reliable or better performance and these newer models are often controlled differently. Computers and their operating systems cannot be expected to know how to control every device, both now and in the future. To solve this problem, operating systems essentially dictate how every type of device should be controlled. The function of the device driver is then to translate these operating system mandated function calls into device specific calls. In theory a new device, which is controlled in a new manner, should function correctly if a suitable driver is available. This new driver will ensure that the device appears to operate as usual from the operating system's point of view.

Under versions of Windows before Vista and versions of Linux before 2.6, all driver execution was co-operative, meaning that if a driver entered an infinite loop it would freeze the system. More recent revisions of these operating systems incorporate kernel preemption, where the kernel interrupts the driver to give it tasks, and then separates itself from the process until it receives a response from the device driver, or gives it more tasks to do.

**Networking**

Currently most operating systems support a variety of networking protocols, hardware, and applications for using them. This means that computers running dissimilar operating systems can participate in a common network for sharing resources such as computing, files, printers, and scanners using either wired or wireless connections. Networks can essentially allow a computer's operating system to access the resources of a remote computer to support the same functions as it could if those resources were connected directly to the local computer. This includes everything from simple communication, to using networked file systems or even sharing another computer's graphics or sound hardware. Some network services allow the resources of a computer to be accessed transparently, such as SSH which allows networked users direct access to a computer's command line interface.

Client/server networking allows a program on a computer, called a client, to connect via a network to another computer, called a server. Servers offer (or host) various services to other network computers and users. These services are usually provided through ports or numbered access points beyond the server's network address. Each port number is usually associated with a maximum of one running program, which is responsible for handling requests to that port. A daemon, being a user program, can in turn access the local hardware resources of that computer by passing requests to the operating system kernel.

Many operating systems support one or more vendor-specific or open networking protocols as well, for example, SNA on IBM systems, DECnet on systems from Digital Equipment Corporation, and Microsoft-specific protocols (SMB) on Windows. Specific protocols for specific tasks may also be supported such as NFS for file access. Protocols like ESound, or esd can be easily extended over the network to provide sound from local applications, on a remote system's sound hardware.

**Security**

A computer being secure depends on a number of technologies working properly. A modern operating system provides access to a number of resources, which are available to software running on the system, and to external devices like networks via the kernel.

The operating system must be capable of distinguishing between requests which should be allowed to be processed, and others which should not be processed. While some systems may simply distinguish between "privileged" and "non-privileged", systems commonly have a form of requester identity, such as a user name. To establish identity there may be a process of authentication. Often a username must be quoted, and each username may have a password. Other methods of authentication, such as magnetic cards or biometric data, might be used instead. In some cases, especially connections from the network, resources may be accessed with no authentication at all (such as reading files over a network share). Also covered by the concept of requester identity is authorization; the particular services and resources accessible by the requester once logged into a system are tied to either the requester's user account or to the variously configured groups of users to which the requester belongs.

In addition to the allow/disallow model of security, a system with a high level of security will also offer auditing options. These would allow tracking of requests for access to resources (such as, "who has been reading this file?"). Internal security, or security from an already running program is only possible if all possibly harmful requests must be carried out through interrupts to the operating system kernel. If programs can directly access hardware and resources, they cannot be secured.

External security involves a request from outside the computer, such as a login at a connected console or some kind of network connection. External requests are often passed through device drivers to the operating system's kernel, where they can be passed onto applications, or carried out directly. Security of operating systems has long been a concern because of highly sensitive data held on computers, both of a commercial and military nature. The United States Government Department of Defense (DoD) created the Trusted Computer System Evaluation Criteria (TCSEC) which is a standard that sets basic requirements for assessing the effectiveness of security. This became of vital importance to operating system makers, because the TCSEC was used to evaluate, classify and select trusted operating systems being considered for the processing, storage and retrieval of sensitive or classified information.

Network services include offerings such as file sharing, print services, email, web sites, and file transfer protocols (FTP), most of which can have compromised security. At the front line of security are hardware devices known as firewalls or intrusion detection/prevention systems. At the operating system level, there are a number of software firewalls available, as well as intrusion detection/prevention systems. Most modern operating systems include a software firewall, which is enabled by default. A software firewall can be configured to allow or deny network traffic to or from a service or application running on the operating system. Therefore, one can install and be running an insecure service, such as Telnet or FTP, and not have to be threatened by a security breach because the firewall would deny all traffic trying to connect to the service on that port.

An alternative strategy, and the only sandbox strategy available in systems that do not meet the Popek and Goldberg virtualization requirements, is the operating system not running user programs as native code, but instead either emulates a processor or provides a host for a p-code based system such as Java.

Internal security is especially relevant for multi-user systems; it allows each user of the system to have private files that the other users cannot tamper with or read. Internal security is also vital if auditing is to be of any use, since a program can potentially bypass the operating system, inclusive of bypassing auditing.

**User interface**

A screenshot of the Bourne Again Shell command line. Each command is typed out after the 'prompt', and then its output appears below, working its way down the screen. The current command prompt is at the bottom.

Every computer that is to be operated by an individual requires a user interface. The user interface is usually referred to as a shell and is essential if human interaction is to be supported. The user interface views the directory structure and requests services from the operating system that will acquire data from input hardware devices, such as a keyboard, mouse or credit card reader, and requests operating system services to display prompts, status messages and such on output hardware devices, such as a video monitor or printer. The two most common forms of a user interface have historically been the command-line interface, where computer commands are typed out line-by-line, and the graphical user interface, where a visual environment (most commonly a WIMP) is present.

**Graphical user interfaces**

A screenshot of the KDE Plasma Desktop graphical user interface. Programs take the form of images on the screen, and the files, folders (directories), and applications take the form of icons and symbols. A mouse is used to navigate the computer.

Most of the modern computer systems support graphical user interfaces (GUI), and often include them. In some computer systems, such as the original implementation of Mac OS, the GUI is integrated into the kernel.

While technically a graphical user interface is not an operating system service, incorporating support for one into the operating system kernel can allow the GUI to be more responsive by reducing the number of context switches required for the GUI to perform its output functions. Other operating systems are modular, separating the graphics subsystem from the kernel and the Operating System. In the 1980s UNIX, VMS and many others had operating systems that were built this way. Linux and Mac OS X are also built this way. Modern releases of Microsoft Windows such as Windows Vista implement a graphics subsystem that is mostly in user-space; however the graphics drawing routines of versions between Windows NT 4.0 and Windows Server 2003 exist mostly in kernel space. Windows 9x had very little distinction between the interface and the kernel.

Many computer operating systems allow the user to install or create any user interface they desire. The X Window System in conjunction with GNOME or KDE Plasma Desktop is a commonly found setup on most Unix and Unix-like (BSD, Linux, Solaris) systems. A number of Windows shell replacements have been released for Microsoft Windows, which offer alternatives to the included Windows shell, but the shell itself cannot be separated from Windows.

Numerous Unix-based GUIs have existed over time, most derived from X11. Competition among the various vendors of Unix (HP, IBM, Sun) led to much fragmentation, though an effort to standardize in the 1990s to COSE and CDE failed for various reasons, and were eventually eclipsed by the widespread adoption of GNOME and K Desktop Environment. Prior to free software-based toolkits and desktop environments, Motif was the prevalent toolkit/desktop combination (and was the basis upon which CDE was developed).

Graphical user interfaces evolve over time. For example, Windows has modified its user interface almost every time a new major version of Windows is released, and the Mac OS GUI changed dramatically with the introduction of Mac OS X in 1999.[27]

**Real-time operating systems**

A real-time operating system (RTOS) is an operating system intended for applications with fixed deadlines (real-time computing). Such applications include some small embedded systems, automobile engine controllers, industrial robots, spacecraft, industrial control, and some large-scale computing systems.

An early example of a large-scale real-time operating system was Transaction Processing Facility developed by American Airlines and IBM for the Sabre Airline Reservations System.

Embedded systems that have fixed deadlines use a real-time operating system such as VxWorks, PikeOS, eCos, QNX, MontaVista Linux and RTLinux. Windows CE is a real-time operating system that shares similar APIs to desktop Windows but shares none of desktop Windows' codebase.[citation needed] Symbian OS also has an RTOS kernel (EKA2) starting with version 8.0b.

Some embedded systems use operating systems such as Palm OS, BSD, and Linux, although such operating systems do not support real-time computing.

**Operating system development as a hobby**

Operating system development is one of the most complicated activities in which a computing hobbyist may engage. A hobby operating system may be classified as one whose code has not been directly derived from an existing operating system, and has few users and active developers. [28]

In some cases, hobby development is in support of a "homebrew" computing device, for example, a simple single-board computer powered by a 6502 microprocessor. Or, development may be for an architecture already in widespread use. Operating system development may come from entirely new concepts, or may commence by modeling an existing operating system. In either case, the hobbyist is his/her own developer, or may interact with a small and sometimes unstructured group of individuals who have like interests.

Examples of a hobby operating system include ReactOS and Syllable.

**Diversity of operating systems and portability**

Application software is generally written for use on a specific operating system, and sometimes even for specific hardware. When porting the application to run on another OS, the functionality required by that application may be implemented differently by that OS (the names of functions, meaning of arguments, etc.) requiring the application to be adapted, changed, or otherwise maintained.

This cost in supporting operating systems diversity can be avoided by instead writing applications against software platforms like Java or Qt. These abstractions have already borne the cost of adaptation to specific operating systems and their system libraries.

Another approach is for operating system vendors to adopt standards. For example, POSIX and OS abstraction layers provide commonalities that reduce porting costs.

**DOMAIN ADMINISTRATION**

**active directory Domain services** Active directory Domain Service (ADDS) is a distributed database that handles network administration and security. This Server Role provides the network administrator an interactive, hierarchical and secure infrastructure in which to administer resources. Once the Server Role ADDS and Dcpromo.exe (DC to a forest) is installed the computer is then promoted to Domain Controller.

The Domain Controller deals with security authentication for all users and devices. Active directory holds all information such as users, groups, computers, printers and files. This information is stored in Objects which are made available within the domain forest (logical security boundary). Objects within a domain are contained in Organisational Units (OUs), allowing network administrators to delegate the relevant security and share permissions for that particular OU or object. This is an efficient method in dealing with many objects in an instance. The advantages with Windows Server 2008r2 running Active Directory Domain Services are:

**Domain Naming System (DNS).** This maps names to IP addresses aiding human interaction User identity with password protected logon using AD and LDAP (Lightweight directory protocol) Advanced Encryption with AES 128 and 256 Kerberos V5 Authentication Protocol Backup services and servers minimizing data redundancy.

Integrating DNS zones with multi-master data replication reduces the use of network bandwidth and minimises data redundancy through secure dynamic updates

**Access Control Lists**

Trusts created to share and delegate information with other domains.

**Domain naming system (DNS)**

The Windows Domain Naming System is a naming system used to map names to IP (Internet Protocol) addresses. DNS is part of the protocol suite TCP/IP (Transmission Control Protocol) which transports data over LANs and WANs. This protocol suite formats, addresses, transmits and routes data to be sent to the receiver. Windows 2008r2 by default will install DNS when AD DS is installed.

**Multiple Domains**

Multiple Domains are configured when one domain needs to share resources with another domain. These domains work in a hierarchical structure and are organized in trees and forests. Forests are the logical security boundary of the network infrastructure with trees containing a hierarchical structure of the domains. The first domain on the network is the root domain which then becomes the parent domain for the next domain added. This sub domain of the parent is called the child domain. To optimize performance in a multi domain environment a network using a WAN connection breaks the network down into sites. Sites structure a network into LAN segments utilizing the high speed bandwidth. Sites help to reduce AD traffic in instances such as logon and replication. More domains can be added to the forest to replicate data over the network. To enable windows server 2008 to share data with other domains trust are created.

**Groups** Groups are used to contain objects which can then be configured in mass saving time with administrative tasks.

**Trusts**

Trusts are authenticated communication links between domains. Trusts allow users from one domain to access resources from another domain. Once two domains have been connected the default trust applied to this gateway is a transitive two-way trust. This default trust creates a child domain to its structure. Authentication from the child domains carries upwards to the trusted domain for any changes in the global catalogue. Types of trusts:

External non-transitive one and two way trusts. Used to access resources from Windows NT and 4.0 domains.

Realm transitive or non-transitive one and two way trusts. Used to share information between third party severs and window servers.

Forest transitive one and two way trusts. Used to communicate between two forests.

Shortcut transitive one and two way trusts. Used shortcut trust to improve logon times between domains.

**Global Catalogue Server**

2008r2 servers are set by default to become Global Catalogue Servers (GC) once they have been promoted to DC. The GC holds information relating to the AD database for that domain. This information is stored in the NTDS.dit file and provides a searchable index for objects within the domain using port 3268. The domain controller does not have information relating to resources outside that domain. DC uses a GC to contain all the information about objects in the forest. To manage this data the GC only holds enough information about an objects attributes to point to the object in the forest. This allows users from one domain to logon from another domain and access resources from within the forest. GC servers communicate with other GC servers to:

Locates User logon Information known as a UPN (unique principle name).

Locates directory Information in the forest.

Provides forest wide searches.

Provides forest wide services.

Directory database changes when made are updated to the GC servers in the forest. All Domain controllers with writeable attributes save data changes to their GC directory. Data is replicated with DFS (distributed file system).

**DFS(**distributed file system)

The DFS service (Dfssvc.exe) is the core component of the DFS physical and logical structure. DFS Namespace allows an administrator to group shared folders stored on many servers into a hierarchal structured namespace. This displays the shared root folder with subfolders that relate to that namespace. DFS namespace structure stores file shares from multiple servers and sites. By using DFS namespace you can expand the availability of resources over the network and connect users automatically to these resources within the AD DS sites.

DFS Replication is a multi-master replication tool which allows an administrator to efficiently replicate shared folders over the network and multiple servers. This procedure is an effective way in dealing with limited bandwidth. Remote Differential Compression (RDC) which is a compression algorithm which enables changes to be made to a file that has been edited. This is then replicated to all the GC servers on the network. For bandwidth and server efficiency changes only take place with the actual data that’s edited. DFS benefits the network with:

Fault tolerant replication of data to multiple locations on a network.

Easy access to shared resources through logical structure.

**Load balancing.**

DFS multi-master replication is implemented when a change has been made to the DFS folder. It is then replicated throughout the network to other DFS servers. Once the initial replication occurs between two servers the master copy is no longer a master copy it is then multi-master copy distributed to all DFS servers.

**DFS Connection Process**

Client connects to a domain/ member server storing DFS using UNC.

The Server then responds to the request and gives the location or of the resource to the user.

Client caches the location of the resource and can now access the resource directly without asking the DFS server.

A client will periodically ask the DFS server of any changes to the location. The time to live before a referral is requested is set by default to 300 seconds (5 minutes) with 1800 seconds (30 minutes).

**SYNCHRONISATION**

The Kerberos authentication protocol requires all computers in the domain to be time synced. Computers 5 minutes out of sync cannot join the network. Windows Server hosts PDC (primary domain controllers) to configure and synchronise with an external NTP (network time protocol) server. All DCs synchronise with the PDC time. Computers and members servers synchronise their time with the authenticating DC.

**BRANCH CACHE**

BranchCache requires 2008r2 and Windows 7 Enterprise / Ultimate. BranchCache is the method of sharing files by caching them to the local network over a wide area network (WAN). When a resource is requested it is held on the local network (high speed bandwidth) this improves performance with the lower bandwidth capabilities of a wide area network (WAN). BranchCache also supports end-to-end encryption between clients and servers.

BranchCache Modes

Requirements

Description

Hosted Mode

Server 2008r2 at each branch.

SSL certificate

Round trip calculated. If this is more than the default 80 milliseconds the file is cached to the local server.

**Distributed Cache Mode**

Windows 7 Enterprise / Ultimate

Computer stores the file in cache. If another computer requests the file it sends a broadcast to all computers and the file is then transferred between the computers.

BranchCache can be configured either through group Policy or netsh (command line scripting). Group policy has five settings:

Branch cache Modes

Settings

Turn On BranchCache

Enable / Disable

Set BranchCache Distributed Cache Mode

Enable / Disable

Set BranchCache Hosted Mode

Configure BranchCache Server

Configure BranchCache for network files

Set latency times (default 80milliseconds) for roundtrip.

Set Percentage of disc space use for client cache,Default set to 5%

**Group Policy (GPO)**

Group policies are held in active directory in SYSVOL. GPOs configure user and computer settings such as passwords, control panel options, firewalls. There are four types of GPOs, local, site, domain and OUs. To configure a domain based policy the group policy object editor is used. The group policy management console is used to link the policy to sites, domains and OUs. Group policies hierarchical order is with the lowest precedence being local, site, domain and finally OUs with the highest precedence (group policy setting to be applied in that instance).

**CONCLUSION**

Windows Server 2008 R2 manages a Multi-Domain environment securely and efficiently with AD DS and the other server roles available at its disposal. AD DS Hierarchical and logical structures simplify server data along with Kerberos, group policies and OUs setting the security and permissions for resources over a network.

BranchCache being one of the new editions to the server package furthers an organisations networking ability to provide necessary resources wherever and whenever requested. One disadvantage could be the added software cost (XP clients). BranchCache requires clients to be using the Windows 7 OS. Another possible disadvantage to take into consideration is the hardware aspect. If hardware specifications are not met what would be the upgrade costs for a company.

From the research carried out it is clear that the Windows server 2008 R2 edition provides a robust secure networking environment giving administrators more control with the server and network infrastructure. In today’s market place the Windows server 2008r R2 operating system enhances an organization of any size with IT productivity and performance and reliability.

================================================================= **SYSTEM HARDENING**

The idea of operating system hardening is aimed at minimizing a computer's exposure to current and even future threats by configuring the operating system and removing applications and devices that are unnecessary. In the modern world, security takes an increasingly predominantly important role. The Information technology industry is facing challenges in public confidence at the discovery of vulnerabilities, and clients are expecting security to be delivered out of the box, even on programs that were designed with no security issues in mind. Software maintainers have to face the challenge of improving the security of their programs. Unfortunately, they are often under-equipped to do so.

Operating system hardening basically means installation of a new server in a more secure fashion and maintenance of the security and integrity of the application software and server. Planning to acquire a virtual private server or a dedicated server calls for adequate preparation of a server-hardening checklist before launching the web application on the server.

The running UNIX which is used by almost 66% of Internet servers first and foremost must configure the host and system file for hardening (Pichnarczyk et al, 1994).

System administrators should make sure that the UNIX server is performing at its best by configuring the system control files. This is essential to optimization of operation of the server. Apart from this, secure patch updates are an essential factor to look into while hardening the server.

***Every Operating system comes with some security vulnerabilities. This calls for hardening of Windows, UNIX and other operating systems to reduce vulnerabilities.***

The Information Systems manager looking after corporate servers, databases and firewalls should have knowledge of the fundamentals of operating system hardening. Operating system hardening is the black art that ensures all known operating system vulnerabilities are plugged, and monitored.

Operating system hardening is an important feature in securing computing. One must ensure that network performance and security does prevent network problems, conduct effective troubleshooting and also quickly take necessary actions to solve possible problems.

We need to know how the network bandwidth is used to help us in accounting, auditing and network planning processes. Monitoring of network traffic and conducting of forensic analysis ensures adherence to company policies any violation is recorded and stopped timely.

The process of securing an operating system is referred to as “hardening” or at times as “lock down.” It is important to an organization’s security because it reduces the chances of attack. Issues come up because various operating systems require different hardening processes within a data center. For example, a system administrator may have deep expertise in one operating system but may possess limited or even no expertise with the other operating systems. System administrators also have a task of ensuring that each server remains locked down consistently to avoid vulnerabilities.

This requires maintenance of a predetermined configuration policy on every server. Implementing safety security measures ensures data is less susceptible to malicious attacks and able to rebound more quickly when attacks occur. One of the best practices for the security of an operating system includes a standardized hardening of all servers using appropriate standard methodologies.

Maintenance of security is a challenge. It depends on the size of the organization and number of servers. This in essence determines the amount of resources required. Use of automated tools can reduce the number of resources required and also helps with consistency. In addition, centralized view of all servers (regardless of the operating system) provides system administrators with the ability of assessing the security state of groups of servers and also lock them down automatically.

**Operating System Hardening**

Hardening of operating systems involves ensuring that the configuring of the system so as to limit the possibility of either internal or even external attack. While the hardening methods vary from one operating system to another, the baselining of Windows, UNIX, Linux, MacOS X or any other system has the same concepts. Some basic hardening techniques include the following:

Non-essential services - It is of importance that an operating system can only be configured to perform tasks for which it is assigned. For example, it is not necessary to have SMTP or HTTP services run on the system only when a host is functioning as a mail server or web.

Fixes and Patches –This is an ongoing endeavor. Operating systems should be updated with the most recent security updates.

Password Management - Many operating systems provide options for strong passwords enforcement. This ensures that users are prevented from configuration of weak passwords which can be easily guessed. Additionally, users are able to regularly change passwords. User accounts are disabled after a number of failed attempts to login.

Unnecessary accounts - Unused user accounts that are unnecessary must be disabled or even deleted from the operating systems. It is crucial to monitor employee turnover; when the employees leave an organization their accounts should be disabled or deleted.

Directory and File Protection – There should be strict controlled measures put in place for accessing directories and files through use of file permissions and access control lists.

File System Encryption - Some file systems support encryption of files and folders. For additional security, it is vital to ensure that partitions in a hard disk are formatted together using a file system that supports encryption features like NTFS for Windows and so on. This should be applied to sensitive data.

Enable Logging - The Operating system should be configured to ensure logging of all errors, activities and warnings.

File Sharing - Unnecessary file sharing should be disabled

Hardening the Network- Different techniques can be used to achieve this such as updating hardware and software (This is a vital part of network hardening; it involves ensuring that the network firmware in the routers are updated regularly with the latest fixes and patches).

Password Protection - Wireless access points and routers have provisions for interfaces that allow the network to be remotely management and accessed. Such devices should therefore be protected by strong passwords.

Unnecessary Services and Protocols - All services and protocols that are unnecessary should be disabled; ideally, the hosts in the network should have such services and protocols deleted. A case in point is that, it is unnecessary to have protocols for AppleTalk installed and configured on any systems in a pure TCP/IP network environment.

Ports - Ports that are not needed should be blocked by a firewall and any associated services in the hosts in the network should be disabled. For example, it is not logical to allow traffic for port 80 to pass through the firewall in a network that has a host acting as a web server.

Wireless Security - There configured of wireless networks up to highest available security level.

Restricted Network Access - A number of steps should be taken in order to prevent unauthorized access to internal networks. First and foremost, a firewall should be used between the network and the internet. Other available options may include the use of access control lists (ACLs) and Network Address Translation (NAT). Authorized remote access should only be enabled using secure tunnels and virtual private networks (Bagchi, 2003).

**Application Hardening**

This involves inclusion of all applications and services installed in host systems in a network in the hardening procedure; this ensures they do not pose as weak or vulnerable links for the security defenses. Services and applications that come installed with default settings with the operating system need to be reviewed.

Web Servers

These are responsible for availing web pages requested by web browsers; web hosting is among the most common web server services available.

