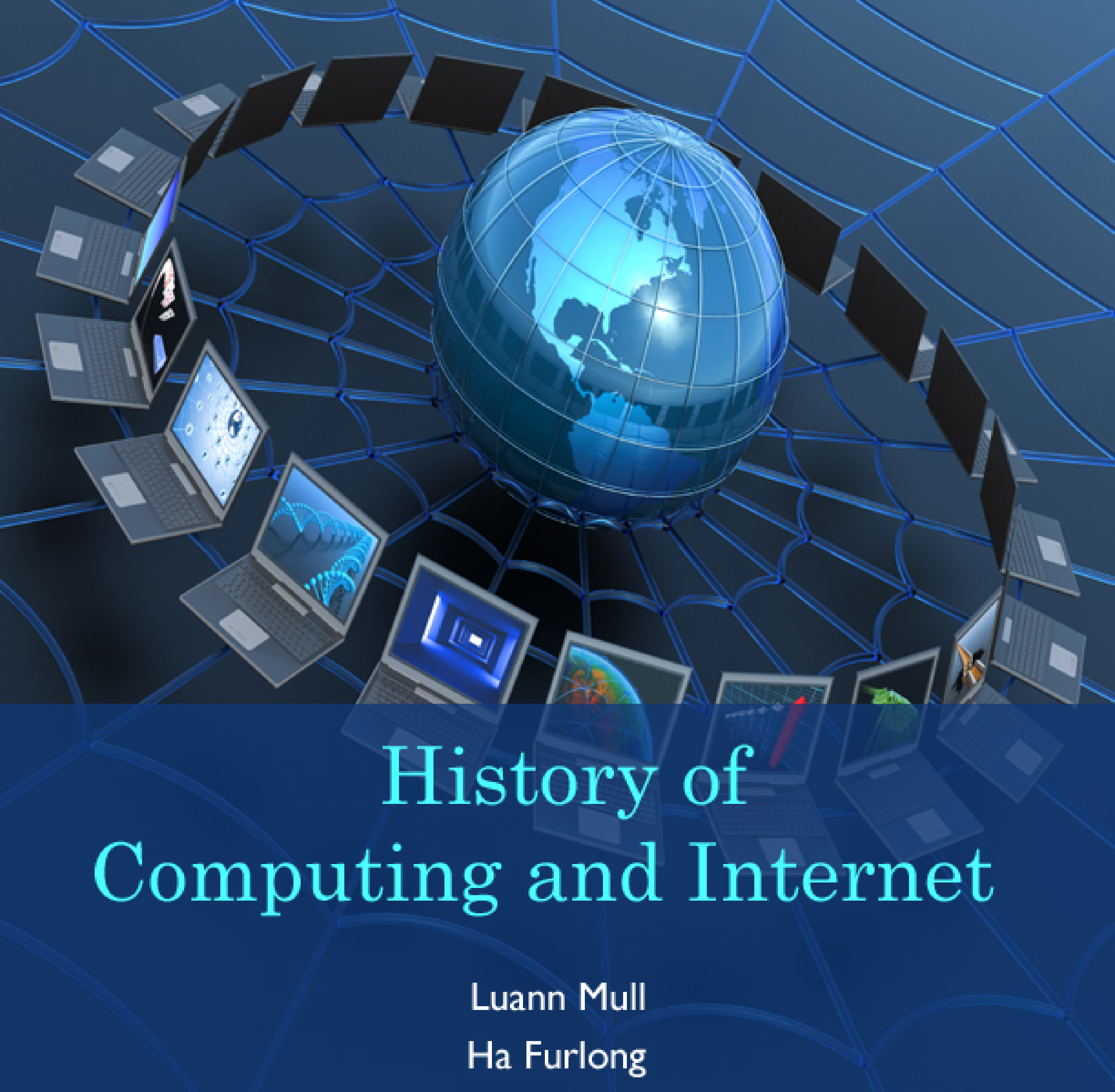


History of Computing and Internet

Luann Mull

Ha Furlong

First Edition, 2012

ISBN 978-81-323-1276-5

Published by:
College Publishing House
4735/22 Prakashdeep Bldg,
Ansari Road, Darya Ganj,
Delhi - 110002
Email: info@wtbooks.com

Table of Contents

Chapter 1

History of Computing

The **history of computing** is longer than the history of computing hardware and modern computing technology and includes the history of methods intended for pen and paper or for chalk and slate, with or without the aid of tables. The **timeline of computing** presents a summary list of major developments in computing by date.