Private sites require methods for authentication in place. Intranet (sites restricted for use to internal members only) approach can be used to block unauthorized access by external users; a firewall is used for this.

Web server logs need to be reviewed routinely for any suspicious activity.

As is the case with all software’s, necessary steps should be taken to ensure that web servers are updated with the latest vendor supplied patches.

Email Servers

To secure mail servers, configuration options that are not needed by the software running in the mail servers should be disabled and the most recent updates supplied by vendors utilized. Authentication should be used so as to ensure that only authorized users are able to receive and send email messages.

FTP Servers

File Transfer Protocol (FTP) allows files to be uploaded to and downloaded from remote servers. Servers can be accessed by either authenticated FTP or anonymous FTP. Caution should be taken when using anonymous FTP accounts; they require regular monitoring as they can very easily be exploited by hackers. In case FTP is authenticated, Secure FTP is used to encrypt login IDs and passwords, rather than allowing data to be transferred as plain text.

DNS Servers

DNS (Domain Name Servers) translate web site URL addresses to the equivalent IP addresses for routing across network devices. DNS software should be updated regularly and security measures like authentication should be employed to prevent ‘illegal’ transfers to unauthorized zones.

Unauthorized access or server intrusion can be mitigated or prevented by disabling or blocking port 53, or by using one or more pre-determined external systems to restrict access to the DNS server.

Service detail

Threats on operating system are real. Operating systems can be susceptible to a number of internal and external attacks due to un-patched vulnerabilities, disgruntled employees or in some case mis-configured server settings. Hardening of the operating system protects the organization against security breaches and malicious attacks. It also improves the overall efficiency of the operating system environment by verifying user permissions, patching vulnerabilities, installation of necessary software updates and deactivating unnecessary programs.

Services Identification

The purpose of each Unix Server should be identified.

How the computer will be used?

What categories of information will be stored in the computer?

What kind of information will be processed by the computer?

What are the security requirements and concerns for the information?

What network services will be provided by the computer?

What are the security requirements for these network services?

Identify the network services that will be provided on the server.

Servers should be dedicated to a single service. This simplifies the configuration and reduces the likelihood of configuration errors. In the case of the servers, the application server should always be limited to www or https services.

The db2 server should be ports 50000 and 50001. This can also eliminate unexpected and unsafe interactions among the services that present opportunities unauthorized intruders. In some cases, it may be required to offer more than one service on a single host computer. For example, the server software from vendors combines the hypertext transfer protocol (HTTP) and the file transfer protocol (FTP) services in a single package (Bookholdt, 1989).

It may therefore be appropriate to provide access to public information through both FTP and HTTP protocols from the same server host. I don’t recommend this because it is a less secure configuration.

Determine how the servers will be connected to your network. There are many concerns that relate to network connections that can affect the configuration and use of any one computer. Many organizations usually use a broadcast technology such as Ethernet for their local area networks. In such cases, information traversing a network segment can be seen by any computer on that segment. This requires that you should only place “trusted” computers on the same network segment, if not, encrypts information before transmitting it.

Environment Analysis

Considerations that must be taken into account before implementation can begin on a UNIX server include; what is the version of operating system that exists on the system? Where will the server be located? What kind of programs will be running on this server? These three questions need to be answered so as to direct the path of a security policy. Depending on the answer to these questions, the level of security can then be determined. For example, if a server simply runs audits, it is most likely then to have the same subnet as that of other servers (Sumitabh, 2006).

Planning

This is where you must define the overall security policies and goals. In many organizations, this step is normally performed at the corporate level, and is likely therefore to have already been completed.

How much security is needed (High, Medium &Low)?

How much security can your business afford (Cost & time)?

What devices and components should be protected?

Security Policy

The server in an open environment that is easily accessible by more than one system administrators is physically vulnerable to threats. There is need to secure data and encrypt it as well as server security protecting these files from There should be server security protection of these files from possible copying and removal. If System Administrator beliefs that the information on the server is sensitive to the organization, then he is at liberty to request higher security.

Password policy

The computers should not be trusted to remain secure and should therefore be physically disconnected if there are any signs of vulnerability.

When more than one administrator is permitted to log in, a server stands a chance of being vulnerable to employees with access.

ACL Policy

Untrained users who are not familiar with the operating system and have access to the server may accidentally damage a system without knowledge of how to recover .There is therefore need to maintain the integrity and privacy of the system and server hardening features to avoid unauthorized malicious penetrations and exploits.

Monitoring

As soon as the infrastructure is built, the systems administrator is required to continuously monitor it for potential attacks and threats.  A viable option is performing weekly audits; this prevents frustrating other network users by slowing down the network with unnecessary traffic.  Problems discovered should then be addressed through the previous phases in order to find the best possible solution.

Incident Response

It beats logic to respond to an attack when exposure has already taken place for a considerable period.  One should react immediately in case of an attack in the system (CERT/CC, 2002).

The Management establishes a security policy based on the risks it is willing to bear. The policy itself does not set goals, but serves as a bridge between management’s goals and the aspect of technical implementation. There is therefore need to develop a server deployment plan that includes security issues. Most deployment plans only address the cost of the computers, installation of applications software, schedules to minimize work disruption, and user training.

Administrators charged with the responsibility of maintaining and operating the servers are the only ones that should be permitted access to the servers. User privileges on the servers should also be determined.  Authentication protocols should also be made.

Authentication can be in two forms: (1) operating system authentication usually employed for authentication of administrators and, (2) network authentication used to authentication information systems users (McLean, 1996).

**Conclusion**

Risks can be mitigated upon by an on going process of hardening of Operating Systems, Network, Database, Application and devices as well as relevant resources of the IT infrastructure. To minimize risk, contingency plan remains the most effective and efficient plan to safe guard the Organization’s Information system from collapse. Operating system hardening, anti-virus solution, periodical security patches up offer prevention, detection and corrective action plan are of benefit to any organization that has an information system in place.

In summary, the risks assessment processes are about making decisions so as to minimize the risks. The impact of an attack and the level of acceptable risk for any given situation is a fundamental policy decision.

Likewise, vulnerabilities are basically design issues that must be addressed during the design, development & implementation of information systems. One of the fundamental problems of risk management is to achieve a cost effective balance between design characteristics and the related counter measures to threats and their impacts on the organization.

Modern computing environments are distributed infrastructures which require any organization to develop intrusion detection strategies for the servers. I strongly believe that there are no sensors on the internal network. Many of the common intrusion detection methods therefore solely depend on the existence of various logs that the operating system can produce and on the availability of auditing tools that analyze these logs. This will be of help when installation of appropriate software tools and configuration of these tools and the operating system so as to collect and manage the necessary information.

The Organization must likewise update its computer deployment plan when relevant changes take place. These changes may be through new technologies, updates to your network architecture, new security threats and the addition of new classes of users or new organizational units. The environment will only work effectively and efficiently if the process is centralized.

It is of importance for financial institutions to develop, implement and monitor appropriate information security programs.

Whether systems are maintained in-house or by a third-party vendor, appropriate security controls and risk management techniques should be employed. A security program is composed of system architecture and effective security policies that should be supported by the risk assessment practices and tools (Bagchi, 2003). IT threats and counter measures evolve continuously.

Consequently, proactive risk assessment processes for institutions should be in place to identify emerging vulnerabilities or threats to the information systems. A solid IT security policy should identify detection, response and prevention measures to be taken. Preventive measures consist of assessment tools for determining vulnerability regularly and periodic system analysis. Intrusion detection tools can also be effective in detecting potential intrusions and system misuse.

This paper will be of help to Security administrators because they will have a basic knowledge of security tools, operating system hardening, intrusion detection and auditing. This knowledge can be applied directly to their servers and many vulnerable loopholes will be filled.

Therefore, it is critical that every operating system security-minded administrator is able to maintain their knowledge of security by researching and referring to various Internet resources.

**Abstract**

Operating system hardening can be defined as the process that addresses security weaknesses in an operating system by implementing the latest operating systems patches, hot fixes and as well updates as follow up to the specific policies and procedures to reduce attacks and system down time.

Hardening is a not a onetime activity, it is an ongoing task that mitigates the risk to performing high quality of computing. We have to install a secure production server in such a way as to remove unwanted devices, not allowing the default setting, fix up the miss configurations, developing new system programming, enhancement of the current configuration and applying new security patches before going to the production environment.

Hardening of the operating system should be able to support high integrity, availability, reliability, scalability privacy and confidentiality at the reduced levels so as to achieve the

Highest level of objective - benefits from the critical Information Technology infrastructure of the organization.

Safeguarding information and protecting the integrity of the organization’s network systems are very vital. IT security personnel in many companies have established policies that are applicable to their entire organization, but it is up to individual departments that manage the systems to implement security policies (Schneier, 1996).

Security professionals also recognize the need for flexibility when it comes to the implementation stage. This is due to the unique requirements of each department. Hardening of an operating system involves the entire removal of all tools that are not essential, utilities and many other systems administration options, any of which could be used to ease the way of a hacker's path to your systems.

===================================================================**DATA STRUCTURES**

Data Structure is logical and mathematical model to store data. So there are basic benefits of data structures: The memory space is properly used. It helps in data protection and management. It is used to organize data in such a way that the insertion deletion, searhing i.e. manipulation of data is done with minimal complexity, and that gives an efficient performance of our computing. By using data structures data can be easily and efficiently exchanged; it allows portability, comprehensibility, and adaptability of information.

Data structures are used in most programming allowing efficient management of large amounts of data. Data structures are the organizing element in software design, for some programming languages, and design methods. Data structures are based on a computer's ability to store, and retrieve data from anywhere in memory; record, and array structures are based on using arithmetic operations to compute the address of the data. The storing of addresses within the structure is called linked data structures. Specific program languages offer built in support for specific data structures, (i.e. one dimensional array are used in C programming, and hash tables are used in Pearl). An array is a type of data structure.

An array is a data structure consisting of a number of variables, having the same data type. A single variable name is given to an array to associate with the variables. Arrays are used by programmers as a means of organizing many data items into a single data structure. Elements of the array are written, and recognized by using subscript, which is parenthesis after the array name. The use of arrays simplifies the writing of a program by allowing the grouping of similar data, rather than writing each item in the program code, saving time, and money.

An example of an array would be days of the week:

Initialize data table

day\_table(1) = 'Sunday'

day\_table(2) = 'Monday'

day\_table(3) = 'Tuesday'

day\_table(4) = 'Wednesday'

day\_table(5) = 'Thursday'

day\_table(6) = 'Friday'

day\_table(7) = 'Saturday'

End

All high level languages share a set of intercepted framework of data structure that composes the languages. These common data structures are strings, arrays, I/O, Stacks, Queues, Linked Lists, Trees, Graphs, Hash tables, and Vectors.

Most programming languages feature some sort of library mechanism that allows data structure implementations to be reused by different programs. Modern languages usually come with standard libraries that implement the most common data structures. Examples are the C++ Standard Template Library, the Java Collections Framework, and Microsoft's .NET Framework.

**Data Structures in C Language** :

A data item refers to a single unit of values. For example, a student’s information may be divided into four items/properties GRNO, name, class, and semester. But the GRNO would be treated as unique/ item. Data are also organized into more complex types of structures. There are two types of data structure are available: 1.Linear 2. Non-Linear.

Linear Structures: In this type of data structure we perform insert, delete, search, update operations sequentially or in an order (like Ascending/Descending). for example you have a list having 5 elements containing A,B,C,D,E,F values if u want to find that on which location E is store in this list, you must compare E with A,B,C,D and finally with E along this you must perform an increment to counter. After that you will find the actual location of your required/search item with the help of counter in this example the value of counter=4.

Examples of Linear Data Structures are as follows: \* Array \* Linked List \* Queue \* Stack 1. Non-Linear: In this type of data structure we perform Traversing, insert, delete, search, update operation randomly. # Examples of Non-Linear Data Structures are as follows: \* Tree \* Graphs.

**Data Structure operations**: The following four operations play a major role in this text.

1. **Traversing**: Accessing each record exactly once so that certain items in the record may be processed. (This accessing and processing is sometimes called "visiting" the record.)

2. **Searching**: Finding the location of the record with a given key value, or finding the locations of all records, which satisfy one or more conditions.

3. **Inserting:** adding a new record to the structure.

4. **Deleting:** Removing a record from the structure.

5. **Sorting**: Arranging the records in some logical order.

**Some Data Structures and their use in programming Languages:**

**STACK:**

A stack is a last in, first out (LIFO) abstract data type and data structure. A stack can have any abstract data type as an element, but is characterized by only two fundamental operations: push and pop. The push operation adds to the top of the list, hiding any items already on the stack, or initializing the stack if it is empty. The pop operation removes an item from the top of the list, and returns this value to the caller. A pop either reveals previously concealed items, or results in an empty list.

A stack-oriented programming language is one that relies on a stack machine model for passing parameters. Several programming languages fit this description, notably Forth, RPL, PostScript, and also many Assembly languages (but on a much lower level).

Some languages, like LISP and Python, do not call for stack implementations, since push and pop functions are available for any list. All Forth-like languages (such as Adobe PostScript) are also designed around language-defined stacks that are directly visible to and manipulated by the programmer.

C++'s Standard Template Library provides a "stack" templated class which is restricted to only push/pop operations. Java's library contains a Stack class that is a specialization of Vector---this could be considered a design flaw, since the inherited get() method from Vector ignores the LIFO constraint of the Stack.

**ARRAYS:**

An array can be defined as the finite ordered set of homogeneous elements. Finite means that there are specific number of elements in an array, ordered means that elements are arranged in a sequence so that the first,second,third…nth element.

In pure functional programs it is common to represent arrays by association lists. Association lists have the disadvantage that the access time varies linearly both with the size of the array (counted in number of entries) and with the size of the index (counted in cons nodes).

**QUEUE:**

A queue is a particular kind of collection in which the entities in the collection are kept in order.It is based on First-In-First-Out (FIFO)principle. In a FIFO data structure, the first element added to the queue will be the first one to be removed. A queue is an example of a linear data structure.

**LINKED LIST:**

It is a method of organizing stored data in a computer memory or on a storage medium based on the logical order of the data and not the physical order. All stored data records are assigned a physical address in memory that the computer uses to locate the information. A linked list arranges the data by logic rather than by physical address.

**Memory Management:**

One of the most important functions of a programming language is to provide facilities for managing memory and the objects that are stored in memory. C provides three distinct ways to allocate memory for objects:

Static memory allocation: space for the object is provided in the binary at compile-time; these objects have an extent (or lifetime) as long as the binary which contains them is loaded into memory

Automatic memory allocation: temporary objects can be stored on the stack, and this space is automatically freed and reusable after the block in which they are declared is exited

Dynamic memory allocation: blocks of memory of arbitrary size can be requested at run-time using library functions such as malloc from a region of memory called the heap; these blocks persist until subsequently freed for reuse by calling the library function free

These three approaches are appropriate in different situations and have various tradeoffs. For example, static memory allocation has no allocation overhead, automatic allocation may involve a small amount of overhead, and dynamic memory allocation can potentially have a great deal of overhead for both allocation and deal location. On the other hand, stack space is typically much more limited and transient than either static memory or heap space, and dynamic memory allocation allows allocation of objects whose size is known only at run-time. Most C programs make extensive use of all three.

Where possible, automatic or static allocation is usually preferred because the storage is managed by the compiler, freeing the programmer of the potentially error-prone chore of manually allocating and releasing storage. However, many data structures can grow in size at runtime, and since static allocations (and automatic allocations in C89 and C90) must have a fixed size at compile-time, there are many situations in which dynamic allocation must be used. Prior to the C99 standard, variable-sized arrays were a common example of this (see malloc for an example of dynamically allocated arrays).

Automatically and dynamically allocated objects are only initialized if an initial value is explicitly specified; otherwise they initially have indeterminate values (typically, whatever bit pattern happens to be present in the storage, which might not even represent a valid value for that type). If the program attempts to access an uninitialized value, the results are undefined. Many modern compilers try to detect and warn about this problem, but both false positives and false negatives occur.

Another issue is that heap memory allocation has to be manually synchronized with its actual usage in any program in order for it to be reused as much as possible. For example, if the only pointer to a heap memory allocation goes out of scope or has its value overwritten before free() has been called, then that memory cannot be recovered for later reuse and is essentially lost to the program, a phenomenon known as a memory leak. Conversely, it is possible to release memory too soon and continue to access it; however, since the allocation system can re-allocate or itself use the freed memory, unpredictable behavior is likely to occur. Typically, the symptoms will appear in a portion of the program far removed from the actual error, making it difficult to track down the problem. Such issues are ameliorated in languages with automatic garbage collection.

**Object-oriented programming**(OOP) is a programming paradigm that represents concepts as "objects" that have data fields (attributes that describe the object) and associated procedures known as methods. Objects, which are usually instances of classes, are used to interact with one another to design applications and computer programs.[1][2] C++, Objective-C, Smalltalk, Java and C# are examples of object-oriented programming languages.

**Overview.**

In programming languages an object is the composition of nouns (like data such as numbers, strings, or variables) and verbs (like actions, such as functions).

An object oriented program may be viewed as a collection of interacting objects, as opposed to the conventional model, in which a program is seen as a list of tasks (subroutines) to perform. In OOP, each object is capable of receiving messages, processing data, and sending messages to other objects. Each object can be viewed as an independent "machine" with a distinct role or responsibility. Actions (or "methods") on these objects are closely associated with the object. For example, OOP data structures tend to "carry their own operators around with them" (or at least "inherit" them from a similar object or class)—except when they must be serialized.

Simple, non-OOP programs may be one "long" list of commands. More complex programs often group smaller sections of these statements into functions or subroutines—each of which might perform a particular task. With designs of this sort, it is common for some of the program's data to be 'global', i.e., accessible from any part of the program. As programs grow in size, allowing any function to modify any piece of data means that bugs can have wide-reaching effects.

In contrast, the object-oriented approach encourages the programmer to place data where it is not directly accessible by the rest of the program. Instead, the data is accessed by calling specially written functions, commonly called methods, which are bundled in with the data. These act as the intermediaries for retrieving or modifying the data they control. The programming construct that combines data with a set of methods for accessing and managing those data is called an object. The practice of using subroutines to examine or modify certain kinds of data was also used in non-OOP modular programming, well before the widespread use of object-oriented programming.

An object-oriented program usually contains different types of objects, each corresponding to a particular kind of complex data to manage, or perhaps to a real-world object or concept such as a bank account, a hockey player, or a bulldozer. A program might contain multiple copies of each type of object, one for each of the real-world objects the program deals with. For instance, there could be one bank account object for each real-world account at a particular bank. Each copy of the bank account object would be alike in the methods it offers for manipulating or reading its data, but the data inside each object would differ reflecting the different history of each account.

Objects can be thought of as encapsulating their data within a set of functions designed to ensure that the data are used appropriately, and to assist in that use. The object's methods typically include checks and safeguards specific to the data types the object contains. An object can also offer simple-to-use, standardized methods for performing particular operations on its data, while concealing the specifics of how those tasks are accomplished. In this way alterations can be made to the internal structure or methods of an object without requiring that the rest of the program be modified. This approach can also be used to offer standardized methods across different types of objects. As an example, several different types of objects might offer print methods. Each type of object might implement that print method in a different way, reflecting the different kinds of data each contains, but all the different print methods might be called in the same standardized manner from elsewhere in the program. These features become especially useful when more than one programmer is contributing code to a project or when the goal is to reuse code between projects.

Object-oriented programming has roots that can be traced to the 1960s. As hardware and software became increasingly complex, manageability often became a concern. Researchers studied ways to maintain software quality and developed object-oriented programming in part to address common problems by strongly emphasizing discrete, reusable units of programming logic[citation needed]. Object-oriented programming focuses on data rather than processes, with programs composed of self-sufficient modules ("classes"), each instance of which ("object") contains all the information needed to manipulate its own data structure ("members"). This was in contrast to the modular programming that had been dominant for many years and that focused on the function of a module, rather than specifically the data, but equally provided for code reuse and self-sufficient reusable units of programming logic, enabling collaboration through the use of linked modules (subroutines).

**History.**

Terminology invoking "objects" and "oriented" in the modern sense of object-oriented programming made its first appearance at MIT in the late 1950s and early 1960s. In the environment of the artificial intelligence group, as early as 1960, "object" could refer to identified items (LISP atoms) with properties (attributes);[3][4] Alan Kay was later to cite a detailed understanding of LISP internals as a strong influence on his thinking in 1966.[5] Another early MIT example was Sketchpad created by Ivan Sutherland in 1960–61; in the glossary of the 1963 technical report based on his dissertation about Sketchpad, Sutherland defined notions of "object" and "instance" (with the class concept covered by "master" or "definition"), albeit specialized to graphical interaction.[6] Also, an MIT ALGOL version, AED-0, linked data structures ("plexes", in that dialect) directly with procedures, prefiguring what were later termed "messages", "methods" and "member functions".[7][8]

The formal programming concept of objects was introduced in the 1960s in Simula 67, a major revision of Simula I, a programming language designed for discrete event simulation, created by Ole-Johan Dahl and Kristen Nygaard of the Norwegian Computing Center in Oslo.[9] Simula 67 was influenced by SIMSCRIPT and C.A.R. "Tony" Hoare's proposed "record classes".[7][10] Simula introduced the notion of classes and instances or objects (as well as subclasses, virtual methods, coroutines, and discrete event simulation) as part of an explicit programming paradigm. The language also used automatic garbage collection that had been invented earlier for the functional programming language Lisp. Simula was used for physical modeling, such as models to study and improve the movement of ships and their content through cargo ports. The ideas of Simula 67 influenced many later languages, including Smalltalk, derivatives of LISP (CLOS), Object Pascal, and C++.

The Smalltalk language, which was developed at Xerox PARC (by Alan Kay and others) in the 1970s, introduced the term object-oriented programming to represent the pervasive use of objects and messages as the basis for computation. Smalltalk creators were influenced by the ideas introduced in Simula 67, but Smalltalk was designed to be a fully dynamic system in which classes could be created and modified dynamically rather than statically as in Simula 67.[11] Smalltalk and with it OOP were introduced to a wider audience by the August 1981 issue of Byte Magazine.

In the 1970s, Kay's Smalltalk work had influenced the Lisp community to incorporate object-based techniques that were introduced to developers via the Lisp machine. Experimentation with various extensions to Lisp (such as LOOP and Flavors introducing multiple inheritance and mixins) eventually led to the Common Lisp Object System (CLOS, a part of the first standardized object-oriented programming language, ANSI Common Lisp), which integrates functional programming and object-oriented programming and allows extension via a Meta-object protocol. In the 1980s, there were a few attempts to design processor architectures that included hardware support for objects in memory but these were not successful. Examples include the Intel iAPX 432 and the Linn Smart Rekursiv.

In 1985, Bertrand Meyer produced the first design of the Eiffel language. Focused on software quality, Eiffel is among the purely object-oriented languages, but differs in the sense that the language itself is not only a programming language, but a notation supporting the entire software lifecycle. Meyer described the Eiffel software development method, based on a small number of key ideas from software engineering and computer science, in Object-Oriented Software Construction. Essential to the quality focus of Eiffel is Meyer's reliability mechanism, Design by Contract, which is an integral part of both the method and language.

Object-oriented programming developed as the dominant programming methodology in the early and mid 1990s when programming languages supporting the techniques became widely available. These included Visual FoxPro 3.0,[12][13][14] C++[citation needed], and Delphi[citation needed]. Its dominance was further enhanced by the rising popularity of graphical user interfaces, which rely heavily upon object-oriented programming techniques. An example of a closely related dynamic GUI library and OOP language can be found in the Cocoa frameworks on Mac OS X, written in Objective-C, an object-oriented, dynamic messaging extension to C based on Smalltalk. OOP toolkits also enhanced the popularity of event-driven programming (although this concept is not limited to OOP). Some[who?] feel that association with GUIs (real or perceived) was what propelled OOP into the programming mainstream.

At ETH Zürich, Niklaus Wirth and his colleagues had also been investigating such topics as data abstraction and modular programming (although this had been in common use in the 1960s or earlier). Modula-2 (1978) included both, and their succeeding design, Oberon, included a distinctive approach to object orientation, classes, and such. The approach is unlike[how?] Smalltalk, and very unlike[how?] C++.

Object-oriented features have been added to many previously existing languages, including Ada, BASIC, Fortran, Pascal, and COBOL. Adding these features to languages that were not initially designed for them often led to problems with compatibility and maintainability of code.

More recently, a number of languages have emerged that are primarily object-oriented, but that are also compatible with procedural methodology. Two such languages are Python and Ruby. Probably the most commercially-important recent object-oriented languages are Visual Basic.NET (VB.NET) and C#, both designed for Microsoft's .NET platform, and Java, developed by Sun Microsystems. Each of these two frameworks shows, in its own way, the benefit of using OOP by creating an abstraction from implementation. VB.NET and C# support cross-language inheritance, allowing classes defined in one language to subclass classes defined in the other language. Developers usually compile Java to bytecode, allowing Java to run on any operating system for which a Java virtual machine is available. VB.NET and C# make use of the Strategy pattern to accomplish cross-language inheritance, whereas Java makes use of the Adapter pattern[citation needed].

Just as procedural programming led to refinements of techniques such as structured programming, modern object-oriented software design methods include refinements[citation needed] such as the use of design patterns, design by contract, and modeling languages (such as UML).

**Fundamental features and concepts .**

A survey by Deborah J. Armstrong of nearly 40 years of computing literature identified a number of fundamental concepts, found in the large majority of definitions of OOP.[15]

Not all of these concepts appear in all object-oriented programming languages. For example, object-oriented programming that uses classes is sometimes called class-based programming, while prototype-based programming does not typically use classes. As a result, a significantly different yet analogous terminology is used to define the concepts of object and instance.

Benjamin C. Pierce and some other researchers view any attempt to distill OOP to a minimal set of features as futile. He nonetheless identifies fundamental features that support the OOP programming style in most object-oriented languages:[16]

**Dynamic dispatch** – when a method is invoked on an object, the object itself determines what code gets executed by looking up the method at run time in a table associated with the object. This feature distinguishes an object from an abstract data type (or module), which has a fixed (static) implementation of the operations for all instances. It is a programming methodology that gives modular component development while at the same time being very efficient.

**Encapsulation** (or multi-methods, in which case the state is kept separate)

**Subtype polymorphism**

**Object inheritance** (or delegation)

**Open recursion** – a special variable (syntactically it may be a keyword), usually called this or self, that allows a method body to invoke another method body of the same object. This variable is late-bound; it allows a method defined in one class to invoke another method that is defined later, in some subclass thereof.

Similarly, in his 2003 book, Concepts in programming languages, John C. Mitchell identifies four main features: dynamic dispatch, abstraction, subtype polymorphism, and inheritance.[17] Michael Lee Scott in Programming Language Pragmatics considers only encapsulation, inheritance and dynamic dispatch.[18]

Additional concepts used in object-oriented programming include:

Classes of objects

Instances of classes

Methods which act on the attached objects.

Message passing

Abstraction

Decoupling.

Decoupling refers to careful controls that separate code modules from particular use cases, which increases code re-usability. A common use of decoupling is to polymorphically decouple the encapsulation (see Bridge pattern and Adapter pattern) - for example, using a method interface that an encapsulated object must satisfy, as opposed to using the object's class.

**Additional features.**

Encapsulation Enforces Modularity

Encapsulation refers to the creation of self-contained modules that bind processing functions to the data. These user-defined data types are called "classes," and one instance of a class is an "object." For example, in a payroll system, a class could be Manager, and Pat and Jan could be two instances (two objects) of the Manager class. Encapsulation ensures good code modularity, which keeps routines separate and less prone to conflict with each other.

Inheritance Passes "Knowledge" Down

Classes are created in hierarchies, and inheritance lets the structure and methods in one class pass down the hierarchy. That means less programming is required when adding functions to complex systems. If a step is added at the bottom of a hierarchy, only the processing and data associated with that unique step must be added. Everything else above that step is inherited. The ability to reuse existing objects is considered a major advantage of object technology.

Polymorphism Takes any Shape

Object-oriented programming lets programmers create procedures for objects whose exact type is not known until runtime. For example, a screen cursor may change its shape from an arrow to a line depending on the program mode. The routine to move the cursor on screen in response to mouse movement can be written for "cursor," and polymorphism lets that cursor take simulating system behaviour. In the late 1960s, SIMULA was the first object-oriented language. In the 1970s, Xerox's Smalltalk was the first object-oriented programming language used to create the graphical user interface (GUI). Today, C++, C#, and Java are the major OOP languages, while Visual Basic.NET, Python and JavaScript are also popular. ACTOR and Eiffel were earlier OOP languages.

**Formal semantics.**

Objects are the run time entities in an object-oriented system. They may represent a person, a place, a bank account, a table of data, or any item that the program has to handle.

There have been several attempts at formalizing the concepts used in object-oriented programming. The following concepts and constructs have been used as interpretations of OOP concepts:

coalgebraic data types[19]

abstract data types (which have existential types) allow the definition of modules but these do not support dynamic dispatch

recursive types

encapsulated state

inheritance

records are basis for understanding objects if function literals can be stored in fields (like in functional programming languages), but the actual calculi need be considerably more complex to incorporate essential features of OOP. Several extensions of System F<: that deal with mutable objects have been studied;[20] these allow both subtype polymorphism and parametric polymorphism (generics)

Attempts to find a consensus definition or theory behind objects have not proven very successful (however, see Abadi & Cardelli, A Theory of Objects[20] for formal definitions of many OOP concepts and constructs), and often diverge widely. For example, some definitions focus on mental activities, and some on program structuring. One of the simpler definitions is that OOP is the act of using "map" data structures or arrays that can contain functions and pointers to other maps, all with some syntactic and scoping sugar on top. Inheritance can be performed by cloning the maps (sometimes called "prototyping").

**OOP languages.**

This section does not cite any references or sources. Please help improve this section by adding citations to reliable sources. Unsourced material may be challenged and removed. (August 2009)

Simula (1967) is generally accepted as the first language with the primary features of an object-oriented language. It was created for making simulation programs, in which what came to be called objects were the most important information representation. Smalltalk (1972 to 1980) is arguably the canonical example, and the one with which much of the theory of object-oriented programming was developed. Concerning the degree of object orientation, the following distinctions can be made:

Languages called "pure" OO languages, because everything in them is treated consistently as an object, from primitives such as characters and punctuation, all the way up to whole classes, prototypes, blocks, modules, etc. They were designed specifically to facilitate, even enforce, OO methods. Examples: Eiffel, Emerald,[21] JADE, Obix, Ruby, Scala, Smalltalk, Self

Languages designed mainly for OO programming, but with some procedural elements. Examples: Delphi/Object Pascal, C++, Java, C#, VB.NET, Python.

Languages that are historically procedural languages, but have been extended with some OO features. Examples: Pascal, Visual Basic (derived from BASIC), Fortran, Perl, COBOL 2002, PHP, ABAP.

Languages with most of the features of objects (classes, methods, inheritance, reusability), but in a distinctly original form. Examples: Oberon (Oberon-1 or Oberon-2) and Common Lisp.

Languages with abstract data type support, but not all features of object-orientation, sometimes called object-based languages. Examples: Modula-2, Pliant, CLU.

**OOP in dynamic languages.**

In recent years, object-oriented programming has become especially popular in dynamic programming languages. Python, Ruby and Groovy are dynamic languages built on OOP principles, while Perl and PHP have been adding object-oriented features since Perl 5 and PHP 4, and ColdFusion since version 5.

The Document Object Model of HTML, XHTML, and XML documents on the Internet has bindings to the popular JavaScript/ECMAScript language. JavaScript is perhaps the best known prototype-based programming language, which employs cloning from prototypes rather than inheriting from a class. Another scripting language that takes this approach is Lua. Earlier versions of ActionScript (a partial superset of the ECMA-262 R3, otherwise known as ECMAScript) also used a prototype-based object model.

**Design patterns.**

Challenges of object-oriented design are addressed by several methodologies. Most common is known as the design patterns codified by Gamma et al.. More broadly, the term "design patterns" can be used to refer to any general, repeatable solution to a commonly occurring problem in software design. Some of these commonly occurring problems have implications and solutions particular to object-oriented development.

**Inheritance and behavioral subtyping.**

It is intuitive to assume that inheritance creates a semantic "is a" relationship, and thus to infer that objects instantiated from subclasses can always be safely used instead of those instantiated from the superclass. This intuition is unfortunately false in most OOP languages, in particular in all those that allow mutable objects. Subtype polymorphism as enforced by the type checker in OOP languages (with mutable objects) cannot guarantee behavioral subtyping in any context. Behavioral subtyping is undecidable in general, so it cannot be implemented by a program (compiler). Class or object hierarchies must be carefully designed, considering possible incorrect uses that cannot be detected syntactically. This issue is known as the Liskov substitution principle.

**Gang of Four design patterns.**

Design Patterns: Elements of Reusable Object-Oriented Software is an influential book published in 1995 by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, often referred to humorously as the "Gang of Four". Along with exploring the capabilities and pitfalls of object-oriented programming, it describes 23 common programming problems and patterns for solving them. As of April 2007, the book was in its 36th printing.

The book describes the following patterns:

Creational patterns (5): Factory method pattern, Abstract factory pattern, Singleton pattern, Builder pattern, Prototype pattern

Structural patterns (7): Adapter pattern, Bridge pattern, Composite pattern, Decorator pattern, Facade pattern, Flyweight pattern, Proxy pattern

Behavioral patterns (11): Chain-of-responsibility pattern, Command pattern, Interpreter pattern, Iterator pattern, Mediator pattern, Memento pattern, Observer pattern, State pattern, Strategy pattern, Template method pattern, Visitor pattern

**Object-orientation and databases.**

Both object-oriented programming and relational database management systems (RDBMSs) are extremely common in software today. Since relational databases don't store objects directly (though some RDBMSs have object-oriented features to approximate this), there is a general need to bridge the two worlds. The problem of bridging object-oriented programming accesses and data patterns with relational databases is known as object-relational impedance mismatch. There are a number of approaches to cope with this problem, but no general solution without downsides.[22] One of the most common approaches is object-relational mapping, as found in libraries like Java Data Objects and Ruby on Rails' ActiveRecord.

There are also object databases that can be used to replace RDBMSs, but these have not been as technically and commercially successful as RDBMSs.

**Real-world modeling and relationships.**

OOP can be used to associate real-world objects and processes with digital counterparts. However, not everyone agrees that OOP facilitates direct real-world mapping (see Negative Criticism section) or that real-world mapping is even a worthy goal; Bertrand Meyer argues in Object-Oriented Software Construction[23] that a program is not a model of the world but a model of some part of the world; "Reality is a cousin twice removed". At the same time, some principal limitations of OOP had been noted.[24] For example, the Circle-ellipse problem is difficult to handle using OOP's concept of inheritance.

However, Niklaus Wirth (who popularized the adage now known as Wirth's law: "Software is getting slower more rapidly than hardware becomes faster") said of OOP in his paper, "Good Ideas through the Looking Glass", "This paradigm closely reflects the structure of systems 'in the real world', and it is therefore well suited to model complex systems with complex behaviours" (contrast KISS principle).

Steve Yegge and others noted that natural languages lack the OOP approach of strictly prioritizing things (objects/nouns) before actions (methods/verbs).[25] This problem may cause OOP to suffer more convoluted solutions than procedural programming.[26]

**OOP and control flow.**

OOP was developed to increase the reusability and maintainability of source code.[27] Transparent representation of the control flow had no priority and was meant to be handled by a compiler. With the increasing relevance of parallel hardware and multithreaded coding, developing transparent control flow becomes more important, something hard to achieve with OOP.[28][29][30][31]

**Responsibility- vs. data-driven design.**

Responsibility-driven design defines classes in terms of a contract, that is, a class should be defined around a responsibility and the information that it shares. This is contrasted by Wirfs-Brock and Wilkerson with data-driven design, where classes are defined around the data-structures that must be held. The authors hold that responsibility-driven design is preferable.

**Criticism.**

A number of well-known researchers and programmers have analysed the utility of OOP. Here is an incomplete list:

Luca Cardelli wrote a paper titled "Bad Engineering Properties of Object-Oriented Languages".[32]

Richard Stallman wrote in 1995, "Adding OOP to Emacs is not clearly an improvement; I used OOP when working on the Lisp Machine window systems, and I disagree with the usual view that it is a superior way to program."[33]

A study by Potok et al.[34] has shown no significant difference in productivity between OOP and procedural approaches.

Christopher J. Date stated that critical comparison of OOP to other technologies, relational in particular, is difficult because of lack of an agreed-upon and rigorous definition of OOP.[35] Date and Darwen[36] propose a theoretical foundation on OOP that uses OOP as a kind of customizable type system to support RDBMS.

Alexander Stepanov compares object orientation unfavourably to multimethods: "I find OOP technically unsound. It attempts to decompose the world in terms of interfaces that vary on a single type. To deal with the real problems you need multisorted algebras - families of interfaces that span multiple types. I find OOP philosophically unsound. It claims that everything is an object. Even if it is true it is not very interesting - saying that everything is an object is saying nothing at all. ...".[37]

Paul Graham has suggested that OOP's popularity within large companies is due to "large (and frequently changing) groups of mediocre programmers." According to Graham, the discipline imposed by OOP prevents any one programmer from "doing too much damage."[38]

Joe Armstrong, the principal inventor of Erlang, is quoted as saying "The problem with object-oriented languages is they've got all this implicit environment that they carry around with them. You wanted a banana but what you got was a gorilla holding the banana and the entire jungle."[39]

Richard Mansfield, author and former editor of COMPUTE! magazine, states that, "Like countless other intellectual fads over the years ("relevance", "communism", "modernism", and so on—history is littered with them), OOP will be with us until eventually reality asserts itself. But considering how OOP currently pervades both universities and workplaces, OOP may well prove to be a durable delusion. Entire generations of indoctrinated programmers continue to march out of the academy, committed to OOP and nothing but OOP for the rest of their lives."[40] He also is quoted as saying "OOP is to writing a program, what going through airport security is to flying".[41]

Steve Yegge, making a roundabout comparison with Functional programming, writes, "Object Oriented Programming puts the Nouns first and foremost. Why would you go to such lengths to put one part of speech on a pedestal? Why should one kind of concept take precedence over another? It's not as if OOP has suddenly made verbs less important in the way we actually think. It's a strangely skewed perspective."[42]

Rich Hickey, creator of Clojure, described object systems as overly simplistic models of the real world. He emphasized the inability of OOP to model time properly, which is getting increasingly problematic as software systems become more concurrent.[43]

Carnegie-Mellon University Professor Robert Harper in March 2011 wrote: "This semester Dan Licata and I are co-teaching a new course on functional programming for first-year prospective CS majors... Object-oriented programming is eliminated entirely from the introductory curriculum, because it is both anti-modular and anti-parallel by its very nature, and hence unsuitable for a modern CS curriculum. A proposed new course on object-oriented design methodology will be offered at the sophomore level for those students who wish to study this topic."

=========================================================================

**Relational database** is a database that has a collection of tables of data items, all of which is formally described and organized according to the relational model. Data in a single table represents a relation, from which the name of the database type comes. In typical solutions, tables may have additionally defined relationships with each other.

In the relational model, each table schema must identify a column or group of columns, called the primary key, to uniquely identify each row. A relationship can then be established between each row in the table and a row in another table by creating a foreign key, a column or group of columns in one table that points to the primary key of another table. The relational model offers various levels of refinement of table organization and reorganization called database normalization. (See Normalization below.) The database management system (DBMS) of a relational database is called an RDBMS, and is the software of a relational database.

The relational database was first defined in June 1970 by Edgar Codd, of IBM's San Jose Research Laboratory.[1] Codd's view of what qualifies as an RDBMS is summarized in Codd's 12 rules. A relational database has become the predominant choice in storing data. Other models besides the relational model include the hierarchical database model and the network model.

**Terminology**

Relational database theory uses mathematical terminology, which are roughly equivalent to the SQL database terminology concerning normalization. The table below summarizes some of the most important relational database terms and their SQL database equivalents. It was first introduced in 1970 following the work of E.F.Codd.

A row or tuple has a relation schema, but an entire database has a relational schema.

A relation is defined as a set of tuples that have the same attributes. A tuple usually represents an object and information about that object. Objects are typically physical objects or concepts. A relation is usually described as a table, which is organized into rows and columns. All the data referenced by an attribute are in the same domain and conform to the same constraints.

The relational model specifies that the tuples of a relation have no specific order and that the tuples, in turn, impose no order on the attributes. Applications access data by specifying queries, which use operations such as select to identify tuples, project to identify attributes, and join to combine relations. Relations can be modified using the insert, delete, and update operators. New tuples can supply explicit values or be derived from a query. Similarly, queries identify tuples for updating or deleting.

Tuples by definition are unique. If the tuple contains a candidate or primary key then obviously it is unique; however, a primary key need not be defined for a row or record to be a tuple. The definition of a tuple requires that it be unique, but does not require a primary key to be defined. Because a tuple is unique, its attributes by definition constitute a superkey.

**Base and derived relations**

In a relational database, all data are stored and accessed via relations. Relations that store data are called "base relations", and in implementations are called "tables". Other relations do not store data, but are computed by applying relational operations to other relations. These relations are sometimes called "derived relations". In implementations these are called "views" or "queries". Derived relations are convenient in that they act as a single relation, even though they may grab information from several relations. Also, derived relations can be used as an abstraction layer.

**Domain**

A domain describes the set of possible values for a given attribute, and can be considered a constraint on the value of the attribute. Mathematically, attaching a domain to an attribute means that any value for the attribute must be an element of the specified set. The character data value 'ABC', for instance, is not in the integer domain, but the integer value 123 is in the integer domain.

**Constraints**

Constraints make it possible to further restrict the domain of an attribute. For instance, a constraint can restrict a given integer attribute to values between 1 and 10. Constraints provide one method of implementing business rules in the database. SQL implements constraint functionality in the form of check constraints. Constraints restrict the data that can be stored in relations. These are usually defined using expressions that result in a boolean value, indicating whether or not the data satisfies the constraint. Constraints can apply to single attributes, to a tuple (restricting combinations of attributes) or to an entire relation. Since every attribute has an associated domain, there are constraints (domain constraints). The two principal rules for the relational model are known as entity integrity and referential integrity.

**Primary keys**

A primary key uniquely specifies a tuple within a table. In order for an attribute to be a good primary key it must not repeat. While natural attributes (attributes used to describe the data being entered) are sometimes good primary keys, surrogate keys are often used instead. A surrogate key is an artificial attribute assigned to an object which uniquely identifies it (for instance, in a table of information about students at a school they might all be assigned a student ID in order to differentiate them). The surrogate key has no intrinsic (inherent) meaning, but rather is useful through its ability to uniquely identify a tuple. Another common occurrence, especially in regards to N:M cardinality is the composite key. A composite key is a key made up of two or more attributes within a table that (together) uniquely identify a record. (For example, in a database relating students, teachers, and classes. Classes could be uniquely identified by a composite key of their room number and time slot, since no other class could have exactly the same combination of attributes. In fact, use of a composite key such as this can be a form of data verification, albeit a weak one.)

**Foreign key**

A foreign key is a field in a relational table that matches the primary key column of another table. The foreign key can be used to cross-reference tables. Foreign keys need not have unique values in the referencing relation. Foreign keys effectively use the values of attributes in the referenced relation to restrict the domain of one or more attributes in the referencing relation. A foreign key could be described formally as: "For all tuples in the referencing relation projected over the referencing attributes, there must exist a tuple in the referenced relation projected over those same attributes such that the values in each of the referencing attributes match the corresponding values in the referenced attributes."

**Stored procedures**

A stored procedure is executable code that is associated with, and generally stored in, the database. Stored procedures usually collect and customize common operations, like inserting a tuple into a relation, gathering statistical information about usage patterns, or encapsulating complex business logic and calculations. Frequently they are used as an application programming interface (API) for security or simplicity. Implementations of stored procedures on SQL RDBMSs often allow developers to take advantage of procedural extensions (often vendor-specific) to the standard declarative SQL syntax. Stored procedures are not part of the relational database model, but all commercial implementations include them.

**Index**

An index is one way of providing quicker access to data. Indices can be created on any combination of attributes on a relation. Queries that filter using those attributes can find matching tuples randomly using the index, without having to check each tuple in turn. This is analogous to using the index of a book to go directly to the page on which the information you are looking for is found, so that you do not have to read the entire book to find what you are looking for. Relational databases typically supply multiple indexing techniques, each of which is optimal for some combination of data distribution, relation size, and typical access pattern. Indices are usually implemented via B+ trees, R-trees, and bitmaps. Indices are usually not considered part of the database, as they are considered an implementation detail, though indices are usually maintained by the same group that maintains the other parts of the database. It should be noted that use of efficient indexes on both primary and foreign keys can dramatically improve query performance. This is because B-tree indexes result in query times proportional to log(n) where n is the number of rows in a table and hash indexes result in constant time queries (no size dependency so long as the relevant part of the index fits into memory).

**Relational operations**

Queries made against the relational database, and the derived relvars in the database are expressed in a relational calculus or a relational algebra. In his original relational algebra, Codd introduced eight relational operators in two groups of four operators each. The first four operators were based on the traditional mathematical set operations:

The union operator combines the tuples of two relations and removes all duplicate tuples from the result. The relational union operator is equivalent to the SQL UNION operator.

The intersection operator produces the set of tuples that two relations share in common. Intersection is implemented in SQL in the form of the INTERSECT operator.

The difference operator acts on two relations and produces the set of tuples from the first relation that do not exist in the second relation. Difference is implemented in SQL in the form of the EXCEPT or MINUS operator.

The cartesian product of two relations is a join that is not restricted by any criteria, resulting in every tuple of the first relation being matched with every tuple of the second relation. The cartesian product is implemented in SQL as the CROSS JOIN operator.