Concrete devices

Computing is intimately tied to the representation of numbers. But long before abstractions like *the number* arose, there were mathematical concepts to serve the purposes of civilization. These concepts are implicit in concrete practices such as :

- *one-to-one correspondence*, a rule to count *how many* items, say on a tally stick, eventually abstracted into *numbers*;
- *comparison to a standard*, a method for assuming *reproducibility* in a measurement, for example, the number of coins;
- the *3-4-5* right triangle was a device for assuring a *right angle*, using ropes with 12 evenly spaced knots, for example.

Numbers

Eventually, the concept of numbers became concrete and familiar enough for counting to arise, at times with sing-song mnemonics to teach sequences to others. All the known languages have words for at least "one" and "two" (although this is disputed), and even some animals like the blackbird can distinguish a surprising number of items.

Advances in the numeral system and mathematical notation eventually led to the discovery of mathematical operations such as addition, subtraction, multiplication, division, squaring, square root, and so forth. Eventually the operations were formalized, and concepts about the operations became understood well enough to be stated formally, and even proven. See, for example, Euclid's algorithm for finding the greatest common divisor of two numbers.

By the High Middle Ages, the positional Hindu-Arabic numeral system had reached Europe, which allowed for systematic computation of numbers. During this period, the representation of a calculation on paper actually allowed calculation of mathematical expressions, and the tabulation of mathematical functions such as the square root and the common logarithm (for use in multiplication and division) and the trigonometric functions. By the time of Isaac Newton's research, paper or vellum was an important computing resource, and even in our present time, researchers like Enrico Fermi would cover random scraps of paper with calculation, to satisfy their curiosity about an equation. Even into the period of programmable calculators, Richard Feynman would unhesitatingly compute any steps which overflowed the memory of the calculators, by hand, just to learn the answer.

Early computation

The earliest known tool for use in computation was the abacus, and it was thought to have been invented in Babylon circa 2400 BC. Its original style of usage was by lines drawn in sand with pebbles. Abaci, of a more modern design, are still used as calculation tools today. This was the first known computer and most advanced system of calculation known to date - preceding Greek methods by 2,000 years.

In 1115 BC, the South Pointing Chariot was invented in ancient China. It was the first known geared mechanism to use a differential gear, which was later used in analog computers. The Chinese also invented a more sophisticated abacus from around the 2nd century BC known as the Chinese abacus).

In the 5th century BC in ancient India, the grammarian Pāṇini formulated the grammar of Sanskrit in 3959 rules known as the Ashtadhyayi which was highly systematized and technical. Panini used metarules, transformations and recursions.