The remaining operators proposed by Codd involve special operations specific to relational databases:

The selection, or restriction, operation retrieves tuples from a relation, limiting the results to only those that meet a specific criterion, i.e. a subset in terms of set theory. The SQL equivalent of selection is the SELECT query statement with a WHERE clause.

The projection operation extracts only the specified attributes from a tuple or set of tuples.

The join operation defined for relational databases is often referred to as a natural join. In this type of join, two relations are connected by their common attributes. SQL's approximation of a natural join is the INNER JOIN operator. In SQL, an INNER JOIN prevents a cartesian product from occurring when there are two tables in a query. For each table added to a SQL Query, one additional INNER JOIN is added to prevent a cartesian product. Thus, for N tables in a SQL query, there must be N-1 INNER JOINS to prevent a cartesian product.

The relational division operation is a slightly more complex operation, which involves essentially using the tuples of one relation (the dividend) to partition a second relation (the divisor). The relational division operator is effectively the opposite of the cartesian product operator (hence the name).

Other operators have been introduced or proposed since Codd's introduction of the original eight including relational comparison operators and extensions that offer support for nesting and hierarchical data, among others.

**Database normalization**

Database normalization is the process of organizing the fields and tables of a relational database to minimize redundancy and dependency. Normalization usually involves dividing large tables into smaller (and less redundant) tables and defining relationships between them. The objective is to isolate data so that additions, deletions, and modifications of a field can be made in just one table and then propagated through the rest of the database using the defined relationships.

Edgar F. Codd, the inventor of the relational model, introduced the concept of normalization and what we now know as the First Normal Form (1NF) in 1970.[1] Codd went on to define the Second Normal Form (2NF) and Third Normal Form (3NF) in 1971,[2] and Codd and Raymond F. Boyce defined the Boyce-Codd Normal Form (BCNF) in 1974.[3] Informally, a relational database table is often described as "normalized" if it is in the Third Normal Form.[4] Most 3NF tables are free of insertion, update, and deletion anomalies.

A standard piece of database design guidance is that the designer should first create a fully normalized design; then selective denormalization can be performed for performance reasons.[5]

**Database normalization Objectives**

A basic objective of the first normal form defined by Edgar Frank "Ted" Codd in 1970 was to permit data to be queried and manipulated using a "universal data sub-language" grounded in first-order logic.[6] (SQL is an example of such a data sub-language, albeit one that Codd regarded as seriously flawed.)[7]

The objectives of normalization beyond 1NF (First Normal Form) were stated as follows by Codd:

1. To free the collection of relations from undesirable insertion, update and deletion dependencies;

2. To reduce the need for restructuring the collection of relations, as new types of data are introduced, and thus increase the life span of application programs;

3. To make the relational model more informative to users;

4. To make the collection of relations neutral to the query statistics, where these statistics are liable to change as time goes by.

**SOFTWARE ENGINEERING**

If you’re in the industry or looking to get into the industry you’ll find yourself a little confused as to some of the labels and roles given to computer programmers.

* Programmer
* Developer
* Analyst
* Architect
* Engineer

The first two terms are freely interchangeable, the dictionary term for a programmer is *one who programs*, the term for developer is *one who develops*. So is there a difference in the terms or the roles? No effectively these two terms are identical and is merely a preference in terminology. Even Microsoft uses these terms interchangeably for their internal developers.

An analyst might not actually write code, in a lot of instances this will be a business analyst, someone who understands the business and also understands how to translate the business needs into terms that can be digested by a team of programmers or engineers. An analysts primary function is to write the specification or requirements documents required for a new project.

A software architect is very much like the traditional architect. They are responsible for building the framework and designing the tools that the team will be using to develop the project. Software architects are senior level developers who have had years of experience in developing software and can quickly turn a functional specification into a framework which is maintainable and future proof. A software architect can also be responsible for mentoring the rest of the development team in how to use the framework.

An engineer is none of the above and could be any of the above. A software engineer is someone that subscribes to an engineering *discipline*.

**Definition of software engineering** Software engineering is the *“systematic approach to the analysis, design, assessment, implementation, test, maintenance and reengineering of software, that is, the application of engineering to software”*

**A short story**

A software engineering graduate finishes university and starts developing software as an independent contractor, he’s smart and decides that his rates will be cheaper than everyone else’s as his costs are relatively low. Without knowing it he finds himself with a client, a small business that’s looking for a solution on the cheap. The first thing business asks for is a quote. So the graduate starts investigating with their database of choice and comes up with a quote.

The business receives the quote and then mulls it over for a week and then decides to go ahead. So the graduate gets out his tools and starts coding away. After a week he has a prototype up and working which is mostly functional to let the client see how it looks. The client reviews the prototype and says “great, I’ll take it straight away” and so the graduate grabs his software and trundles down to the business and installs the software on an old pc out the back. After he’s finished the business owner grabs the graduate and asks that he install the software on another machine, shrugging the graduate installs the software on a second pc, the graduate sticks around for a while and everyone is happy. The software is performing well, so he leaves feeling happy about the outcome.

The following day the graduate gets a phone call, the business owner is very angry and states that the software program doesn’t work and he’s losing business. The graduate is worried and so heads to the small business straight away, he walks in and investigates the software he installed on the machine out the back, everything is working fine. He then investigates the software on the other machine which he installed at the last minute. It’s working fine there. So what’s the problem?

After investigating and then discussing that he can’t find any issues with the software, the business owner reveals that he assumed that the software was supposed to be networked together. The graduate *face palms*, that wasn’t discussed as part of the original quote. And so the graduate goes away for another week after discussing how the software is supposed to work networked, and starts banging out code. One week later he finishes the networked version and runs some cursory tests and it works fine.

Eventually after much back and forth the graduate ends up with a functioning piece of software for the small business. The graduate was forced to under quote at every step of the way, the business lost revenue whenever the program encountered errors, eventually becoming angry and hiring another graduate to write another program.

So what went wrong?

* Was it the code being full of bugs?
* The business not telling the graduate up front about all of the requirements?
* Was it the quoting process?
* Lack of experience?
* *All of the above.*

Software engineering could have saved the graduate from making all of those mistakes.

Software engineering is a relatively new discipline in comparison with say *civil engineering*. As a species we have been building bridges for thousands upon thousands of years. We’ve been building software for less than a hundred years, so it’s a zygote in comparison to a fully grown adult.

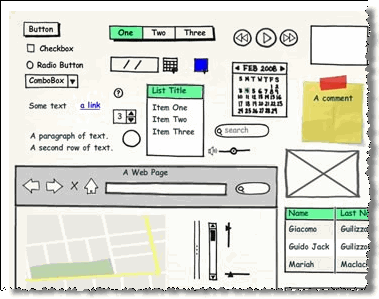
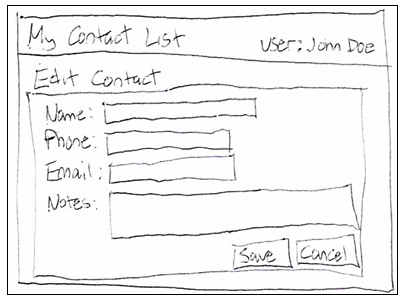
Software engineering is the discipline of how to write code which is testable (Unit Testable), is loosely coupled (Dependency Injection), well designed (Domain Driven Design), and quick to implement new features (Agile).



Even for professionals, we don’t get it right every time. We like to take shortcuts to save time and save the customer money, but we end up paying for those mistakes later on.

**What a software engineer would have done**

A software engineer works as a contractor for a firm, the software engineer is expensive to hire and takes a long time to come up with a solution. The engineer goes to the clients premises and takes his notebook and starts making notes and drawing diagrams. He then begins interviewing the client and asking questions in regards to how the business works, what are their current processes, what they are looking to achieve with the software they wish to write.

The engineer then draws up a functional specification including data flow diagrams and begins work on a throw away prototype. Using the throw away prototype he gathers the clients together and the clients discuss how the software should look and function. Several iterations of prototyping comes up with a solution the clients are happy with. Through the prototyping process the engineer uncovers that the software needs to be networked, it needs to handle GST, large sums of money, BPAY, Credit Cards, interface with an existing Point of Sale system and micro-transactions.

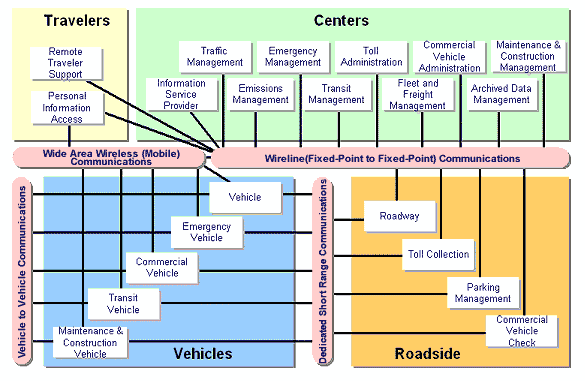
The software engineer finalises his functional specification and starts working on a quote based on those requirements. The quote is hedged as some of the estimates have a 50%-100% inaccuracy as to how long they will take to implement and test. The engineer hands the quote to the client and gets paid for the time spent on site.

The client is impressed with the Software Engineer as they have seen the prototype and can’t wait to use the final product. Even though they software engineer is far more expensive than the graduate. And spent 4 weeks onsite without writing any code, the client has the functional specification with screen shots and accurate descriptions of all the functionality that they wished for, and the process weeded out requirements that the clients never knew they needed.

The clients hire the Software Engineering firm to produce the final software product. The Engineer goes back to the firm and begins to work on the framework to the software program, after about a week he has the foundation produced which has 100% test coverage and will meet all of the requirements that the clients are looking for, except he needs some help filling in the finer details. The basic splashes of colour have been added to the canvas, the lines have all been drawn and inked, now all that is needed is for fine detail to be done.

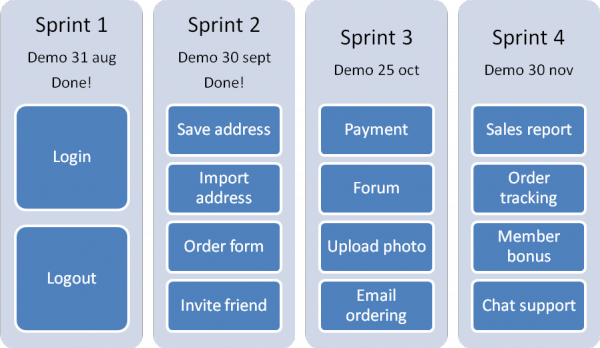
The Engineer gathers his team of juniors and graduates, he also grabs a graphic designer, and 1-2 testers from Quality Assurance. The meeting consists of discussing the Functional Requirements. Who is the client, what is the software supposed to do. The graphic designer and the QA team ask a few questions and then leave the meeting. The graphic designer have been given the approved wire frames from the client and is starting to do the finalised UI Design.

The graduates and juniors continue the meeting and start discussing the framework. A couple of days pass and now the team knows how the framework is going to work and how the framework is going to meet the requirements.



The team then contacts the project owner (a representative of the client). The team will be doing development in a rapid 2 week sprint. Before the first sprint can begin they need to determine what functionality will be in the first sprint. The product after each sprint will be a small feature set of the final product but each feature will be fully tested and fleshed out. After discussing the list of requirements the team and client agrees on 12 or so items that should be worked on for the first sprint, 3 of which will take more than 2 weeks to perform and will not be in the testable product at the end of the sprint but however it will be worked on as a priority as it will be mandatory for the next sprint.

The tasks are divided up and given to individual team members, which they go off and start writing unit tests which describe the functions that they’re about to write. The next day the Engineer pulls the team aside and have a quick stand up meeting to discuss their progress. Tasks are reassigned and development continues. 2 days before the sprint is set to finish the Software Engineer calls a stop work on new functionality. He reviews all current features which have been completed and reviews their test coverage (Does this test accurately reflect the requirement, does the function fulfil the requirement), after the review the Software Engineer drops 2 features as they do not pass the initial inspection. The rest are merged into a test build.



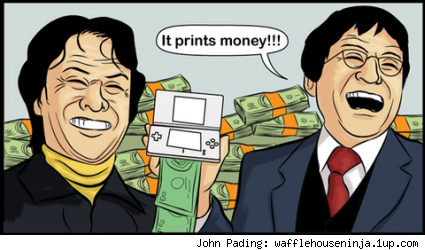
The Quality Assurance team is then called to action, they review the test build against the requirements and determine any bugs that they find or any requirements that have not been met with the functionality that has just been implemented. 2 more features have been found to be defective, these features are dropped from the final product and then will be fixed in the next sprint.

The first sprint is over, 12 features are fully tested and go into the first sprint to be shown to the customer, 7 features are dropped from the sprint to be worked on for the second sprint. The client is given access to the first iteration of the software product, each feature that is shown is fully working and passed internal quality assurance.

The client is impressed. This is exactly how the program was described, and it only took 2 weeks to get a start on the software product. This process goes on for 2 more sprints and in the final sprint the client shows an Alpha version of the software product to employees to have a play with. After using the software they come up with new features that they want. The 2 features are easily catered for and are included in the final sprint, 1 of the new features was not easily catered for and required an additional quote or estimate. The quote was drawn up and an accurate estimate given – this is accurate NOW because the framework is fully fleshed out, the design is fully qualified and the team members know the ins and outs of the framework. The client weighs their options and decide not the go with the new feature that was quoted.

However after 4 sprints (2 months) the clients have seen evidence of fast progress, they have used the software program, the engineers have implemented new features very quickly (because their tests indicated to them what features were broken due to new functionality), and the software program is fully tested to work in a production environment.

The client was extremely happy with the final product even though the initial lead up to the final product took longer than expected, the final results were outstanding. The client made millions from the software product, even though they paid more for the product up front.



So why doesn’t everyone do this?

Let’s face it, Software Engineering is Hard

One of the main reasons why people don’t follow software engineering principles or misapply them is due to the illusion that by *NOT* following those principles they will save themselves and their client’s money.

**Why do these failures occur?**

It’s difficult to answer and there is *NO SILVER BULLET*[and no magical solution to stop projects from failing. The best we can do is to increase communication with the client. 20 years ago software engineering was all about fully gathering as many requirements as possible and then fully fleshing out a design and then telling a programmer what to do, like a Cog in a machine.

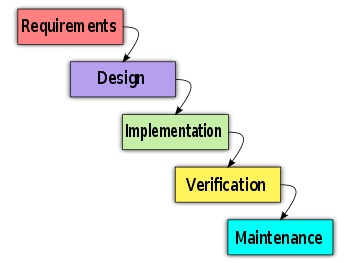
*It didn’t work then, and it doesn’t work now*

We’ve learned from those mistakes and have developed new practices to help avoid the problems we had with the Waterfall method of software production. Waterfall is expensive, and produces obsolete and ponderous software. A software product is only as good as its fully tested and qualified feature set.

**Waterfall Method**

Take the example above Company B is a Software Engineering firm also, but is using old software engineering techniques which have been proven to be expensive and slow (Waterfall Method).

The client decides to go with Company B. Using waterfall they gather all the requirements up front hire an analyst, and draw up a fully fleshed out diagram of the process of the entire software product including hiring designers to fully flesh out and draw images of the final product. It takes 6 months before they even hire a programmer to turn that fully fleshed out specification into code.



Sound Familiar?

Using the example above using new Software Engineering methodologies (Agile, SCRUM, Test Driven Development, Domain Driven Design, Design Patterns, Peer Reviewing, Quality Assurance). The entire process is fully done in 6 months.

When Waterfall is better than Agile

In the above instance is biased towards Agile for a singular type of solution (mainly web). Where most web applications are User Interface driven.

There are strong cases where the waterfall method is actually better than agile, and should be adopted instead. There are drawbacks to the agile software development methodology.

* Requires a good experienced team of developers
* Requires a client who is open to collaborating on the software project
* Requires a software product which has a User Interface for the client to use
* Requires more discipline to do documentation, agile does not mean *No Documentation*

Where waterfall is best used where an agile approach is simply *Not Suitable*. For instance writing COM Services, or server applications with no client facing UI.

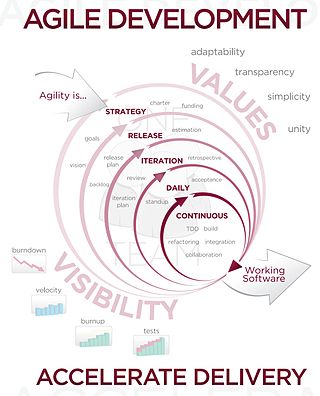
When Agile should be used over Waterfall

Agile has its drawbacks, as does waterfall. Agile is best used with a product which has a strong UI focus, as it requires client interaction to help define the requirements.

The main reason it’s called agile is because *in comparison* to waterfall methodology. You can change requirements half way through implementing the software program and it will *cost you less* to implement. This is because using waterfall, if a requirement has changed, the entire process is scrapped and starts from square 1. Re doing the fully fleshed out specification document, and then changes to code becomes a nightmare because the framework has been written for the first version was designed to work with the first version and will require a complete re-write to work with the second version… it is very rigid.

The agile version doesn’t make assumptions nor holds anything in concrete, it is designed to be able to be changed between sprints, the requirements are not set in stone and so time is not wasted on fully fleshing out exactly how the software program is going to work in every minute detail. Which saves the client time and money on wasted effort.

You will notice however that even describing the agile method it is far more involved than when I described how the graduate (who used no process) performed the same job. Using no process in the end took the graduate longer than what he initially estimated, cost the client business, and cost the client more money than what they originally estimated.



**Software Engineering is expensive.**

Let’s not be under any illusions here, software engineering is expensive, and old software engineering is even more expensive. However when its done right using what we’ve learned in the past 20 years it can also be done quickly and efficiently with guaranteed results.

**OFFICE AUTOMATION AND COLLABORATION**

Office automation systems are computer based information systems that collect, process, store and transmit electronic messages, documents and other forms of communications among individuals, work groups and organizations. Such systems can increase the productivity of managerial end users and other professional and staff personnel by significantly reducing the time and effort needed to produce, access and receive business communications.

An overview of office automation systems are given below:

1) Office publishing systems are word processing and desktop publishing systems.

2) Image processing systems are image processing, optical scanning and storage, document management and interactive video systems.

3) Electronic communication systems are electronic meeting, tele-conferencing and telecommunicating systems.

Office support systems are electronic calendar, ticket file, notebook, directory system, work scheduling and task management systems. The above are described in brief:

### Office Publishing Systems

Word processing - It is the use of computer systems to automate the transformation of ideas into a readable form of communication. It involves manipulating text data to produce information products in the form of documents (letters, memos, forms and reports).

Desktop publishing - Organisations can use and develop publishing systems to produce their own printed materials. They can design and print their own newsletters, manuals, brochures and books with several types of styles and graphics and colours on each page.

Voice store and forward systems - It is also known as voice mail or structured verbal-response exchanges. It has the capability of storing verbal data and/or communicating it to another location. It is one of the latest systems and found useful to corporate houses and the society at large.

Electronic Meeting systems - If electronic meeting systems are used, people may not appreciate spending money on travel to attend meetings away from their normal work location. It involves the use of video and audio communications to allow conferences and meetings to be held with participants who may be scattered across a room, a building or a country.

Teleconferencing - Participants at remote sites key in their presentation and responses whenever convenient for their online terminals or workstations which are connected to a central conference computer. Since all participants don't have to do this at the same time, this form of EMS is called computer's conferencing. Sessions are held in real time with major participants at remote site who take part with voice input of questions and responses.

Telecommuting - It is the use of telecommunications by workers to replace commuting enabling them to work from their homes. It is also used to describe the use of telecommunication to carry on work activities from temporary locations other than offices and homes. Some people consider telecommuting as the creation of virtual offices.

Multimedia - Another evolving set of technologies for systematically managing graphics, drawings and photographs in addition to text and data are multimedia datamanagement systems . Some of these are called hypermedia systems.

### Image Processing Systems

Image processing systems, also known as electronic image management system, allow users to electronically capture, store, process and retrieve images of documents. Image databasemanagement systems are becoming more versatile. The LAN-based image processing systems are also being common, which include several servers each dedicated to a specific function.

### Document Management Systems

These systems are sophisticated image processing systems. These often interface with other office systems such as word processing, desktop publishing, electronic mail, voice mail etc.

These systems allow digitized ‘voice notes' to be attached to specific document images and may have other special features as well as the ability to store multi-colour documents and colour photographs.

### The Automated Office

Office automation is defined as a separate sub-system or included within information processing. It includes a wide range of support facilities for knowledge work and activities.

**COLLABORATION SYSTEM**

A Collaboration system is designed for one basic purpose, to help unite employees or people that are working on a similar task, or it could be the exact same task, and the system helps unite them to complete their task and achieve whatever goal that task sets out to do. It is basically a multi user program, meaning that more than one person could use it. What are some examples of collaboration systems, and what history does this specific system have? Now that you know what the definition of a collaboration system is, let us look at the history of the collaboration system.

    This system can be pin-pointed to about the 1980's, where where a computer engineer named Richman and a software designer names Slovak, said that group works needed to be bind together so they could work and get projects down. Thats the quick version of what these two said, but they began to work on a system that could tie the work of multiple employees in different places, computers, and different hours could put there work together and not have issues, this system also works on the same computer, but employees can log on at different times and continue their work. Well in 1990's that system was in full affect and companies like IBM were designing new and better collaboration systems, and companies like boeing began using this system globally. How does this system help and what are some examples of this system in action? let us take a look.

    Instant messaging is a very popular computer/internet program that is offered by [http://Yahoo.com/IM](http://yahoo.com/IM), Google, and AOL, just to mention a few. Well the instant messaging system is also a collaboration system, in the sense that employees can chat for free to get a task or work done, i myself used the IM in school while working on a project in a group. This is an example of a colaboration system, another is the collaboration system used by oracle that recently saved the company 5,683,000 dollars, small amount if you think about the amount of revenues oracle makes annually, but systems like collaboration system can save companies substantial amounts of money when two or more systems are used, but where the real money is saved apart from the 5.6 million dollars is in communication costs and messaging that saved oracle in 2000's a great amount of money. more information can be found on <http://oracle.com/infoapps> .

    I hope this information could help you understand what a Collaboration system is, and how it works. In conclusion, a collaboration system is a system that unites many users in order to communicate and efficiently achieve their work, task, and/or goals that are being worked on. Just to emphasize a bit more, this support site is sort of a collaboration system as well, because all of us students contribute to this site through id names and pass codes and have adding our part without damaging another’s work.

A collaboration system is an IT based set of tools that that create a workflow of information to specific teams and the members.  This allows individual to share ideas and their talents with other members so that the task can be finished both efficiently and effectively.

There are numerous factors that influence a collaboration system, but there are two fundamental aspects that not only need to be customizes according to the goals, but need to be taken into consideration when it comes to corporate culture.  The two are:

*Unstructured collaboration*- chasing answers to the unknown questions, using IT tools to share information about the problems at stake, increasing personal productivity

*Structured collaboration*- sharing common knowledge, written rules, structured and set workflow that does not change

Different collaborations for different corporate cultures

In order for the employees to stay motivated, there has to be a perfect fit in regard to which collaboration technique would be implemented in order to create a [collaborative working environment](http://en.wikipedia.org/wiki/Collaborative_working_environment). For example, a structured collaboration would not fit in the Google corporate culture because Google is very innovative and feed off of new ideas and challenging unanswered questions.  With a structured collaboration, innovation would be non-existent because of the boundary of common knowledge.