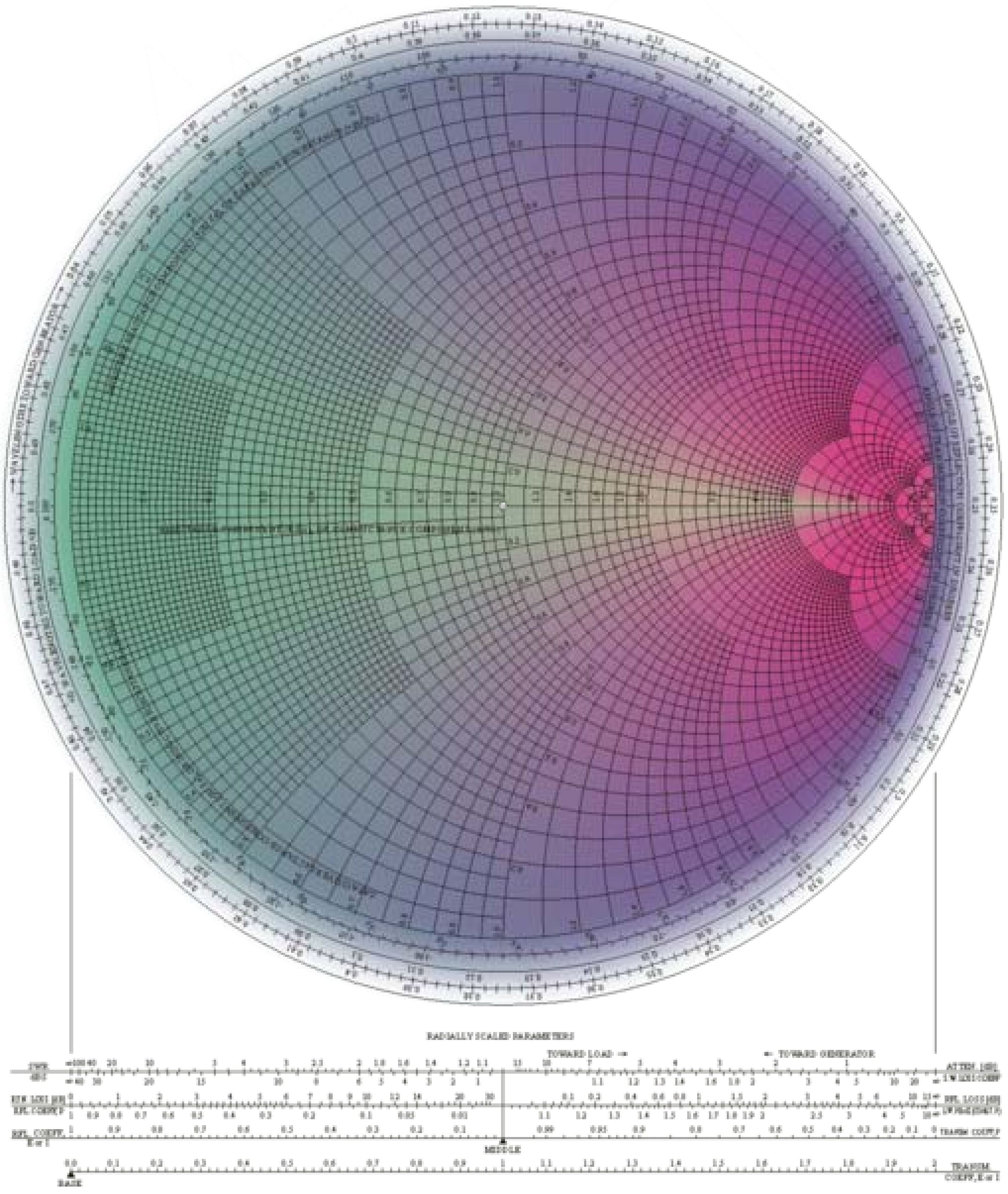
The Antikythera mechanism is believed to be the earliest known mechanical analog computer. It was designed to calculate astronomical positions. It was discovered in 1901 in the Antikythera wreck off the Greek island of Antikythera, between Kythera and Crete, and has been dated to *circa* 100 BC.

Mechanical analog computer devices appeared again a thousand years later in the medieval Islamic world and were developed by Muslim astronomers, such as the equatorium by Arzachel, the mechanical geared astrolabe by Abū Rayhān al-Bīrūnī, and the torquetum by Jabir ibn Aflah. Muslim mathematicians also made important advances in cryptography, such as the development of cryptanalysis and frequency analysis by Alkindus. Programmable machines were also invented by Muslim engineers, such as the automatic flute player by the Banū Mūsā brothers, and Al-Jazari's humanoid robots and *castle clock*, which is considered to be the first programmable analog computer.

During the Middle Ages, several European philosophers made attempts to produce analog computer devices. Influenced by the Arabs and Scholasticism, majorcan philosopher Ramon Llull (1232–1315) devoted a great part of his life to define and design several *logical machines* that, by combining simple and undeniable philosophical truths, could produce all the possible knowledge. This machines were never really built, for they were more of a thought experiment devoted to the production of new knowledge by systematic ways; however they could make simple logical operations, they still needed a human being for interpretation of results. Moreover, they lacked of a versatile architecture, each machine serving only to very concrete purposes. No matter what, Llull's work had a severe impact on Gottfried Leibniz (early 18th century), who redeveloped his ideas further and could build several calculating tools with them.

Indeed, when John Napier discovered logarithms for computational purposes in the early 17th century, there followed a period of considerable progress by inventors and scientists in making calculating tools. The apex of this early era of formal computing can be seen in the difference engine and its successor the Analytical Engine, which was never completely constructed but was designed in detail, both by Charles Babbage. The analytical engine combined concepts from his work and that of others to create a device that if constructed as designed would have possessed many properties of a modern electronic computer. These properties include such features as an internal "scratch memory" equivalent to RAM, multiple forms of output including a bell, a graph-plotter, and simple printer, and a programmable input-output "hard" memory of punch cards which it could modify as well as read. The key advancement which Babbage's devices possessed beyond those created before his was that each component of the device was independent of the rest of the machine, much like the components of a modern electronic computer. This was a fundamental shift in thought; previous computational devices served only a single purpose, but had to be at best dissasembled and reconfigured to solve a new problem. Babbage's devices could be reprogramed to solve new problems by the entry of new data, and act upon previous calculations within the same series of instructions. Ada Lovelace took this concept one step further, by creating a program for the analytical engine to calculate Bernoulli numbers, a complex calculation requiring a recursive algorithm. This is considered to the first example of a true computer program, a series of instructions that act upon data not known in full until the program is run.

Several examples of analog compututation survived into recient times. A planimeter is a device which does integrals, using distance as the analog quantity. Until the 1980s, HVAC systems used air both as the analog quantity and the controlling element. Unlike modern digital computers, analog computers are not very flexible, and need to be reconfigured (i.e., reprogrammed) 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.



A Smith Chart is a well-known nomogram.

Since computers were rare in this era, the solutions were often *hard-coded* into paper forms such as nomograms, which could then produce analog solutions to these problems, such as the distribution of pressures and temperatures in a heating system.

None of the early computational devices were really computers in the modern sense, and it took considerable advancement in mathematics and theory before the first modern computers could be designed.

Navigation and astronomy

Starting with known special cases, the calculation of logarithms and trigonometric functions can be performed by looking up numbers in a mathematical table, and interpolating between known cases. For small enough differences, this linear operation was accurate enough for use in navigation and astronomy in the Age of Exploration. The uses of interpolation have thrived in the past 500 years: by the twentieth century Leslie Comrie and W.J. Eckert systematized the use of interpolation in tables of numbers for punch card calculation.

In our time, even a student can simulate the motion of the planets, an N-body differential equation, using the concepts of numerical approximation, a feat which even Isaac Newton could admire, given his struggles with the motion of the Moon.

Weather prediction

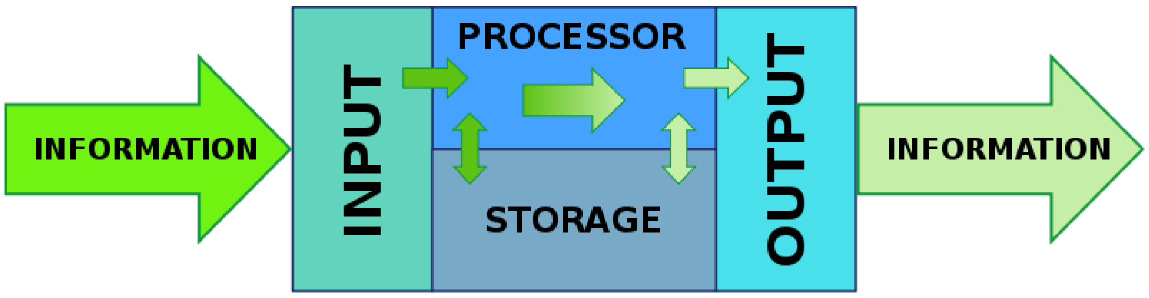
The numerical solution of differential equations, notably the Navier-Stokes equations was an important stimulus to computing, with Lewis Fry Richardson's numerical approach to solving differential equations. To this day, some of the most powerful computer systems of the Earth are used for weather forecasts.

Symbolic computations

By the late 1960s, computer systems could perform symbolic algebraic manipulations well enough to pass college-level calculus courses.

Chapter 2

History of Computing Hardware



Computing hardware is a platform for information processing (block diagram)

The **history of computing hardware** is the record of the ongoing effort to make computer hardware faster, cheaper, and capable of storing more data.

Before the development of the general-purpose computer, most calculations were done by humans. Tools to help humans calculate 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; there are pictures of enormous rooms filled with desks at which computers (often young women) used their machines to jointly perform calculations, as for instance, aerodynamic ones required for in aircraft design.

Calculators have continued to develop, but computers add the critical element of conditional response and larger memory, allowing automation of both numerical calculation and in general, automation of many symbol-manipulation tasks. Computer technology has undergone profound changes every decade since the 1940s.

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.

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 through manual manipulations 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 WWII were of this type. Or, 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.

Both analog and digital mechanical techniques continued to be developed, producing many practical computing machines. Electrical methods rapidly improved the speed and precision of calculating machines, at first by providing motive power for mechanical calculating devices, and later directly as the medium for representation of numbers. Numbers could be represented by voltages or currents and manipulated by linear electronic amplifiers. Or, numbers could be represented as discrete binary or decimal digits, and electrically controlled switches and combinational circuits could perform mathematical operations.