On the other hand, a structured collaboration model would fit well under a manufacturing company like Nummi because constant workflow is critical to the organizations output.  A well organized structure would fit well with a structured collaboration model because it’s hierarchy based and needs leadership to keep the flow moving.

In addition to corporate culture, the structure of the firm needs to be evaluated first to get a better understanding of which collaboration model would best fit.  For example, if a firm has an organic structure with no managerial hierarchy, then an unstructured collaboration would be best because this model fosters innovation in a group setting with without any power struggle.

Positive aspects to structured collaboration

-easier to organize

-excellent for hierarchal structured organizations

-same process that increases proficiency

-no contradictory in relation to information

-all members on the team understand and acknowledge their position

Limitations to structured collaboration

-Does not foster innovation

-Same workflow information with no variance at all

-Can cause groupthink

-Encourages lack of creativity

-Intended for repetitive industry (i.e. manufacturing)

-Needs to me managed

Examples of collaboration tools

One example of a collaboration tool is the “collaborative web based mind mapping”.  What this allows individuals to do is to organize ideas while numerous group members can change anything on the map at the same time.

This tool will most likely benefit an unstructured collaboration because this tool fosters the ability to create innovative ideas.  This [IntraLinks](http://www.intralinks.com) helps not only teams to collect and categorize ideas, this can also help the team see a different perspective that would not readily been seen otherwise.

=========================================================================

**Malware**, short for malicious software, is software used to disrupt computer operation, gather sensitive information, or gain access to private computer systems.[1] It can appear in the form of code, scripts, active content, and other software.[2] 'Malware' is a general term used to refer to a variety of forms of hostile or intrusive software.[3] In all countries it is a serious criminal offence to create and distribute malware, but it continues to be produced for various reasons, such as demonstrating a capability or making money.[citation needed]

Malware includes computer viruses, ransomware, worms, trojan horses, rootkits, keyloggers, dialers, spyware, adware, malicious BHOs, rogue security software and other malicious programs; the majority of active malware threats are usually worms or trojans rather than viruses.[4] In law, malware is sometimes known as a computer contaminant, as in the legal codes of several U.S. states.[5][6] Malware is different from defective software, which is a legitimate software but contains harmful bugs that were not corrected before release. However, some malware is disguised as genuine software, and may come from an official company website in the form of a useful or attractive program which has the harmful malware embedded in it along with additional tracking software that gathers marketing statistics.[7]

Software such as anti-virus, anti-malware, and firewalls are relied upon by users at home, small and large organizations around the globe to safeguard against malware attacks which helps in identifying and preventing the further spread of malware in the network.

**Purposes**

Many early infectious programs, including the first Internet Worm, were written as experiments or pranks. Today, malware is used primarily to steal sensitive information of personal, financial, or business importance by black hat hackers with harmful intentions.[9]

Malware is sometimes used broadly against government or corporate websites to gather guarded information,[10] or to disrupt their operation in general. However, malware is often used against individuals to gain personal information such as social security numbers, bank or credit card numbers, and so on. Left unguarded, personal and networked computers can be at considerable risk against these threats. (These are most frequently counter-acted by various types of firewalls, anti-virus software, and network hardware).[11]

Since the rise of widespread broadband Internet access, malicious software has more frequently been designed for profit. Since 2003, the majority of widespread viruses and worms have been designed to take control of users' computers for black-market exploitation.[12] Infected "zombie computers" are used to send email spam, to host contraband data such as child pornography,[13] or to engage in distributed denial-of-service attacks as a form of extortion.[14]

Another strictly for-profit category of malware has emerged, called spyware. These programs are designed to monitor users' web browsing, display unsolicited advertisements, or redirect affiliate marketing revenues to the spyware creator. Spyware programs do not spread like viruses; instead they are generally installed by exploiting security holes. They can also be packaged together with user-installed software, such as peer-to-peer applications.[15]

**Proliferation**

Preliminary results from Symantec published in 2008 suggested that "the release rate of malicious code and other unwanted programs may be exceeding that of legitimate software applications."[16] According to F-Secure, "As much malware [was] produced in 2007 as in the previous 20 years altogether."[17] Malware's most common pathway from criminals to users is through the Internet: primarily by e-mail and the World Wide Web.[18]

The prevalence of malware as a vehicle for Internet crime, along with the challenge of anti-malware software to keep up with the continuous stream of new malware, has seen the adoption of a new mindset for individuals and businesses using the Internet. With the amount of malware currently being distributed, some percentage of computers will always be infected. For businesses, especially those that sell mainly over the Internet, this means they need to find a way to operate despite security concerns. The result is a greater emphasis on back-office protection designed to protect against advanced malware operating on customers' computers.[19] A 2013 Webroot study shows that 64% of companies allow remote access to servers for 25% to 100% of their workforce and that companies with more than 25% of their employees accessing servers remotely have higher rates of malware threats.[20]

On March 29, 2010, Symantec Corporation named Shaoxing, China, as the world's malware capital.[21] A 2011 study from the University of California, Berkeley, and the Madrid Institute for Advanced Studies published an article in Software Development Technologies, examining how entrepreneurial hackers are helping enable the spread of malware by offering access to computers for a price. Microsoft reported in May 2011 that one in every 14 downloads from the Internet may now contain malware code. Social media, and Facebook in particular, are seeing a rise in the number of tactics used to spread malware to computers.[22]

Recent studies show that malware evolution is tending towards a mobile nature due to the popularity of devices such as smartphones.[23]

**Infectious malware: viruses and worms**

The best-known types of malware, viruses and worms, are known for the manner in which they spread, rather than any specific types of behavior. The term computer virus is used for a program that has infected some executable software and, when run, causes the virus to spread to other executables. On the other hand, a worm is a program that actively transmits itself over a network to infect other computers. These definitions lead to the observation that a virus requires user intervention to spread, whereas a worm spreads itself automatically.[24]

Using this distinction, infections transmitted by email or Microsoft Word documents, which rely on the recipient opening a file or email to infect the system, would be classified as viruses rather than worms.

**Trojan horses**

For a malicious program to accomplish its goals, it must be able to run without being detected, shut down, or deleted. When a malicious program is disguised as something normal or desirable, users may willfully install it without realizing it. This is the technique of the Trojan horse or trojan. In broad terms, a Trojan horse is any program that invites the user to run it, concealing harmful or malicious code. The code may take effect immediately and can lead to many undesirable effects, such as deleting the user's files or installing additional harmful software.[citation needed]

One of the most common ways that spyware is distributed is as a Trojan horse, bundled with a piece of desirable software that the user downloads from the Internet. When the user installs the software, the spyware is installed along with it. Spyware authors who attempt to act in a legal fashion may include an end-user license agreement that states the behavior of the spyware in loose terms, which users may not read or understand.[citation needed]

**Rootkits**

Once a malicious program is installed on a system, it is essential that it stays concealed, to avoid detection. Software packages known as rootkits allow this concealment, by modifying the host's operating system so that the malware is hidden from the user. Rootkits can prevent a malicious process from being visible in the system's list of processes, or keep its files from being read.[25]

Some malicious programs contain routines to defend against removal, not merely to hide themselves. An early example of this behavior is recorded in the Jargon File tale of a pair of programs infesting a Xerox CP-V time sharing system:

Each ghost-job would detect the fact that the other had been killed, and would start a new copy of the recently-stopped program within a few milliseconds. The only way to kill both ghosts was to kill them simultaneously (very difficult) or to deliberately crash the system.

**Backdoors**

A backdoor is a method of bypassing normal authentication procedures. Once a system has been compromised, one or more backdoors may be installed in order to allow easier access in the future.[27] Backdoors may also be installed prior to malicious software, to allow attackers entry.

The idea has often been suggested that computer manufacturers preinstall backdoors on their systems to provide technical support for customers, but this has never been reliably verified. Crackers typically use backdoors to secure remote access to a computer, while attempting to remain hidden from casual inspection. To install backdoors crackers may use Trojan horses, worms, or other methods.[citation needed]

**Vulnerability to malware**

In this context, as throughout, it should be borne in mind that the “system” under attack may be of various types, e.g. a single computer and operating system, a network or an application.

Various factors make a system more vulnerable to malware:

**Security defects in software**

Malware exploits security defects (security bugs, or vulnerabilities) in the design of the operating system, in applications (such as browsers—avoid using Internet Explorer 8 or earlier, e.g. on Windows XP[28]), or in (old versions of) browser plugins such as Adobe Flash Player, Adobe Acrobat / Reader, or Java (see Java SE critical security issues).[29][30] Sometimes even installing new versions of such plugins does not automatically uninstall old versions. Security advisories from such companies announce security-related updates.[31] Common vulnerabilities are assigned CVE IDs and listed in the US National Vulnerability Database. Secunia PSI[32] is an example of software, free for personal use, that will check a PC for vulnerable out-of-date software, and attempt to update it.

Most systems contain bugs, or loopholes, which may be exploited by malware. A typical example is a buffer-overrun vulnerability, in which an interface designed to store data, in a small area of memory, allows the caller to supply more data than will fit. This extra data then overwrites the interface's own executable structure (past the end of the buffer and other data). In this manner, malware can force the system to execute malicious code, by replacing legitimate code with its own payload of instructions (or data values) copied into live memory, outside the buffer area.

**Insecure design or user error**

Originally, PCs had to be booted from floppy disks. Until recently, it was common for a computer to boot from an external boot device by default. This meant that the computer would, by default, boot from a floppy disk, USB flash drive, or CD—and malicious boot code could be used to install malware or boot into a modified operating system. Autorun or autoplay features may allow code to be automatically executed from a floppy disk, CD-ROM or USB device with or without the user’s permission. Older email software would automatically open HTML email containing malicious JavaScript code; users may also unwarily open (execute) malicious email attachments.[citation needed]

**Over-privileged users and over-privileged code**

Over-privileged users: some systems allow all users to modify their internal structures. This was the standard operating procedure for early microcomputer and home computer systems, where there was no distinction between an Administrator or root, and a regular user of the system. In some systems, non-administrator users are over-privileged by design, in the sense that they are allowed to modify internal structures of the system. In some environments, users are over-privileged because they have been inappropriately granted administrator or equivalent status.

Over-privileged code: some systems allow code executed by a user to access all rights of that user. Also standard operating procedure for early microcomputer and home computer systems. Malware, running as over-privileged code, can use this privilege to subvert the system. Almost all currently popular operating systems, and also many scripting applications allow code too many privileges, usually in the sense that when a user executes code, the system allows that code all rights of that user. This makes users vulnerable to malware in the form of e-mail attachments, which may or may not be disguised.

**Use of the same operating system**

Homogeneity: e.g. when all computers in a network run the same operating system; upon exploiting one, one worm can exploit them all:[33] For example, Microsoft Windows or Mac OS X have such a large share of the market that concentrating on either could enable an exploited vulnerability to subvert a large number of systems. Instead, introducing diversity, purely for the sake of robustness, could increase short-term costs for training and maintenance. However, having a few diverse nodes would deter total shutdown of the network, and allow those nodes to help with recovery of the infected nodes. Such separate, functional redundancy could avoid the cost of a total shutdown.

**Anti-malware strategies**

As malware attacks become more frequent, attention has begun to shift from viruses and spyware protection, to malware protection, and programs that have been specifically developed to combat malware. (Other preventive and recovery measures, such as backup and recovery methods, are mentioned in the computer virus article).

**Anti-virus and anti-malware software**

A specific component of the Anti virus and anti-malware software commonly referred as the on-access or real-time scanner, hooks deep into the operating system's core or kernel functions in a manner similar to how certain malware itself would attempt to operate, though with the user's informed permission for protecting the system. Any time the operating system accesses a file, the on-access scanner checks if the file is a 'legitimate' file or not. If the file is considered a malware by the scanner, the access operation will be stopped, the file will be dealt by the scanner in pre-defined way (how the Anti-virus program was configured during/post installation) and the user will be notified. This may considerably slow down the operating system depending on how well the scanner was programmed. The goal is to stop any operations the malware may attempt on the system before they occur, including activities which might exploit bugs or trigger unexpected operating system behavior.[citation needed]

Anti-malware programs can combat malware in two ways:

They can provide real time protection against the installation of malware software on a computer. This type of malware protection works the same way as that of antivirus protection in that the anti-malware software scans all incoming network data for malware and blocks any threats it comes across.

Anti-malware software programs can be used solely for detection and removal of malware software that has already been installed onto a computer. This type of anti-malware software scans the contents of the Windows registry, operating system files, and installed programs on a computer and will provide a list of any threats found, allowing the user to choose which files to delete or keep, or to compare this list to a list of known malware components, removing files that match.[citation needed]

Real-time protection from malware works identically to real-time antivirus protection: the software scans disk files at download time, and blocks the activity of components known to represent malware. In some cases, it may also intercept attempts to install start-up items or to modify browser settings. Because many malware components are installed as a result of browser exploits or user error, using security software (some of which are anti-malware, though many are not) to "sandbox" browsers (essentially isolate the browser from the computer and hence any malware induced change) can also be effective in helping to restrict any damage done.[citation needed]

Examples of Microsoft Windows anti virus and anti-malware software include the optional Microsoft Security Essentials[34] (for Windows XP, Vista and Windows 7) for real-time protection, the Windows Malicious Software Removal Tool[35] (now included with Windows (Security) Updates on "Patch Tuesday", the second Tuesday of each month), and Windows Defender (an optional download in the case of Windows XP).[36] Additionally, several capable antivirus software programs are available for free download from the Internet (usually restricted to non-commercial use).[37] A test has found a free program to be competitive with commercial competitors.[38] Microsoft's System File Checker can be used to check for and repair corrupted system files.

Some viruses disable System Restore and other important Windows tools such as Task Manager and Command Prompt. Many such viruses can be removed by rebooting the computer, entering Windows safe mode with networking,[39] and then using system tools or Microsoft Safety Scanner.[40]

**Known good**

Typical malware products detect issues based on heuristics or signatures – i.e., based on information that can be assessed to be bad. Some products[41][42] take an alternative approach when scanning documents such as Word and PDF, by regenerating a new, clean file, based on what is known to be good from schema definitions of the file (a patent for this approach exists).[43]

**Website security scans**

As malware also harms the compromised websites (by breaking reputation, blacklisting in search engines, etc.), some websites offer vulnerability scanning. [44] [45] [46] [47] Such scans check the website, detect malware, may note outdated software, and may report known security issues.

**Eliminating over-privileged code**

Over-privileged code dates from the time when most programs were either delivered with a computer or written in-house, and repairing it would serve to render most antivirus software essentially redundant. It would, however, have appreciable consequences for the user interface and system management.[citation needed]

The system would have to maintain privilege profiles, and know which to apply for each user and program.[citation needed]

In the case of newly installed software, an administrator would need to set up default profiles for the new code.[citation needed]

Eliminating vulnerability to rogue device drivers is probably harder than for arbitrary rogue executable. Two techniques, used in VMS, that can help are memory mapping only the registers of the device in question and a system interface associating the driver with interrupts from the device.[citation needed]

**Grayware**

Grayware (or greyware) is a general term that refers to applications or files that are not directly classified as malware (like worms or trojan horses), but can still negatively affect the performance of computers and involve significant security risks.[27]

It describes applications that behave in an annoying or undesirable manner, and yet are less serious or troublesome than malware. Grayware encompasses spyware, adware, dialers, joke programs, remote access tools and any other program apart from a virus, that is designed to harm the performance of computers. The term is in use since around 2004.[48]

Another term, PUP, which stands for Potentially Unwanted Program, refers to applications that would be considered unwanted despite having been downloaded by the user (users may fail to read a download agreement). PUPs include spyware, adware, and dialers.

**History of viruses and worms**

Before Internet access became widespread, viruses spread on personal computers by infecting the executable boot sectors of floppy disks. By inserting a copy of itself into the machine code instructions in these executables, a virus causes itself to be run whenever a program is run or the disk is booted. Early computer viruses were written for the Apple II and Macintosh, but they became more widespread with the dominance of the IBM PC and MS-DOS system. Executable-infecting viruses are dependent on users exchanging software or boot-able floppies and thumb drives so they spread rapidly in computer hobbyist circles.[citation needed]

The first worms, network-borne infectious programs, originated not on personal computers, but on multitasking Unix systems. The first well-known worm was the Internet Worm of 1988, which infected SunOS and VAX BSD systems. Unlike a virus, this worm did not insert itself into other programs. Instead, it exploited security holes (vulnerabilities) in network server programs and started itself running as a separate process. This same behavior is used by today's worms as well.

With the rise of the Microsoft Windows platform in the 1990s, and the flexible macros of its applications, it became possible to write infectious code in the macro language of Microsoft Word and similar programs. These macro viruses infect documents and templates rather than applications (executables), but rely on the fact that macros in a Word document are a form of executable code.[citation needed]

Today, worms are most commonly written for the Windows OS, although a few like Mare-D[49] and the Lion worm[50] are also written for Linux and Unix systems. Worms today work in the same basic way as 1988's Internet Worm: they scan the network and use vulnerable computers to replicate. Because they need no human intervention, worms can spread with incredible speed. The SQL Slammer infected thousands of computers in a few minutes.[51]

=========================================================================

**Information security**, sometimes shortened to InfoSec, is the practice of defending information from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction. It is a general term that can be used regardless of the form the data may take (electronic, physical, etc...)[1]

Two major aspects of information security are:

IT security: Sometimes referred to as computer security, Information Technology Security is information security applied to technology (most often some form of computer system). It is worthwhile to note that a computer does not necessarily mean a home desktop. A computer is any device with a processor and some memory (even a calculator). IT security specialists are almost always found in any major enterprise/establishment due to the nature and value of the data within larger businesses. They are responsible for keeping all of the technology within the company secure from malicious cyber attacks that often attempt to breach into critical private information or gain control of the internal systems.

Information assurance: The act of ensuring that data is not lost when critical issues arise. These issues include but are not limited to: natural disasters, computer/server malfunction, physical theft, or any other instance where data has the potential of being lost. Since most information is stored on computers in our modern era, information assurance is typically dealt with by IT security specialists. One of the most common methods of providing information assurance is to have an off-site backup of the data in case one of the mentioned issues arise.

Governments, military, corporations, financial institutions, hospitals, and private businesses amass a great deal of confidential information about their employees, customers, products, research and financial status. Most of this information is now collected, processed and stored on electronic computers and transmitted across networks to other computers.

Should confidential information about a business' customers or finances or new product line fall into the hands of a competitor or a black hat hacker, a business and its customers could suffer widespread, irreparable financial loss, not to mention damage to the company's reputation. Protecting confidential information is a business requirement and in many cases also an ethical and legal requirement.

For the individual, information security has a significant effect on privacy, which is viewed very differently in different cultures.

The field of information security has grown and evolved significantly in recent years. There are many ways of gaining entry into the field as a career. It offers many areas for specialization including securing network(s) and allied infrastructure, securing applications and databases, security testing, information systems auditing, business continuity planning and digital forensics, etc.

This article presents a general overview of information security and its core concepts.

**History**

Since the early days of writing, politicians, diplomats and military commanders understood that it was necessary to provide some mechanism to protect the confidentiality of correspondence and to have some means of detecting tampering. Julius Caesar is credited with the invention of the Caesar cipher ca. 50 B.C., which was created in order to prevent his secret messages from being read should a message fall into the wrong hands, but for the most part protection was achieved through the application of procedural handling controls. Sensitive information was marked up to indicate that it should be protected and transported by trusted persons, guarded and stored in a secure environment or strong box. As postal services expanded, governments created official organisations to intercept, decipher, read and reseal letters (e.g. the UK Secret Office and Deciphering Branch in 1653).

In the mid 19th century more complex classification systems were developed to allow governments to manage their information according to the degree of sensitivity. The British Government codified this, to some extent, with the publication of the Official Secrets Act in 1889. By the time of the First World War, multi-tier classification systems were used to communicate information to and from various fronts, which encouraged greater use of code making and breaking sections in diplomatic and military headquarters. In the United Kingdom this led to the creation of the Government Code and Cypher School in 1919. Encoding became more sophisticated between the wars as machines were employed to scramble and unscramble information. The volume of information shared by the Allied countries during the Second World War necessitated formal alignment of classification systems and procedural controls. An arcane range of markings evolved to indicate who could handle documents (usually officers rather than men) and where they should be stored as increasingly complex safes and storage facilities were developed. Procedures evolved to ensure documents were destroyed properly and it was the failure to follow these procedures which led to some of the greatest intelligence coups of the war (e.g. U-570).

The end of the 20th century and early years of the 21st century saw rapid advancements in telecommunications, computing hardware and software, and data encryption. The availability of smaller, more powerful and less expensive computing equipment made electronic data processing within the reach of small business and the home user. These computers quickly became interconnected through the Internet.

The rapid growth and widespread use of electronic data processing and electronic business conducted through the Internet, along with numerous occurrences of international terrorism, fueled the need for better methods of protecting the computers and the information they store, process and transmit. The academic disciplines of computer security and information assurance emerged along with numerous professional organizations – all sharing the common goals of ensuring the security and reliability of information systems.

**Definitions**

The definitions of InfoSec suggested in different sources are summarised below (adopted from [2]).

1. "Preservation of confidentiality, integrity and availability of information. Note: In addition, other properties, such as authenticity, accountability, non-repudiation and reliability can also be involved." (ISO/IEC 27000:2009) [3]

2. "The protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to provide confidentiality, integrity, and availability." (CNSS, 2010) [4]

3. "Ensures that only authorized users (confidentiality) have access to accurate and complete information (integrity) when required (availability)." (ISACA, 2008) [5]

4. "Information Security is the process of protecting the intellectual property of an organisation." (Pipkin, 2000) [6]

5. "...information security is a risk management discipline, whose job is to manage the cost of information risk to the business." (McDermott and Geer, 2001)[7]

6. "A well-informed sense of assurance that information risks and controls are in balance." (Anderson, J., 2003)[8]

7. "Information security is the protection of information and minimises the risk of exposing information to unauthorised parties." (Venter and Eloff, 2003) [9]

8. "Information Security is a multidisciplinary area of study and professional activity which is concerned with the development and implementation of security mechanisms of all available types (technical, organisational, human-oriented and legal) in order to keep information in all its locations (within and outside the organisation’s perimeter) and, consequently, information systems, where information is created, processed, stored, transmitted and destructed, free from threats. Threats to information and information systems may be categorised and a corresponding security goal may be defined for each category of threats. A set of security goals, identified as a result of a threat analysis, should be revised periodically to ensure its adequacy and conformance with the evolving environment. The currently relevant set of security goals may include: confidentiality, integrity, availability, privacy, authenticity & trustworthiness, non-repudiation, accountability and auditability." (Cherdantseva and Hilton, 2013)[2]

**Profession**

Information security is a stable and growing profession – Information security professionals are very stable in their employment; more than 80 percent had no change in employer or employment in the past year, and the number of professionals is projected to continuously grow more than 11 percent annually over the next five years.[10]

**Basic principles**

**Key concepts**

The CIA triad (confidentiality, integrity and availability) is one of the core principles of information security.[11] (The members of the classic InfoSec triad -confidentiality, integrity and availability - are interchangeably referred to in the literature as security attributes, properties, security goals, fundamental aspects, information criteria, critical information characteristics and basic building blocks.) There is continuous debate about extending this classic trio.[2][citation needed] Other principles such as Accountability[12] have sometimes been proposed for addition – it has been pointed out[citation needed] that issues such as Non-Repudiation do not fit well within the three core concepts, and as regulation of computer systems has increased (particularly amongst the Western nations) Legality is becoming a key consideration for practical security installations.[citation needed]

In 1992 and revised in 2002 the OECD's Guidelines for the Security of Information Systems and Networks[13] proposed the nine generally accepted principles: Awareness, Responsibility, Response, Ethics, Democracy, Risk Assessment, Security Design and Implementation, Security Management, and Reassessment. Building upon those, in 2004 the NIST's Engineering Principles for Information Technology Security[14] proposed 33 principles. From each of these derived guidelines and practices.