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.

Here we covers major developments in the history of computing hardware, and attempts to put them in context. The history of computing article treats methods intended for pen and paper, with or without the aid of tables. Since 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.

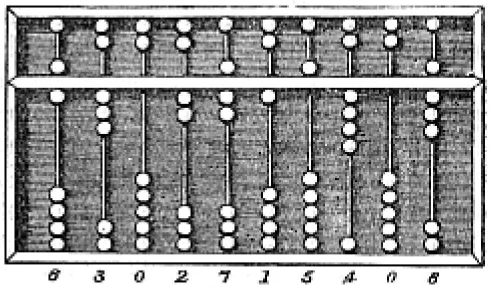
Earliest true hardware

Devices have been used to aid computation for thousands of years, mostly using one-to-one correspondence with our fingers. The earliest counting device was probably a form of tally stick. Later record keeping aids throughout the Fertile Crescent included calculi (clay spheres, cones, etc.) which represented counts of items, probably livestock or grains, sealed in containers. Counting rods is one example.

The abacus was early used for arithmetic tasks. What we now call the Roman abacus was used in 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, 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.

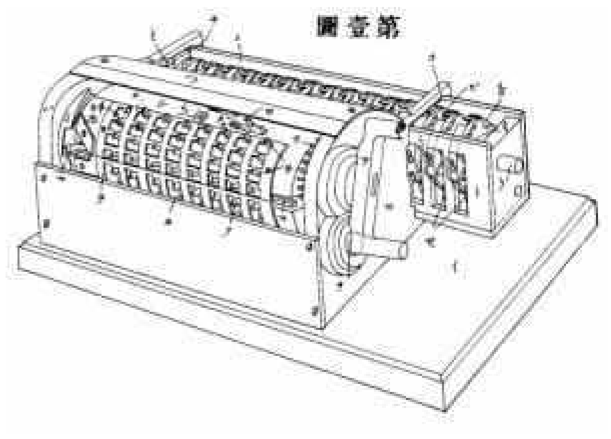
Several analog computers were constructed in ancient and medieval times to perform astronomical calculations. These include the Antikythera mechanism and the astrolabe from ancient Greece (c. 150–100 BC), which are generally regarded as the earliest known mechanical analog computers. Other early versions of mechanical devices used to perform one or another type of calculations include the planisphere and other mechanical computing devices invented by Abū Rayhān al-Bīrūnī (c. AD 1000); the equatorium and universal latitude-independent astrolabe by Abū Ishāq Ibrāhīm al-Zarqālī (c. AD 1015); the astronomical analog computers of other medieval Muslim astronomers and engineers; and the astronomical clock tower of Su Song (c. AD 1090) during the Song Dynasty.

The "castle clock", an astronomical clock invented by Al-Jazari in 1206, is thought to be the earliest programmable analog computer. It displayed the zodiac, the solar and lunar orbits, a crescent moon-shaped pointer traveling across a gateway causing automatic doors to open every hour, and five robotic musicians who play music when struck by levers operated by a camshaft attached to a water wheel. The length of day and night could be re-programmed every day in order to account for the changing lengths of day and night throughout the year.



Suanpan (the number represented on this abacus is 6,302,715,408)

Scottish mathematician and physicist 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, an abacus-like device used for multiplication and division. Since real numbers can be represented as distances or intervals on a line, the slide rule was invented in the 1620s to allow multiplication and division operations to be carried out significantly faster than was previously possible. Slide rules were used by generations of engineers and other mathematically involved professional workers, until the invention of the pocket calculator.



Yazu Arithmometer. Patented in Japan in 1903. Note the lever for turning the gears of the calculator.

Wilhelm Schickard, a German polymath, designed a calculating clock in 1623, unfortunately a fire destroyed it 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.