In 2002, Donn Parker proposed an alternative model for the classic CIA triad that he called the six atomic elements of information. The elements are confidentiality, possession, integrity, authenticity, availability, and utility. The merits of the Parkerian hexad are a subject of debate amongst security professionals.[citation needed]

In 2013, based on the extensive literature analysis, the Information Assurance & Security (IAS) Octave has been developed and proposed as an extension of the CIA-traid. The IAS Octave is one of four dimensions of a Reference Model of Information Assurance & Security (RMIAS). The IAS Octave includes confidentiality, integrity, availability, privacy, authenticity & trustworthiness, non-repudiation, accountability and auditability.',[2][15] The IAS Octave as a set of currently relevant security goals has been evaluated via a series of interviews with InfoSec and IA professionals and academics. In,[15] definitions for every member of the IAS Octave are outlined along with the applicability of every security goal (key factor) to six components of an Information System.

**Confidentiality**

Confidentiality refers to preventing the disclosure of information to unauthorized individuals or systems. For example, a credit card transaction on the Internet requires the credit card number to be transmitted from the buyer to the merchant and from the merchant to a transaction processing network. The system attempts to enforce confidentiality by encrypting the card number during transmission, by limiting the places where it might appear (in databases, log files, backups, printed receipts, and so on), and by restricting access to the places where it is stored. If an unauthorized party obtains the card number in any way, a breach of confidentiality has occurred.

Confidentiality is necessary for maintaining the privacy of the people whose personal information is held in the system.

**Integrity**

In information security, data integrity means maintaining and assuring the accuracy and consistency of data over its entire life-cycle.[16] This means that data cannot be modified in an unauthorized or undetected manner. This is not the same thing as referential integrity in databases, although it can be viewed as a special case of Consistency as understood in the classic ACID model of transaction processing. Integrity is violated when a message is actively modified in transit. Information security systems typically provide message integrity in addition to data confidentiality.

**Availability**

For any information system to serve its purpose, the information must be available when it is needed. This means that the computing systems used to store and process the information, the security controls used to protect it, and the communication channels used to access it must be functioning correctly. High availability systems aim to remain available at all times, preventing service disruptions due to power outages, hardware failures, and system upgrades. Ensuring availability also involves preventing denial-of-service attacks, such as a flood of incoming messages to the target system essentially forcing it to shut down.

**Authenticity**

In computing, e-Business, and information security, it is necessary to ensure that the data, transactions, communications or documents (electronic or physical) are genuine. It is also important for authenticity to validate that both parties involved are who they claim to be. Some information security systems incorporate authentication features such as "digital signatures", which give evidence that the message data is genuine and was sent by someone possessing the proper signing key.

**Non-repudiation**

In law, non-repudiation implies one's intention to fulfill their obligations to a contract. It also implies that one party of a transaction cannot deny having received a transaction nor can the other party deny having sent a transaction.

It is important to note that while technology such as cryptographic systems can assist in non-repudiation efforts, the concept is at its core a legal concept transcending the realm of technology. It is not, for instance, sufficient to show that the message matches a digital signature signed with the sender's private key, and thus only the sender could have sent the message and nobody else could have altered it in transit. The alleged sender could in return demonstrate that the digital signature algorithm is vulnerable or flawed, or allege or prove that his signing key has been compromised. The fault for these violations may or may not lie with the sender himself, and such assertions may or may not relieve the sender of liability, but the assertion would invalidate the claim that the signature necessarily proves authenticity and integrity and thus prevents repudiation.

**Information security analysts**

Information security analysts are information technology (IT) specialists who are accountable for safeguarding all data and communications that are stored and shared in network systems. In the financial industry, for example, information security analysts might continually upgrade firewalls that prohibit superfluous access to sensitive business data and might perform defencelessness tests to assess the effectiveness of security measures.

Electronic commerce uses technology such as digital signatures and public key encryption to establish authenticity and non-repudiation.

**Risk management**

The Certified Information Systems Auditor (CISA) Review Manual 2006 provides the following definition of risk management: "Risk management is the process of identifying vulnerabilities and threats to the information resources used by an organization in achieving business objectives, and deciding what countermeasures, if any, to take in reducing risk to an acceptable level, based on the value of the information resource to the organization."[17]

There are two things in this definition that may need some clarification. First, the process of risk management is an ongoing, iterative process. It must be repeated indefinitely. The business environment is constantly changing and new threats and vulnerabilities emerge every day. Second, the choice of countermeasures (controls) used to manage risks must strike a balance between productivity, cost, effectiveness of the countermeasure, and the value of the informational asset being protected.

Risk analysis and risk evaluation processes have their limitations since, when security incidents occur, they emerge in a context, and their rarity and even their uniqueness give rise to unpredictable threats. The analysis of these phenomena which are characterized by breakdowns, surprises and side-effects, requires a theoretical approach which is able to examine and interpret subjectively the detail of each incident.[18]

Risk is the likelihood that something bad will happen that causes harm to an informational asset (or the loss of the asset). A vulnerability is a weakness that could be used to endanger or cause harm to an informational asset. A threat is anything (manmade or act of nature) that has the potential to cause harm.

The likelihood that a threat will use a vulnerability to cause harm creates a risk. When a threat does use a vulnerability to inflict harm, it has an impact. In the context of information security, the impact is a loss of availability, integrity, and confidentiality, and possibly other losses (lost income, loss of life, loss of real property). It should be pointed out that it is not possible to identify all risks, nor is it possible to eliminate all risk. The remaining risk is called "residual risk".

A risk assessment is carried out by a team of people who have knowledge of specific areas of the business. Membership of the team may vary over time as different parts of the business are assessed. The assessment may use a subjective qualitative analysis based on informed opinion, or where reliable dollar figures and historical information is available, the analysis may use quantitative analysis.

The research has shown that the most vulnerable point in most information systems is the human user, operator, designer, or other human[19] The ISO/IEC 27002:2005 Code of practice for information security management recommends the following be examined during a risk assessment:

In broad terms, the risk management process consists of:

Identification of assets and estimating their value. Include: people, buildings, hardware, software, data (electronic, print, other), supplies.

Conduct a threat assessment. Include: Acts of nature, acts of war, accidents, malicious acts originating from inside or outside the organization.

Conduct a vulnerability assessment, and for each vulnerability, calculate the probability that it will be exploited. Evaluate policies, procedures, standards, training, physical security, quality control, technical security.

Calculate the impact that each threat would have on each asset. Use qualitative analysis or quantitative analysis.

Identify, select and implement appropriate controls. Provide a proportional response. Consider productivity, cost effectiveness, and value of the asset.

Evaluate the effectiveness of the control measures. Ensure the controls provide the required cost effective protection without discernible loss of productivity.

For any given risk, management can choose to accept the risk based upon the relative low value of the asset, the relative low frequency of occurrence, and the relative low impact on the business. Or, leadership may choose to mitigate the risk by selecting and implementing appropriate control measures to reduce the risk. In some cases, the risk can be transferred to another business by buying insurance or outsourcing to another business.[20] The reality of some risks may be disputed. In such cases leadership may choose to deny the risk.

**Controls**

Selecting proper controls and implementing those will initially help an organization to bring down risk to acceptable levels. Control selection should follow and should be based on the risk assessment. Controls can vary in nature but fundamentally they are ways of protecting the confidentiality, integrity or availability of information. ISO/IEC 27001:2005 has defined 133 controls in different areas, but this is not exhaustive.You can implement additional controls according to requirement of the organization. ISO 27001:2013( Still it's in drafted version) has cut down the number of controls to 113.

**Administrative**

Administrative controls (also called procedural controls) consist of approved written policies, procedures, standards and guidelines. Administrative controls form the framework for running the business and managing people. They inform people on how the business is to be run and how day to day operations are to be conducted. Laws and regulations created by government bodies are also a type of administrative control because they inform the business. Some industry sectors have policies, procedures, standards and guidelines that must be followed – the Payment Card Industry (PCI) Data Security Standard required by Visa and MasterCard is such an example. Other examples of administrative controls include the corporate security policy, password policy, hiring policies, and disciplinary policies.

Administrative controls form the basis for the selection and implementation of logical and physical controls. Logical and physical controls are manifestations of administrative controls. Administrative controls are of paramount importance.

**Logical**

Logical controls (also called technical controls) use software and data to monitor and control access to information and computing systems. For example: passwords, network and host based firewalls, network intrusion detection systems, access control lists, and data encryption are logical controls.

An important logical control that is frequently overlooked is the principle of least privilege. The principle of least privilege requires that an individual, program or system process is not granted any more access privileges than are necessary to perform the task. A blatant example of the failure to adhere to the principle of least privilege is logging into Windows as user Administrator to read Email and surf the Web. Violations of this principle can also occur when an individual collects additional access privileges over time. This happens when employees' job duties change, or they are promoted to a new position, or they transfer to another department. The access privileges required by their new duties are frequently added onto their already existing access privileges which may no longer be necessary or appropriate.

**Physical**

Physical controls monitor and control the environment of the work place and computing facilities. They also monitor and control access to and from such facilities. For example: doors, locks, heating and air conditioning, smoke and fire alarms, fire suppression systems, cameras, barricades, fencing, security guards, cable locks, etc. Separating the network and workplace into functional areas are also physical controls.

An important physical control that is frequently overlooked is the separation of duties. Separation of duties ensures that an individual can not complete a critical task by himself. For example: an employee who submits a request for reimbursement should not also be able to authorize payment or print the check. An applications programmer should not also be the server administrator or the database administrator – these roles and responsibilities must be separated from one another.[21]

**Defense in depth**

Information security must protect information throughout the life span of the information, from the initial creation of the information on through to the final disposal of the information. The information must be protected while in motion and while at rest. During its lifetime, information may pass through many different information processing systems and through many different parts of information processing systems. There are many different ways the information and information systems can be threatened. To fully protect the information during its lifetime, each component of the information processing system must have its own protection mechanisms. The building up, layering on and overlapping of security measures is called defense in depth. The strength of any system is no greater than its weakest link. Using a defense in depth strategy, should one defensive measure fail there are other defensive measures in place that continue to provide protection.

Recall the earlier discussion about administrative controls, logical controls, and physical controls. The three types of controls can be used to form the basis upon which to build a defense-in-depth strategy. With this approach, defense-in-depth can be conceptualized as three distinct layers or planes laid one on top of the other. Additional insight into defense-in- depth can be gained by thinking of it as forming the layers of an onion, with data at the core of the onion, people the next outer layer of the onion, and network security, host-based security and application security forming the outermost layers of the onion. Both perspectives are equally valid and each provides valuable insight into the implementation of a good defense-in-depth strategy.

**Security classification for information**

An important aspect of information security and risk management is recognizing the value of information and defining appropriate procedures and protection requirements for the information. Not all information is equal and so not all information requires the same degree of protection. This requires information to be assigned a security classification.

The first step in information classification is to identify a member of senior management as the owner of the particular information to be classified. Next, develop a classification policy. The policy should describe the different classification labels, define the criteria for information to be assigned a particular label, and list the required security controls for each classification.

Some factors that influence which classification information should be assigned include how much value that information has to the organization, how old the information is and whether or not the information has become obsolete. Laws and other regulatory requirements are also important considerations when classifying information.

The Business Model for Information Security enables security professionals to examine security from systems perspective, creating an environment where security can be managed holistically, allowing actual risks to be addressed.

The type of information security classification labels selected and used will depend on the nature of the organization, with examples being:

In the business sector, labels such as: Public, Sensitive, Private, Confidential.

In the government sector, labels such as: Unclassified, Sensitive But Unclassified, Restricted, Confidential, Secret, Top Secret and their non-English equivalents.

In cross-sectoral formations, the Traffic Light Protocol, which consists of: White, Green, Amber, and Red.

All employees in the organization, as well as business partners, must be trained on the classification schema and understand the required security controls and handling procedures for each classification. The classification of a particular information asset that has been assigned should be reviewed periodically to ensure the classification is still appropriate for the information and to ensure the security controls required by the classification are in place.

**Access control**

Access to protected information must be restricted to people who are authorized to access the information. The computer programs, and in many cases the computers that process the information, must also be authorized. This requires that mechanisms be in place to control the access to protected information. The sophistication of the access control mechanisms should be in parity with the value of the information being protected – the more sensitive or valuable the information the stronger the control mechanisms need to be. The foundation on which access control mechanisms are built start with identification and authentication.

Identification is an assertion of who someone is or what something is. If a person makes the statement "Hello, my name is John Doe" they are making a claim of who they are. However, their claim may or may not be true. Before John Doe can be granted access to protected information it will be necessary to verify that the person claiming to be John Doe really is John Doe.

Authentication is the act of verifying a claim of identity. When John Doe goes into a bank to make a withdrawal, he tells the bank teller he is John Doe—a claim of identity. The bank teller asks to see a photo ID, so he hands the teller his driver's license. The bank teller checks the license to make sure it has John Doe printed on it and compares the photograph on the license against the person claiming to be John Doe. If the photo and name match the person, then the teller has authenticated that John Doe is who he claimed to be.

There are three different types of information that can be used for authentication:

Something you know: things such as a PIN, a password, or your mother's maiden name.

Something you have: a driver's license or a magnetic swipe card.

Something you are: biometrics, including palm prints, fingerprints, voice prints and retina (eye) scans.

Strong authentication requires providing more than one type of authentication information (two-factor authentication). The username is the most common form of identification on computer systems today and the password is the most common form of authentication. Usernames and passwords have served their purpose but in our modern world they are no longer adequate.[citation needed] Usernames and passwords are slowly being replaced with more sophisticated authentication mechanisms.

After a person, program or computer has successfully been identified and authenticated then it must be determined what informational resources they are permitted to access and what actions they will be allowed to perform (run, view, create, delete, or change). This is called authorization. Authorization to access information and other computing services begins with administrative policies and procedures. The policies prescribe what information and computing services can be accessed, by whom, and under what conditions. The access control mechanisms are then configured to enforce these policies. Different computing systems are equipped with different kinds of access control mechanisms—some may even offer a choice of different access control mechanisms. The access control mechanism a system offers will be based upon one of three approaches to access control or it may be derived from a combination of the three approaches.

The non-discretionary approach consolidates all access control under a centralized administration. The access to information and other resources is usually based on the individuals function (role) in the organization or the tasks the individual must perform. The discretionary approach gives the creator or owner of the information resource the ability to control access to those resources. In the Mandatory access control approach, access is granted or denied basing upon the security classification assigned to the information resource.

Examples of common access control mechanisms in use today include role-based access control available in many advanced database management systems—simple file permissions provided in the UNIX and Windows operating systems, Group Policy Objects provided in Windows network systems, Kerberos, RADIUS, TACACS, and the simple access lists used in many firewalls and routers.

To be effective, policies and other security controls must be enforceable and upheld. Effective policies ensure that people are held accountable for their actions. All failed and successful authentication attempts must be logged, and all access to information must leave some type of audit trail.[citation needed]

Also, need-to-know principle needs to be in affect when talking about access control. Need-to-know principle gives access rights to a person to perform their job functions. This principle is used in the government, when dealing with difference clearances. Even though two employees in different departments have a top-secret clearance, they must have a need-to-know in order for information to be exchanged. Within the need-to-know principle, network administrators grant the employee least amount privileges to prevent employees access and doing more than what they are supposed to. Need-to-know helps to enforce the confidential-integrity-availability (C‑I‑A) triad. Need-to-know directly impacts the confidential area of the triad.

**Cryptography**

Information security uses cryptography to transform usable information into a form that renders it unusable by anyone other than an authorized user; this process is called encryption. Information that has been encrypted (rendered unusable) can be transformed back into its original usable form by an authorized user, who possesses the cryptographic key, through the process of decryption. Cryptography is used in information security to protect information from unauthorized or accidental disclosure while the information is in transit (either electronically or physically) and while information is in storage.

Cryptography provides information security with other useful applications as well including improved authentication methods, message digests, digital signatures, non-repudiation, and encrypted network communications. Older less secure applications such as telnet and ftp are slowly being replaced with more secure applications such as ssh that use encrypted network communications. Wireless communications can be encrypted using protocols such as WPA/WPA2 or the older (and less secure) WEP. Wired communications (such as ITU‑T G.hn) are secured using AES for encryption and X.1035 for authentication and key exchange. Software applications such as GnuPG or PGP can be used to encrypt data files and Email.

Cryptography can introduce security problems when it is not implemented correctly. Cryptographic solutions need to be implemented using industry accepted solutions that have undergone rigorous peer review by independent experts in cryptography. The length and strength of the encryption key is also an important consideration. A key that is weak or too short will produce weak encryption. The keys used for encryption and decryption must be protected with the same degree of rigor as any other confidential information. They must be protected from unauthorized disclosure and destruction and they must be available when needed. Public key infrastructure (PKI) solutions address many of the problems that surround key management.

**Process**

The terms reasonable and prudent person, due care and due diligence have been used in the fields of Finance, Securities, and Law for many years. In recent years these terms have found their way into the fields of computing and information security. U.S.A. Federal Sentencing Guidelines now make it possible to hold corporate officers liable for failing to exercise due care and due diligence in the management of their information systems.

In the business world, stockholders, customers, business partners and governments have the expectation that corporate officers will run the business in accordance with accepted business practices and in compliance with laws and other regulatory requirements. This is often described as the "reasonable and prudent person" rule. A prudent person takes due care to ensure that everything necessary is done to operate the business by sound business principles and in a legal ethical manner. A prudent person is also diligent (mindful, attentive, and ongoing) in their due care of the business.

In the field of Information Security, Harris[22] offers the following definitions of due care and due diligence:

"Due care are steps that are taken to show that a company has taken responsibility for the activities that take place within the corporation and has taken the necessary steps to help protect the company, its resources, and employees." And, [Due diligence are the] "continual activities that make sure the protection mechanisms are continually maintained and operational."

Attention should be made to two important points in these definitions. First, in due care, steps are taken to show - this means that the steps can be verified, measured, or even produce tangible artifacts. Second, in due diligence, there are continual activities - this means that people are actually doing things to monitor and maintain the protection mechanisms, and these activities are ongoing.

=========================================================================

**Tor (The Onion Router)** is free software for enabling online anonymity. Tor directs Internet traffic through a free, worldwide, volunteer network consisting of more than four thousand relays[6] to conceal a user's location or usage from anyone conducting network surveillance or traffic analysis. Using Tor makes it more difficult to trace Internet activity, including "visits to Web sites, online posts, instant messages, and other communication forms", back to the user[7] and is intended to protect the personal privacy of users, as well as their freedom and ability to conduct confidential business by keeping their internet activities from being monitored.

"Onion Routing" refers to the layers of the encryption used. The original data, including its destination, are encrypted and re-encrypted multiple times, and are sent through a virtual circuit comprising successive, randomly selected Tor relays. Each relay decrypts a "layer" of encryption to reveal only the next relay in the circuit, in order to pass the remaining encrypted data on to it. The final relay decrypts the last layer of encryption and sends the original data, without revealing or even knowing its sender, to the destination. This method reduces the chance of the original data being understood in transit and, more notably, conceals the routing of it.[8]

As the 2013 anonymity-stripping attacks EgotisticalGiraffe[9] on Freedom Hosting users demonstrated, it is possible to attack Tor users indirectly, e.g., via vulnerabilities in servers and web browsers.[10] However, an NSA appraisal released by The Guardian in October of that year characterized Tor as "[s]till the King of high secure, low latency Internet anonymity" and that "[t]here are no contenders for the throne in waiting".

**History**

An alpha version of the free software, with the onion routing network "functional and deployed", was announced on 20 September 2002.[2] Roger Dingledine, Nick Mathewson, and Paul Syverson presented "Tor: The Second-Generation Onion Router" at the thirteenth USENIX Security Symposium on 13 August 2004.[12] Although the name Tor originated as an acronym of The Onion Routing project (TOR project), the current project no longer considers the name to be an acronym, and therefore, does not use all capital letters.[13]

Originally sponsored by the U.S. Naval Research Laboratory,[12] which had been instrumental in the early development of onion routing under the aegis of DARPA, Tor was financially supported by the Electronic Frontier Foundation from 2004 to 2005.[14] Tor software is now developed by the Tor Project, which has been a 501(c)(3) research-education nonprofit organization [15] based in the United States of America [1] since December 2006. It has a diverse base of financial support;[14] the U.S. State Department, the Broadcasting Board of Governors, and the National Science Foundation are major contributors.[16] As of 2012, 80% of the Tor Project's $2M annual budget comes from the United States government, with the Swedish government and other organizations providing the rest,[17] including NGOs and thousands of individual sponsors.[18]

In March 2011, the Tor Project was awarded the Free Software Foundation's 2010 Award for Projects of Social Benefit on the following grounds: "Using free software, Tor has enabled roughly 36 million people around the world to experience freedom of access and expression on the Internet while keeping them in control of their privacy and anonymity. Its network has proved pivotal in dissident movements in both Iran and more recently Egypt."[19]

Foreign Policy named Dingledine, Mathewson, and Syverson among its 2012 Top 100 Global Thinkers "for making the web safe for whistleblowers."[20]

In 2013, Jacob Appelbaum described Tor as a "part of an ecosystem of software that helps people regain and reclaim their autonomy. It helps to enable people to have agency of all kinds; it helps others to help each other and it helps you to help yourself. It runs, it is open and it is supported by a large community spread across all walks of life.".[21]

Edward Snowden used the Tor Network to send information about PRISM to the Washington Post and The Guardian in June 2013.[22]

**Operation**

Tor aims to conceal its users' identities and their network activity from surveillance and traffic analysis by separating identification and routing. It is an implementation of onion routing, which encrypts and then randomly bounces communications through a network of relays run by volunteers around the globe. These onion routers employ encryption in a multi-layered manner (hence the onion metaphor) to ensure perfect forward secrecy between relays, thereby providing users with anonymity in network location. That anonymity extends to the hosting of censorship-resistant content via Tor's anonymous hidden service feature.[12] Furthermore, by keeping some of the entry relays (bridge relays) secret, users can evade Internet censorship that relies upon blocking public Tor relays.[23]

Because the internet address of the sender and the recipient are not both in cleartext at any hop along the way, anyone eavesdropping at any point along the communication channel cannot directly identify both ends. Furthermore, to the recipient it appears that the last Tor node (the exit node) is the originator of the communication rather than the sender.