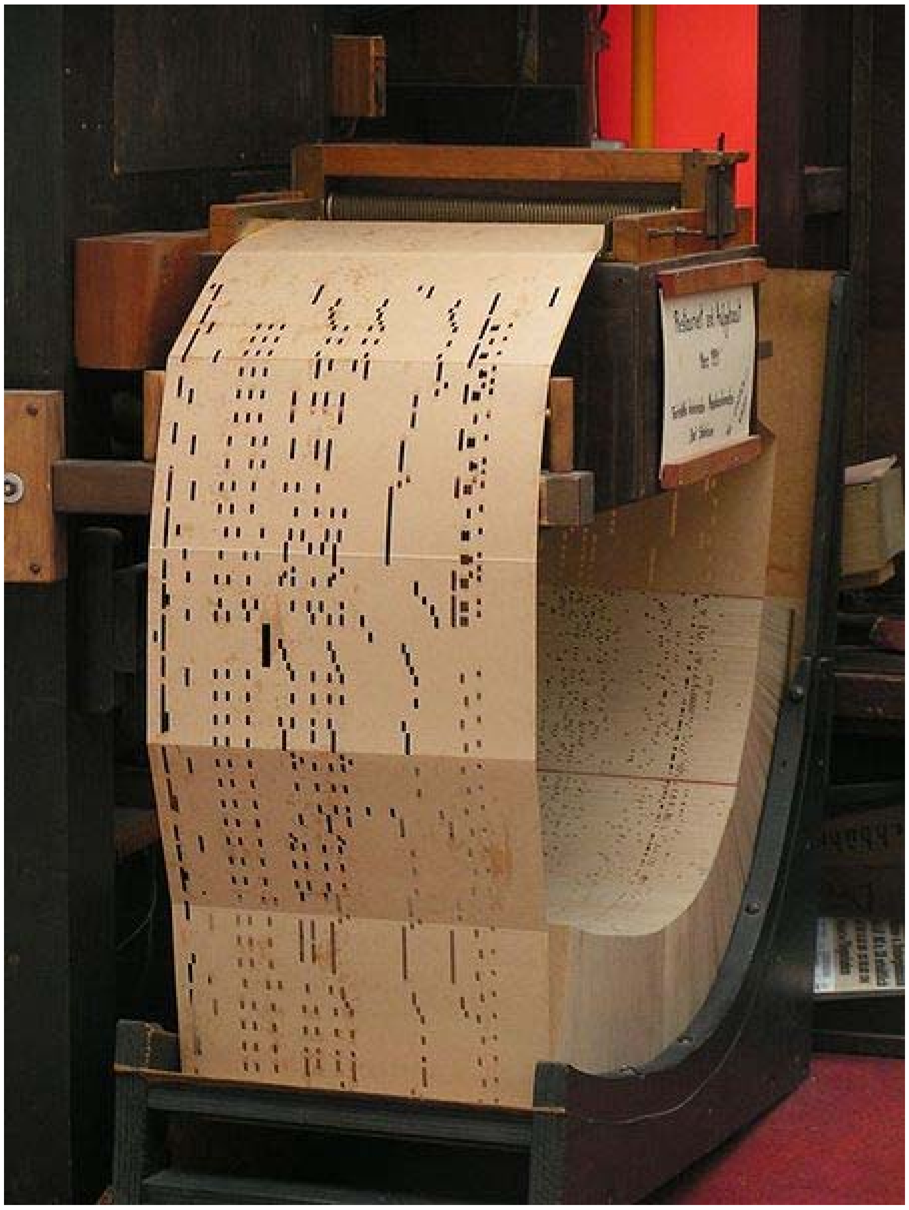
In 1642, while still a teenager, Blaise Pascal started some pioneering work on calculating machines and after three years of effort and 50 prototypes he invented the mechanical calculator. He built twenty of these machines (called the Pascaline) in the following ten years.

Gottfried Wilhelm von Leibniz invented the Stepped Reckoner and his famous cylinders 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."

Around 1820, Charles Xavier Thomas created the first successful, mass-produced mechanical calculator, the Thomas Arithmometer, that could add, subtract, multiply, and divide. It was mainly based on Leibniz' work. Mechanical calculators, like the base-ten addiator, the comptometer, the Monroe, the Curta and the Addo-X remained in use until the 1970s. Leibniz also described the binary numeral system, a central ingredient of all modern computers. However, up to the 1940s, many subsequent designs (including Charles Babbage's machines of the 1822 and even ENIAC of 1945) were based on the decimal system; ENIAC's ring counters emulated the operation of the digit wheels of a mechanical adding machine.

In Japan, Ryōichi 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 to the soroban (Japanese abacus). Carry and end of calculation were determined automatically. More than 200 units were sold, mainly to government agencies such as the Ministry of War and agricultural experiment stations.