**Originating traffic**

Users of a Tor network run an onion proxy on their machine. The Tor software periodically negotiates a virtual circuit through the Tor network, using multi-layer encryption, ensuring perfect forward secrecy. At the same time, the onion proxy software presents a SOCKS interface to its clients. SOCKS-aware applications may be pointed at Tor, which then multiplexes the traffic through a Tor virtual circuit.

Once inside a Tor network, the traffic is sent from router to router, ultimately reaching an exit node at which point the cleartext packet is available and is forwarded on to its original destination. Viewed from the destination, the traffic appears to originate at the Tor exit node.

Tor's application independence sets it apart from most other anonymity networks: it works at the Transmission Control Protocol (TCP) stream level. Applications whose traffic is commonly anonymised using Tor include Internet Relay Chat (IRC), instant messaging, and World Wide Web browsing. When browsing the Web, Tor often is coupled with Polipo or Privoxy proxy servers. Privoxy is a filtering proxy server that aims to add privacy at the application layer. The Polipo proxy server can speak the SOCKS 4 & SOCKS 5 protocols and does HTTP 1.1 pipelining well, so it can enhance Tor's communication latency. TorProject.org therefore recommends that Polipo be used together with the Tor anonymising network.[24]

On older versions of Tor (resolved May–July 2010),[25] as with many anonymous web surfing systems, direct Domain Name System (DNS) requests are usually still performed by many applications without using a Tor proxy. This allows someone monitoring a user's connection to determine (for example) which WWW sites they are viewing using Tor, even though they cannot see the content being viewed. Using Privoxy or the command "torify" included with a Tor distribution is a possible solution to this problem.[26]

Additionally, applications using SOCKS5 – which supports name-based proxy requests – can route DNS requests through Tor, having lookups performed at the exit node and thus, receiving the same anonymity as other Tor traffic.[27]

As of Tor release 0.2.0.1-alpha, Tor includes its own DNS resolver, which will dispatch queries over the mix network. This should close the DNS leak and can interact with Tor's address mapping facilities to provide the Tor hidden service (.onion) access to non-SOCKS-aware applications.[25]

**Hidden services**

Tor can also provide anonymity to websites and other servers. Servers configured to receive inbound connections only through Tor are called hidden services. Rather than revealing a server's IP address (and thus its network location), a hidden service is accessed through its onion address. The Tor network understands these addresses and can route data to and from hidden services, even to those hosted behind firewalls or network address translators (NAT), while preserving the anonymity of both parties. Tor is necessary to access hidden services.[28]

Hidden services have been deployed on the Tor network since 2004.[29] Other than the database that stores the hidden-service descriptors,[30] Tor is decentralized by design; there is no direct readable list of all hidden services, although a number of hidden services catalog publicly known onion addresses.

Because hidden services do not use exit nodes, connection to a hidden service is encrypted end-to-end and not subject to eavesdropping. There are, however, security issues involving Tor hidden services. For example, services that are reachable through Tor hidden services and the public Internet, are susceptible to correlation attacks and thus not perfectly hidden. Other pitfalls include misconfigured services (e.g. identifying information included by default in web server error responses),[28] uptime and downtime statistics, intersection attacks, and user error.

**Weaknesses**

Like all current low latency anonymity networks, Tor cannot and does not attempt to protect against monitoring of traffic at the boundaries of the Tor network, i.e., the traffic entering and exiting the network. While Tor does provide protection against traffic analysis, it cannot prevent traffic confirmation (also called end-to-end correlation).[31][32]

In spite of known weaknesses and attacks listed here, Tor and the alternative network system JonDonym (Java Anon Proxy, JAP) are considered more resilient than alternatives such as VPNs. Were a local observer on an ISP or WLAN to attempt to analyze the size and timing of the encrypted data stream going through the VPN, Tor, or JonDo system, the latter two would be harder to analyze, as demonstrated by a 2009 study.[33]

Researchers from the University of Michigan developed a network scanner allowing identification of 86 percent of live Tor “bridges” with a single scan.[34]

**Bad Apple attack**

Steven J. Murdoch and George Danezis from University of Cambridge presented an article at the 2005 IEEE Symposium on security and privacy on traffic-analysis techniques that allow adversaries with only a partial view of the network to infer which nodes are being used to relay the anonymous streams.[35] These techniques greatly reduce the anonymity provided by Tor. Murdoch and Danezis have also shown that otherwise unrelated streams can be linked back to the same initiator. This attack, however, fails to reveal the identity of the original user.[35] Murdoch has been working with—and has been funded by—Tor since 2006.

There is an attack on Tor where, if an Autonomous System (AS) exists on both path from Alice to entry relay and from exit relay to Bob, that AS is able to de-anonymize the path. In 2012, LASTor [36] proposed a method to avoid this attack. They also propose a path selection algorithm to reduce latency of communications in Tor.

In March 2011, researchers with the Rocquencourt, France based National Institute for Research in Computer Science and Control (Institut national de recherche en informatique et en automatique, INRIA) documented an attack that is capable of revealing the IP addresses of BitTorrent users on the Tor network. The "bad apple attack" exploits Tor's design and takes advantage of insecure application use to associate the simultaneous use of a secure application with the IP address of the Tor user in question. One method of attack depends on control of an exit node or hijacking tracker responses, while a secondary attack method is based in part on the statistical exploitation of distributed hash table tracking.[37] According to the study:

This attack against Tor consists of two parts: (a) exploiting an insecure application to reveal the source IP address of, or trace, a Tor user and (b) exploiting Tor to associate the use of a secure application with the IP address of a user (revealed by the insecure application). As it is not a goal of Tor to protect against application-level attacks, Tor cannot be held responsible for the first part of this attack. However, because Tor's design makes it possible to associate streams originating from secure application with traced users, the second part of this attack is indeed an attack against Tor. We call the second part of this attack the bad apple attack. (The name of this attack refers to the saying 'one bad apple spoils the bunch.' We use this wording to illustrate that one insecure application on Tor may allow to trace other applications.)[37]

The results presented in the bad apple attack research paper are based on an attack in the wild launched against the Tor network by the authors of the study. The attack targeted six exit nodes, lasted for 23 days, and revealed a total of 10,000 IP addresses of active Tor users. This study is particularly significant because it is the first documented attack designed to target P2P file sharing applications on Tor.[37] BitTorrent may generate as much as 40% of all traffic on Tor.[38] Furthermore, the bad apple attack is effective against insecure use of any application over Tor, not just BitTorrent.[37]

**Exit nodes should not be trusted**

In September 2007, Dan Egerstad, a Swedish security consultant, revealed that he had intercepted usernames and passwords for a large number of e-mail accounts by operating and monitoring Tor exit nodes.[39] As Tor does not, and by design cannot, encrypt the traffic between an exit node and the target server, any exit node is in a position to capture any traffic passing through it that does not use end-to-end encryption such as TLS. While this may not inherently breach the anonymity of the source, traffic intercepted in this way by self-selected third parties can expose information about the source in either or both of payload and protocol data.[40] Furthermore, Egerstad is circumspect about the possible subversion of Tor by intelligence agencies –

"If you actually look in to where these Tor nodes are hosted and how big they are, some of these nodes cost thousands of dollars each month just to host because they're using lots of bandwidth, they're heavy-duty servers and so on. Who would pay for this and be anonymous?" [41]

In October 2011, a research team from ESIEA (a French engineering school) claimed to have discovered a way to compromise the Tor network by decrypting communication passing over it.[42][43] The technique they describe requires creating a map of Tor network nodes, controlling one third of them, and then acquiring their encryption keys and algorithm seeds. Then, using these known keys and seeds, they claim the ability to decrypt two encryption layers out of three. They claim to break the third key by a statistical-based attack. In order to redirect Tor traffic to the nodes they controlled, they used a denial-of-service attack. A response to this claim has been published on the official Tor Blog stating that these rumours of Tor's compromise are greatly exaggerated.

**IT ASSET MANAGEMENT**

IT Asset Management (ITAM) considers assets from “cradle to grave,” e.g., from planning through acquisition, discovery, optimization, management, and ultimately, disposal. From a best practices perspective, ITAM considers all the physical, financial, and contractual aspects of IT assets throughout their life cycles. The goals of a comprehensive ITAM program should be:

•To cost-effectively acquire the most appropriate IT assets for the organization with

Minimal risk and maximum benefit

•To maximize IT asset utilization

•To retire or re-purpose IT assets when they no longer provide benefit commensurate

With the cost to maintain them

•To provide decision support mechanisms in support of specific internal and external

Requirements, e.g., mergers and acquisitions , audits, regulatory compliance, etc.

Additionally, effective ITAM requires a balance of people, processes and tools. A team approach,with participation by Purchasing/Procurement, Contracts/Legal, IT, Business Units, Finance, and Asset Management, ensures this balance.

**Potential ITAM Benefits**

**HARD BENEFITS**

Are those that you can quantify in dollars saved. Most common examples are:

•Reduced IT spending because unused hardware and software products have been

reassigned or eliminated

•Just-in-time licensing and maintenance payments

•Improved cash flow

•Paying only what you owe because you can reconcile invoices with asset management

inventory, payment and contract data.

•Replacing multiple spreadsheets and databases with a centralized, accurate and up-to-date asset repository

**SOFT BENEFITS**

Are those that are measured in terms of time and labor costs saved improvements to processes, and the ability to perform activities that might otherwise be difficult or impossible. Soft benefits can significantly outstrip “hard” benefits. However, they are more difficult to quantify. Most common examples are:

•Reduced labor costs associated with time saved by accurate, consolidated asset reports

•Reduction in numbers of Service Desk personnel

•Better budgeting, buying and sourcing decisions, e.g. less time needed to consolidate data for decision-making

•Increased credibility with senior management based on comprehensive, cross-functional IT asset information

•Ability to capture and redeploy assets from personnel who have left the company

•Ability to implement IT standard configurations, thus leveraging suppliers and reducing

internal and external support costs

•Automated procurement of assets

•Ability to justify IT service costs based on hard data

•Improved service management

**COST AVOIDANCE**

This represents the opportunity costs, money that would have been spent if the ITAM program had not been in place. Most common examples are money saved when you:

•Harvest and reuse current IT assets instead of buying new ones

•Negotiate better volume agreements with your hardware and software suppliers

•Transfer software licenses (where contract rights allow) to new employees when others leave

•Stop leasing payments as soon as leases expire because you have accurate contract expiry information

•Do not overbuy software licenses or capacity just to be compliant in case of an audit

•Avoid supplier-driven software audit penalties

•Improve vendor management

**RISK MITIGATION**

This represents reductions in organizational liability that could cost the organization considerably and hurt its reputation if the ITAM program had not been in place.Most common examples are:

•Software license compliance

•Reduction in liability due to business change

•Reduction in contractual liability

•Managed electronic software distribution

•Reduction in liability due to employees’ personal software being loaded on company systems

•Proper asset disposal in compliance with regulations

•Increased security

•Reduction in liability due to information management issues resolved with proper

disposal processes

•Compliance with governmental regulations, such as Sarbanes-Oxley (See Section 3)

Stages of ITAM

=========================================================================

**Question Answers**

1. \_\_\_\_\_\_\_\_\_ is an area of a computer that holds data that is waiting to be processed. Memory
2. A computer processes data in a device called the \_\_\_\_\_\_\_\_\_\_\_. CPU
3. \_\_\_\_\_\_\_\_(hard drives, disks, tapes, etc.) is the area where data can be left on a permanent basis while it is not needed for processing. Storage
4. The results produced by a computer are known as computer \_\_\_\_\_\_\_\_\_\_\_\_. Output
5. The term "hardware" refers to the computer itself and to components called \_\_\_\_\_\_\_\_\_\_\_ that expand the computer's input, output, and storage capabilities. peripheral devices
6. \_\_\_\_\_\_\_\_\_\_ allows one or more words in a document to act as a link to another document. Hypertext
7. Computer hardware in and of itself does not provide a particularly useful mind tool. To be useful, a computer requires a computer program or \_\_\_\_\_\_\_\_\_\_, which is a set of instructions that tells a computer how to perform a particular task. Software
8. Traditionally, computers have been classified into four categories, from least to most powerful, are microcomputers, minicomputers, mainframe computers, and \_\_\_\_\_\_\_\_\_\_\_. Supercomputers
9. Microcomputers, also known as \_\_\_\_\_\_\_\_\_ computers, are typically found in homes and small businesses. Prices range from $500 to $5,000 but consumers typically purchase systems in the middle of this range, spending from $1,000 to $2,000. Personal
10. A \_\_\_\_\_\_\_\_\_\_ is somewhat more powerful than a microcomputer and can carry out the processing tasks for several people working at terminals that are connected to the minicomputer. Minicomputer
11. A \_\_\_\_\_\_\_\_ is an input and output device that resembles a microcomputer because it has a keyboard and screen. Terminal
12. \_\_\_\_\_\_\_\_\_\_\_ are large, fast, and fairly expensive computers, generally used by business or government to provide centralized storage, processing and management for large amounts of data. Mainframes
13. \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ are physical materials that provide long-term storage for computer data. Storage media
14. A \_\_\_\_\_\_\_\_ \_\_\_\_\_\_ can store billions of characters on a non-removable disk platter. Hard drive
15. A CD-ROM drive and a \_\_\_\_\_ drive are storage devices that use laser technology to read data from optical disks. DVD
16. A \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_\_ is a storage device that writes data on floppy disks. floppy disc drive
17. Most of the computers used in people's homes are based on one of two major platforms--PCs and \_\_\_\_\_ Macs
18. Windows normally runs on the \_\_\_\_ platform. windows
19. Computers that operate in essentially the same way are said to be \_\_\_\_\_\_\_\_. compatible
20. A computer \_\_\_\_\_\_\_\_ is a collection of computers and other devices that have been connected in order to share data, hardware, and software. Network
21. The world's largest network, the \_\_\_\_\_\_\_\_\_\_, provides many information services, but the most popular is the World Wide Web, often referred to simply as the Web. internet
22. If you type the formula a1+b1/2 into a spreadsheet cell. What is the first mathematical operation that occurs? \_\_\_\_\_\_\_\_\_\_ Division
23. In a math equation, the computer will calculate whatever is in the parentheses first. It then processes \_\_\_\_\_\_\_\_\_\_\_\_\_ next. exponents
24. In a spreadsheet, referring to a cell with an address such as $B$5 is called using a \_\_\_\_\_\_\_\_\_\_ address. absolute
25. In a spreadsheet, how would you reference the range of cells in column B including rows 3 through 11, using relative addressing? \_\_\_\_\_\_\_ B3:B11
26. Office \_\_\_\_\_\_\_\_\_\_\_ systems include E-mail, word processing, voice mail, scheduling, databases, and more. automation
27. The means by which humans and computers communicate is referred to as the \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_. user interface
28. A \_\_\_\_\_\_\_\_ is a message displayed by the computer that asks for input from the user. PROMPT
29. A \_\_\_\_\_\_\_\_\_\_ is an instruction you input to tell the computer to carry out a task. command
30. \_\_\_\_\_\_\_\_\_ specifies the sequence and punctuation for command words and parameters. syntax
31. COBOL is a \_\_\_\_\_\_\_\_\_\_ language. compiled
32. If you misspell a command word, leave out required punctuation, or type the command words out of order, you have made a \_\_\_\_\_\_\_\_\_\_ error and the computer will display an error message. syntax
33. An interface that requires the user to type commands is referred to as a \_\_\_\_\_\_\_\_\_\_\_-\_\_\_\_\_. command line
34. Round option buttons sometimes called "\_\_\_\_\_\_\_\_ buttons," allow you to select only one of the options. radio
35. Square \_\_\_\_\_\_\_\_\_\_\_\_ allow you to select any or all of the options. checkboxes
36. The more dots your screen displays in the matrix, the higher its \_\_\_\_\_\_\_\_\_. resolution
37. Software for accessing the World Wide Web is called a \_\_\_\_\_\_\_\_\_\_\_. browser
38. You can search for information on a specific topic using a \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_. search engine
39. \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ refers to the ability of computers to solve problems and perform tasks that were once thought to be uniquely human. AI (Artificial Intelligence)
40. An employee at IBM(ibm.com) would probably have the domain name \_\_\_\_\_\_\_\_ after the "at" symbol in his work e-mail address ibm.com
41. \_\_\_\_\_\_\_\_\_\_\_\_ is normally thought of as a set of instructions for the computer and its associated data, which are stored in electronic format, that direct the computer to accomplish certain tasks Software
42. People who illegally copy, distribute, or modify software are often called \_\_\_\_\_\_\_\_\_\_. Pirates
43. Illegal copies of software are referred to as \_\_\_\_\_\_\_\_\_\_ software. Pirated
44. A \_\_\_\_\_\_\_\_\_\_ is a legal contract which defines the ways in which you may use the computer program. License
45. A \_\_\_\_\_ license allows the software to be used on all computers in one location. site
46. \_\_\_\_\_\_\_\_ is "try before you buy" software. Shareware
47. \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ software, or "freeware", is software that has been donated to the public, so it is not owned by the author. Public domain
48. Mail \_\_\_\_\_\_\_\_ is a feature supported by many word processors that enables you to generate form letters. Merge
49. There are two basic categories of software. \_\_\_\_\_\_\_\_\_\_\_\_ software is a program designed for use by end-users. Applications
50. \_\_\_\_\_\_\_\_\_\_\_\_\_\_ software helps the user carry out a specific task. Application
51. A \_\_\_\_\_\_\_\_\_ bit is an extra bit of information added to every piece of data that is transmitted to make sure it is transmitted accurately. Parity
52. An \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ is essentially the controller of all activities that take place on your computer. Operating Systems
53. In addition to providing external services, an operating system will also provide \_\_\_\_\_\_\_\_\_\_\_\_\_ services, which are "behind the scenes" and ensure that the computer is functioning properly. (managing hard drive,find errors in hardware,finding memory) Internal
54. Any part of a computer system that might be used by a computer program is called a \_\_\_\_\_\_\_\_\_\_\_. Resource
55. \_\_\_\_\_\_\_\_\_\_\_ is the most popular operating system for personal computers today. Windows
56. \_\_\_\_\_\_\_\_\_\_\_\_\_ is a service which allows you to work on several projects at a time. Multitasking
57. \_\_\_\_\_\_\_\_ is an operating system that was developed in 1969 by AT&T's Bell Laboratories. UNIX
58. Before you can store data on a disk, the disk must be \_\_\_\_\_\_\_\_\_. Formatted
59. System software which helps the computer control a peripheral device, such as a printer or sound card, is called a device \_\_\_\_\_\_\_\_\_. Driver
60. A \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ allows a programmer to create a program using english-like instructions. programming language
61. \_\_\_\_\_\_\_\_\_\_\_ provides a way for people to collaborate their efforts on a project. Groupware
62. \_\_\_\_\_\_\_\_\_\_\_\_ software makes calculations based on numbers and formulas the user enters. spreadsheets
63. A \_\_\_\_\_\_\_ text file stores information in a text file similar to how you would on index cards. flat
64. You can use \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ software to control another computer remotely. Remote control
65. \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ is a way to use your computer to transfer funds among accounts, download transactions directly from your bank, and pay bills, all via the Web. online banking
66. Many operating systems are downwardly \_\_\_\_\_\_\_\_\_\_\_\_ which means that they can run programs meant for earlier versions. compatiable
67. New software you purchase will typically come with a \_\_\_\_\_\_ program that leads you through the installation process. setup
68. \_\_\_\_\_ stores its contents only as long as the computer is on. RAM
69. \_\_\_\_\_\_ can be written to and erased by the computer. RAM
70. To increase the speed of data access, a computer might use a \_\_\_\_\_\_\_\_, which is a special area in computer memory that holds data that you are most likely going to use soon. Cache
71. The smallest unit of memory is 1 bit or byte
72. There are \_\_\_\_\_\_\_ different characters that can be stored in 1 byte of memory 256
73. A \_\_\_\_\_\_\_ is a named collection of data that exists on a storage medium such as a floppy disk, hard disk, or a CD. file
74. A unique set of letters and numbers that identifies a file is called a \_\_\_\_\_\_\_\_\_. filename
75. A filename might be followed by a \_\_\_\_\_\_\_\_\_ which normally describes the type of file. extension
76. A group of sectors is called a \_\_\_\_\_\_\_\_. cluster
77. An example of a \_\_\_\_\_\_\_\_\_ in Windows is the asterisk, which can be used to select files that have filenames that include the letter combinations that you type. wildcard
78. An \_\_\_\_\_\_\_\_\_\_ tells the computer how to perform a specific task. executable
79. A \_\_\_\_\_\_ \_\_\_\_\_\_\_ contains pictures, words, and numbers that you can view, save, print, edit, and send using executables made specifically for that task. Data file
80. \_\_\_\_\_\_\_\_ is programmed once at the factory, and cannot be overwritten. It contains basic information for the system. ROM
81. A \_\_\_\_\_\_\_\_\_\_\_\_\_ utility can rearrange the files on a disk so that they are stored in contiguous, or back to back sectors of the disk. defragmentation
82. Floppy disks, hard drives, cdrom drives, and dvd drives are random access devices, while tape drives are \_\_\_\_\_\_\_\_\_. Sequential
83. In Windows, a \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ provides a way to refer to a particular storage device. Device letter
84. An operating system maintains a list of files called a \_\_\_\_\_\_\_\_\_\_ or folder for each CD-ROM, DVD or disk. Directory
85. The main directory of a drive is sometimes referred to as the \_\_\_\_ directory. Root
86. A file specification, more commonly known as the \_\_\_\_\_\_\_, consists of the drive, folder, filename, and extension that identifies a file. path
87. A \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ is a program which helps you find, rename, move, copy, and delete files or folders. file manager
88. A \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ is the substance that contains data, which can be a disk, tape, CD, paper, or DVD. storage medium
89. A backup made on magnetic tape is called a \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_. type backup
90. Each 1 or 0 that represents data is called a \_\_\_\_. bit
91. Printers and scanners are examples of \_\_\_\_\_\_\_\_\_\_\_ devices. Peripheral
92. Eight bits make a \_\_\_\_\_. byte
93. The minimum amount of memory that is required to store one character, or one letter, is 1 \_\_\_\_\_\_\_\_. byte
94. The storage technology used for tapes, floppy disks, and hard disks is \_\_\_\_\_\_\_\_ storage. magnetic
95. When files are stored in many noncontiguous (non back-to-back) clusters, they are said to be \_\_\_\_\_\_\_\_\_. fragmented
96. CD and DVD storage technologies are classified as \_\_\_\_\_\_\_\_\_\_\_\_ storage. optical
97. The \_\_\_\_\_\_\_\_ utility can restore deleted files because they will not truly be deleted until you write something over them. undelete
98. When you store a file on a disk, the operating system records the cluster number that contains the beginning of the file in a table called a \_\_\_\_\_\_\_\_\_\_\_. allocation
99. A \_\_\_\_\_\_\_ \_\_\_\_\_\_ contains minimal operating system files and is often used for troubleshooting or installing a new operating system. book disc
100. In Windows, the \_\_\_\_\_\_\_\_\_\_ contains the settings that the computer needs to correctly use its software and hardware devices Registry
101. Digital computers use the \_\_\_\_\_\_\_\_\_ number system, also called "base 2." Binary
102. \_\_\_\_\_\_\_ is the data representation code used on most mainframes and microcomputers Ascii
103. An area in the computer system unit that temporarily holds data before and after it is processed is called \_\_\_\_\_. RAM
104. A \_\_\_\_\_\_\_\_\_\_ translates a program written in a high-level language into object code, or low-level instructions that the operating system can understand. compiler
105. An \_\_\_\_\_\_\_\_\_\_\_\_\_\_ is a set of steps for carrying out a task or solving a problem. The exact format of the algorithm depends on the programming language that will be used to write the program. Algorithm
106. In a program, values are stored in structures called \_\_\_\_\_\_\_\_\_\_\_\_. Variables
107. Another way of expressing an algorithm, which looks more like a programming language than structured English, is known as \_\_\_\_\_\_\_\_\_\_\_\_\_\_. Pseudocode
108. The computer normally reads a program from top to bottom. A program \_\_\_\_\_\_\_\_\_ statement modifies the order in which the computer executes the statements. controls
109. Statements such as FOR, WHILE, and DO WHILE are \_\_\_\_\_\_\_\_\_\_\_\_ control structures. Repetition
110. The \_\_\_\_\_\_ does a basic check for problems in the computer during bootup. POST (Power On Self Test)
111. Disk storage which is used to simulate Random Access Memory (RAM) is called \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_. Virtual memory
112. \_\_\_\_\_\_\_ cannot be overwritten and contains instructions that help a computer prepare processing tasks. ROM
113. The \_\_\_\_\_ memory holds data such as your computer system configuration but requires a small amount of electricity to retain it's data. This power is provided by a small battery on the motherboard CMOS
114. All computers need a \_\_\_\_\_\_\_\_ which takes place from the time you turn on the computer and the time it is ready for you to enter commands. Boot process
115. In a personal computer, the CPU (Central Processing Unit) is a single integrated circuit called a \_\_\_\_\_\_\_\_\_. Microprocessor
116. A Decision \_\_\_\_\_\_\_\_\_ System allows users to create data models of "what-if" scenarios, and provides the tools the decision maker needs to examine the data. Support
117. A Java program created for the internet is called an \_\_\_\_\_\_\_\_\_\_. Applet
118. The time to complete an instruction cycle is measured in millions of cycles, or \_\_\_\_\_\_\_. Mhz
119. \_\_\_\_\_\_\_ is a special high-speed memory that gives the CPU access to data very quickly. Cache
120. A \_\_\_\_\_ is a computer which is based on a central processing unit with a complex instruction set. CISC
121. \_\_\_\_\_\_ machines use a microprocessor with a streamlined set of instructions CISC
122. When a computer has more than one processor, it can perform \_\_\_\_\_\_\_\_\_\_ processing, which uses more than one processor at a time to increase the amount of processing that a computer can accomplish in a certain amount of time. Parallel
123. \_\_\_ stands for input/output and refers to collecting data for the microprocessor to process and transporting results to an output device, like your monitor, or putting it in a storage device like your hard drive. I/O, which stands for input/output
124. Groupware requires computers to be \_\_\_\_\_\_\_\_\_\_\_\_ together. networked
125. \_\_\_\_\_\_\_\_\_\_\_ is where some Information System functions are hired out to a third party contractor. Outsourcing
126. A \_\_\_\_\_\_\_\_\_\_\_ card connects to the monitor. graphics
127. A \_\_\_\_\_\_ card is for transmitting data over phone lines. Modem
128. An \_\_\_\_\_\_\_\_\_\_\_ port is any connector that passes data in and out of a peripheral device or computer Expansion
129. A set of standard processing tasks that measure the performance of computer software or hardware is called a \_\_\_\_\_\_\_\_\_\_\_ test. benchmark
130. The first step in the software development cycle is to define the \_\_\_\_\_\_\_\_\_. problem
131. The size of a hard drive, today, is currently measured in \_\_\_\_\_\_\_\_, while it was once measured in megabytes or kilobytes gigabytes
132. An \_\_\_\_\_\_\_\_\_ system is also known as a knowledge-based system expert
133. In addition to access time, another measure of hard drive speed is \_\_\_\_, which is the measure of how fast a drive spins. Rpm
134. The \_\_\_\_\_\_\_\_\_ phase of the software development cycle involves making sure the program is consistently producing accurate and desired results. testing
135. High-performance workstations and servers will often use a \_\_\_\_ drive over an EIDE drive. SCSI
136. Computer \_\_\_\_\_\_\_\_\_ normally focuses on the design of computer hardware and peripheral devices. Engineering
137. Computer \_\_\_\_\_\_\_\_ normally focuses on making the computer work more efficiently and effectively. science
138. The highest Information Technology position in a company is the head of the IS department, the \_\_\_\_\_. CIO
139. Information \_\_\_\_\_\_\_\_\_ focuses mainly on the application of computers in an organizational or business environment. Systems
140. In a \_\_\_\_\_\_\_ type of interface, in addition to keyboard commands, you can also click on icons and menu choices. GUI
141. The instructions which are currently being executed are stored in \_\_\_\_\_. RAM
142. The maximum \_\_\_\_\_\_\_\_\_ of a monitor is the maximum number of pixels it can display. resolution
143. The \_\_\_\_\_\_\_\_\_ phase of the Software Development Life Cycle is when you would be most likely to first create flowcharts. design
144. A(n) \_\_\_\_\_ graphics card displays images more quickly than a normal graphics card. Accelerated
145. \_\_\_\_\_\_\_ memory stores images for the graphics card before they are displayed. Video
146. A document \_\_\_\_\_\_\_\_\_\_ can take a letter, or some other document you have, and convert it into a digital representation which it transmits to the computer. Scanner
147. A(n) \_\_\_\_\_ slot in a laptop uses \_\_\_\_\_\_\_ cards and is often used to add a modem or network card. PCMCIA
148. When a computer allows you to switch out devices while it is on, it is called \_\_\_ swap. HOT
149. A \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ is the person who interviews people to determine the requirements of a program, and designs the program. Systems Analyst
150. The quality of sharpness depends on the \_\_\_ that the printer can print. Dpi
151. A(n) \_\_\_\_\_ test is a test which is done by the software publisher's test team. alpha
152. 5. A(n) \_\_\_\_\_\_ test is a test which is done by a team of off-site testers. beta
153. Webmasters, programmers, and chip designers all depend on computers for the existence of their jobs. These jobs are called \_\_\_\_\_\_\_\_\_-\_\_\_\_\_\_\_\_\_\_ jobs. Computer-specific
154. A computer network which is restrained to a small area like a campus or building, is called a \_\_\_\_\_. lan
155. Within the same LAN, you can use different types of hardware, operating systems, and \_\_\_\_\_\_\_\_. cables
156. Networks which span a large area, like the entire world, are called \_\_\_\_\_'s. wan
157. A computer which is not connected to any kind of network is called a \_\_\_\_\_\_\_-\_\_\_\_\_\_ computer. Stand-alone
158. When you connect your computer to a LAN (local area network), it becomes a \_\_\_\_\_\_\_\_\_\_\_. Workstation
159. Your computer's physical resources are called \_\_\_\_\_\_ resources. local
160. The resources of the network which you have access to, such as a printer or other computer's hard drives, are called \_\_\_\_\_\_\_ resources. network
161. A network \_\_\_\_\_\_\_\_ is a computer which serves the other computers on the network. server
162. Each \_\_\_\_ is a device on the network. node
163. A network \_\_\_\_\_\_\_ is also known as a network supervisor and creates user accounts as well as manages the network Administrator
164. \_\_\_\_\_\_ is the protocol used on the internet for transferring large files FTP (file transfer protocol)
165. Drive \_\_\_\_\_\_\_\_ is when you assign a drive letter to a network drive mapping
166. When multiple users use one copy of software running off of a server, it is called \_\_\_\_\_\_\_\_\_ a program. sharing
167. File \_\_\_\_\_\_\_\_\_ is a precaution which allows only one user to edit a data file. locking
168. The printer which you want the computer to print to when a printer is not specified is called the \_\_\_\_\_\_\_\_ printer default
169. A \_\_\_\_ is a small circuit board which allows the network to be possible by sending and receiving data NIC (network card)
170. The two most popular network types are Token Ring and \_\_\_\_\_\_\_. ethernet
171. \_\_\_\_\_\_\_\_\_ cable, which is sometimes referred to UTP or STP, has a RJ-45 connector on both ends. Twisted-pair
172. \_\_\_\_\_\_ cable looks similar to a cable-TV cable and has a BNC connector on each end. coaxial
173. Sometimes you will not use cables in a network, but will instead use radio or infrared signals. These networks are called \_\_\_\_\_\_\_\_\_\_ networks. wireless
174. A network \_\_\_\_ connects workstations and broadcasts every packet to all of its ports. hub
175. A dedicated \_\_\_\_ server is dedicated to providing programs and data for workstations but does not process any data. file
176. A \_\_\_\_\_\_\_\_\_\_ file server acts as both a file server and workstation. Non-dedicated
177. A \_\_\_\_\_ server receives print requests from other computers and prints them on its printer. print
178. On a spreadsheet, the name assigned to a column OR a row is known as a \_\_\_\_\_\_\_. label
179. A print \_\_\_\_\_\_ is where the print jobs are held before they are printed. queue
180. An \_\_\_\_\_\_\_\_\_\_ server is a computer which runs one application and returns the processed results of requests to the appropriate workstation. application
181. A \_\_\_\_ computer has many terminals which lets many people use the same computer. Each terminal has a keyboard and a screen but they do not process any data and they do not have a local hard drive, but instead use the \_\_\_\_\_ computer's resources. host
182. Your microcomputer can simulate a terminal by using terminal \_\_\_\_\_\_\_\_\_\_\_\_ software. Emulation
183. \_\_\_\_\_\_-\_\_\_\_\_\_ processing results in immediate updates(it processes the jobs as it gets them). real-time
184. composed of two parts: The Network \_\_\_\_\_\_ software which is installed on a file server, and the Network client software, which handles drive mapping, login information, and more. server
185. A \_\_\_\_\_\_ installation updates the Windows Registry and Start menu. It also copies some of the program files to your computer. This is used so that you can run the program off of the network server. workstation
186. The quality of a monitor or the image displayed on a monitor is measured by its \_\_\_\_\_\_\_\_\_\_\_\_. resolution
187. A \_\_\_\_\_\_ license allows multiple users to use the software. It is often much cheaper than buying many single-user licenses network
188. \_\_\_\_\_\_\_\_ software, which is also known as "document routing software," automatically takes a document from one person to the next after getting the necessary approval. workflow
189. E-mail is a \_\_\_\_\_\_\_-\_\_\_\_\_-\_\_\_\_\_\_\_\_\_ technology, since a server stores your messages and then forwards them to your workstation. Store-and-forward
190. The internet started with the \_\_\_\_\_\_\_, which was created in 1969, and connected computers at four universities. ARPANET
191. A computer on the internet that provides a service is known as an \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_. internet host
192. \_\_\_\_\_\_\_\_\_\_\_\_ software lets your computer transmit and receive data using TCP/IP. Internet communications
193. The ideal password is a \_\_\_\_\_\_\_\_\_ alphanumeric arrangement. random
194. A \_\_\_\_\_\_ connects computers in one office or building lan
195. A \_\_\_\_\_\_\_\_ connects several offices scattered across the country. wan (wide area network)
196. The protocol, or set of communications rules, which is most used on the Internet is \_\_\_\_\_\_. TCP/IP
197. An \_\_\_\_\_\_\_\_\_ service provider is a company that provides Internet access to individuals, organizations, and businesses. internet
198. An Applications \_\_\_\_\_\_\_\_\_\_\_\_ is a person in a company who designs and creates programs to meet end-users needs. developer
199. A connection which uses a phone line to temporarily connect to the internet is called a \_\_\_\_\_\_-\_\_\_ connection. dial up
200. The unique number which refers to every computer which is connected to the Internet is called an \_\_\_\_\_ \_\_\_\_\_\_\_\_\_. IP address
201. 192.161.12.143 is an example of an \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_. IP address
202. The port number on a computer for accessing FTP (File Transfer Protocol) is port \_\_\_\_\_. 21
203. The Internet backbone has many \_\_\_\_\_\_\_\_, which direct traffic by use of the IP address. routers
204. Sometimes referred to as a FQDN, most people refer to easy-to-remember names like cocacola.com as \_\_\_\_\_\_\_\_\_\_ names. domain
205. The \_\_\_\_\_-\_\_\_\_\_\_\_ domain of the domain name indicates whether it is a college, government agency, commercial business, non-profit organization, etc. top level
206. A web site is composed of many \_\_\_\_\_\_\_\_\_. web pages
207. Each and every web page on the Internet has a \_\_\_\_, which is an Internet address for web pages. URL
208. The acronym HTTP is short for \_\_\_\_\_\_\_\_\_ Transfer Protocol Hypertext
209. \_\_\_\_ servers are not part of the Web but are part of the internet and are often used to store and transfer files. FTP
210. On a webpage, a \_\_\_\_\_, sometimes called a hypertext \_\_\_\_\_ allows you to go to other pages through them. link
211. \_\_\_\_\_\_\_\_ is the process of taking a file from a remote computer and putting it on your computer's hard drive. downloading
212. When you are sending a file from your local computer to a remote computer, it is called \_\_\_\_\_\_\_\_\_\_\_. uploading
213. A discussion group takes place \_\_\_\_\_\_\_\_\_\_\_, which means that the participants in the conversation are not all online at the same time. It is similar to a bulletin board where everybody posts their comments and questions. Asynchronously
214. A record in a database or spreadsheet is made up of \_\_\_\_\_\_\_\_\_. fields
215. In a chat room, you can communicate \_\_\_\_\_\_\_\_\_\_\_, meaning you are talking to people who are currently online at the same time. Synchronously
216. In an HTML document, there are HTML \_\_\_\_\_\_\_ which act as commands to the internet browser. tags
217. All information in a computer, whether it's video, sound, text, pictures, etc., is stored as a string of \_\_\_\_\_\_. bits
218. \_\_\_\_\_\_\_\_\_\_\_ speed is the maximum speed that a modem can communicate with the modems owned by your ISP. connection
219. The speed that your computer can send or receive data is called your \_\_\_\_\_\_\_\_\_\_ rate. transfer
220. The most common transfer protocols for use on a terminal or modem are X-modem, Y-modem, Z-modem, and Kermit. Which one of these is generally the fastest for file transfers? \_\_\_\_\_\_\_\_\_ Z modem
221. A mistake which is made by the computer user is called an \_\_\_\_\_\_\_\_ error operator
222. A power \_\_\_\_\_\_\_ is where your computer loses all power by no fault of your own. This is normally from a malfunction at your local power plant or a downed power line. failure
223. A copy of data is called a \_\_\_\_\_\_\_. backup
224. With a \_\_\_\_\_\_\_\_\_\_ backup, you make a full backup in regular intervals and then make a \_\_\_\_\_\_\_\_\_\_ backup with all data that has changed since the last full backup. differential
225. In a \_\_\_\_\_\_\_\_ \_\_\_\_\_\_ LAN network, the computers are connected to form a loop, and use a token to pass a message around the network. token ring
226. A battery which provides power during power failures or power outages and provides a steady flow of power in case you have a power surge or spike is called a \_\_\_\_\_. UPS
227. A computer component's reliability is measured by a statistic which is called a \_\_\_\_\_. MTBF
228. The \_\_\_\_\_\_\_ of a virus is what it wants to accomplish, or its true mission on your computer. Payload
229. A \_\_\_\_\_ virus is a virus which attaches itself to a program like a game or application. File
230. A \_\_\_\_\_\_ propagates itself on a system, infecting files, but cannot spread to other computers without human intervention. Virus
231. A \_\_\_\_\_\_ \_\_\_\_\_\_\_\_ virus infects the files your computer uses when it is turned on, or its system files. boot sector
232. A \_\_\_\_\_\_ virus attaches itself to a worksheet or document and spreads when the user opens the infected file. Macro
233. A \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ is a program which appears to do something of use to the user, while it is actually doing something else. trojan horse
234. A \_\_\_\_\_\_\_ is a program which enters a computer and then propagates itself throughout the Internet. They normally do not destroy data, but instead slow down the computer and take up hard drive space. Worm
235. \_\_\_\_\_\_\_\_\_ software, sometimes called virus detection software such as McAfee VirusScan, can find and remove viruses. Antivirus
236. A \_\_\_\_\_\_\_\_\_ diagram is used to show how data flows to and from processes in a system. data flow
237. A \_\_\_\_\_\_\_\_\_ is a number used to determine if any byte within a program has been changed. Checksum
238. A virus \_\_\_\_\_\_\_\_\_\_\_ is a series of bytes which is unique for a certain virus. It acts as an identifier which Antivirus software can use. Signature
239. There are three files on a disk, an ASCII text file, a word processor document, and a sound file. You can only delete one file, but you want to free up the most space possible. Which file should you delete to free up the most space? \_\_\_\_\_\_\_\_ sound
240. With an \_\_\_\_\_\_\_\_\_\_\_ backup, you make a full backup at regular intervals, while using a separate tape to store the files that change each day after that. Incremental
241. Rules that limit what each user can do are called \_\_\_\_\_\_ \_\_\_\_\_\_\_. user rights
242. A special hole left by the programmer for emergency situations that can be used by a hacker to enter a system without having to hack each security precaution is called a \_\_\_\_\_\_\_ \_\_\_\_\_\_. trap door
243. \_\_\_\_\_\_\_\_\_\_\_ is used to scramble information, so that it cannot be understood unless it is properly deciphered or decrypted. Encryption
244. \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_ encryption uses two keys, one key which can encrypt the message, and one which can decrypt the message. public key
245. Companies will often have a \_\_\_\_\_\_\_\_\_\_ which will help keep hackers and potentially hazardous programs from getting on your company computer. Firewall
246. \_\_\_\_\_\_\_ stores information for a website so that it can "remember" you, when you come back. Cookie
247. \_\_\_\_\_\_\_\_\_\_\_\_ refers to the time that a computer system is not available for use downtime
248. \_\_\_\_\_\_\_\_\_\_\_\_ refers to the time that a computer system is not available for use Redundant
249. One bit can give \_\_\_\_ different messages. 2
250. The binary system is base \_\_\_\_. 2
251. ASCII is one of the most popular character representation codes and uses \_\_ bits which allows it to have 128 total characters. 7
252. One of the most popular character representation codes, ANSI uses \_\_ bits to represent 256 different characters. 8
253. \_\_\_\_\_\_\_\_\_\_ is a character representation code which uses 16 bits to represent 65536 characters unicode
254. Data \_\_\_\_\_\_\_\_\_\_\_\_ is the process used to shrink files so that they take up less space. Compression
255. \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ compresses files into one smaller file.File compreesion
256. 1 Kilobyte equals \_\_\_\_\_\_ bytes 1024
257. \_\_\_\_\_\_\_\_\_ is the amount of data that can be sent over a communications channel line in one second. Bandwidth
258. Transmissions which send one bit after another are known as \_\_\_\_\_\_\_\_\_\_\_ transmissions. Serial
259. When transmitting data by \_\_\_\_\_\_\_\_\_ transmission, all the bits of a byte are sent at the same time. Parallel
260. The \_\_\_\_\_\_\_\_\_ of a network is the layout of the communication channels in a communications system topology
261. The \_\_\_\_\_\_\_\_\_ topology connects all computers directly to one device, usually a switch. Star
262. The \_\_\_\_\_\_\_ topology hooks each computer up to its neighbors in a long chain. Bus
263. The \_\_\_\_\_\_\_\_ topology hooks each computer to its neighbor and the last computer to the first, making a loop. Ring
264. The \_\_\_\_\_\_\_\_\_ topology connects each computer to every other computer. Mesh
265. One way of checking if a transmitted byte was sent accurately is by using a \_\_\_\_\_\_\_ bit, which has information on the number of 1 bits. Parity
266. A \_\_\_\_\_\_\_ protocol transmits data at a fixed rate agreed by the sender and receiver Synchronous
267. An \_\_\_\_\_\_\_\_\_\_\_\_ protocol transmits data with start and stop bits. Asynchronous
268. \_\_\_\_\_\_\_\_\_\_ communication lets you transmit but not receive, or vice versa simplex
269. \_\_\_\_\_\_-\_\_\_\_\_\_\_\_ communication lets you transmit and receive, but not at the same time half duplex
270. \_\_\_\_\_\_-\_\_\_\_\_\_\_\_ communication lets you send and receive at the same time. full duplex
271. An \_\_\_\_\_\_\_\_\_\_ system gathers, stores, and provides information to people information
272. A \_\_\_\_\_\_\_\_\_\_ is a website that offers a broad range of resources and services--i.e. most search engines nowadays also offer news, email, weather, sports updates, etc.. portal
273. An \_\_\_\_\_\_\_\_\_\_\_\_ is a set of people who work together to accomplish a set goal. Organization
274. A \_\_\_\_\_\_\_\_\_\_ statement tells what an organization hopes to achieve. Mission
275. \_\_\_\_\_\_\_\_\_ is the use of computers or other machines to make certain processes more efficient. Automation
276. \_\_\_\_\_\_\_\_\_\_\_\_ is to input, record, or process information in a computer or system of computers. Computerization
277. \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ management keeps records of the employees employed in an organization along with their salaries, skills, etc Human resources
278. There are two types of information: \_\_\_\_\_\_\_\_\_ and internal. External
279. In a spreadsheet, the intersection of a row and a column is known as a \_\_\_\_\_\_\_\_. Cell
280. \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ tools help people find logical solutions to problems by letting them make a model of their problem. Information analysis
281. An organization which automates its day-to-day office tasks uses an \_\_\_\_\_\_\_\_\_\_.system Automation
282. A Cell \_\_\_\_\_\_\_ can consist of just one cell, or a combination of one or more rows and columns. Range
283. An \_\_\_\_\_\_\_\_\_ is an internal network used by a company to provide its employees with access to information Intranet
284. A \_\_\_\_\_\_\_\_\_ support system helps employees make decisions for semi-structured problems decisions
285. A knowledge-based system, or \_\_\_\_\_\_ system, analyzes data and comes up with a decision. EXPERT
286. When developing expert system applications, it helps to have an expert system \_\_\_\_\_. Shell
287. Using a method called \_\_\_\_\_\_\_ logic, an expert system can take unsure data, along with the percent of confidence in your unsure data and give you an answer along with its percentage of having given you a correct answer. Fuzzy
288. \_\_\_\_\_\_\_\_ networks simulate the brain and can learn, remember, and even process information. Neural
289. A \_\_\_\_\_\_\_\_ system stores items purchased and calculates the total cost for each sale. point of sale

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++