1801: punched card technology



Punched card system of a music machine, also referred to as Book music

In 1801, Joseph-Marie Jacquard developed a loom in which the pattern being woven was controlled by punched cards. The series of cards could be changed without changing the mechanical design of the loom. This was a landmark achievement in programmability.

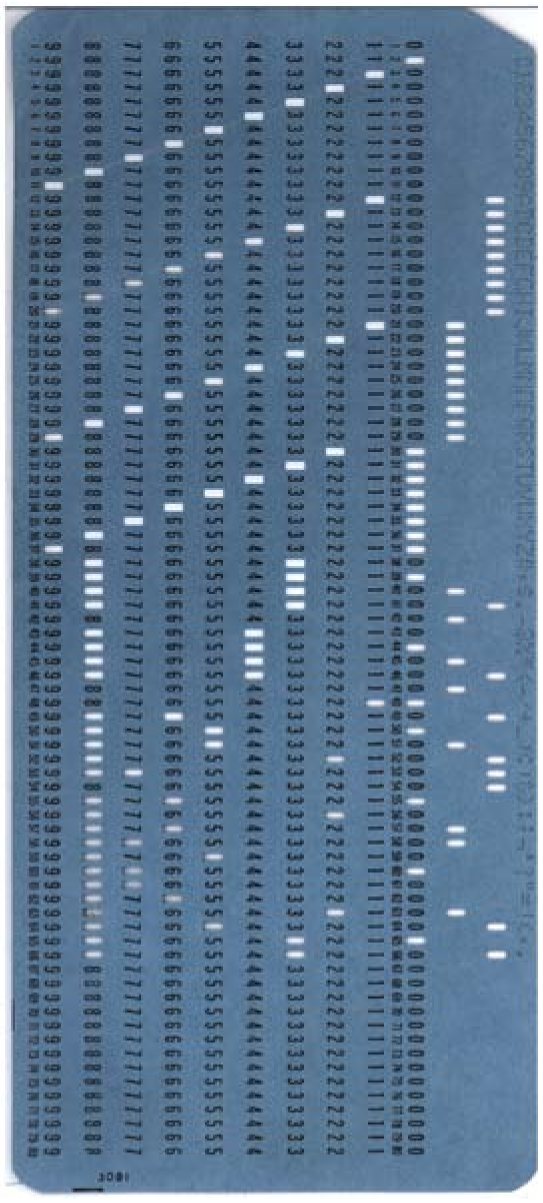
In 1833, Charles Babbage moved on from developing his difference engine (for navigational calculations) to a general purpose design, the Analytical Engine, which drew directly on Jacquard's punched cards for its program storage. In 1835, Babbage described his analytical engine. It was a general-purpose programmable computer, employing punch cards for input and a steam engine for power, using the positions of gears and shafts to represent numbers. 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, 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, Lord Byron's daughter, translated and added notes to the "*Sketch of the Analytical Engine*" by Federico Luigi, Conte Menabrea. this appears to be the first published description of programming.

IBM 407 tabulating machine, (1961)

A reconstruction of the Difference Engine II, an earlier, more limited design, has been operational since 1991 at the 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.

Following Babbage, although unaware of his earlier work, was 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.

In the late 1880s, the American 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 such as piano rolls or looms), not data. "After some initial trials with paper tape, he settled on punched cards..." Hollerith came to use punched cards after observing how railroad conductors encoded personal characteristics of each passenger with punches on their tickets. To process these punched cards he invented the tabulator, and the key punch machine. These three inventions were the foundation of the modern information processing industry. His machines used mechanical relays (and solenoids) to increment mechanical counters. Hollerith's method was used in the 1890 United States Census and the completed results were "... finished months ahead of schedule and far under budget". Indeed years faster than the prior census had required. Hollerith's company eventually became the core of IBM. IBM developed punch card technology into a powerful tool for business data-processing and produced an extensive line of 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 or mutilate," became a catch phrase for the post-World War II era.



Punched card with the extended alphabet

Leslie Comrie's articles on punched card methods and 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 or perform multiplication and division using floating point representations, all on punched cards and unit record machines. Those same machines had been used during WWII for cryptographic statistical processing. In the image of the tabulator, note the patch panel, which is visible on the right side of the tabulator. A row of toggle switches is above the patch panel. The Thomas J. Watson Astronomical Computing Bureau, Columbia University performed astronomical calculations representing the state of the art in computing.

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 stack 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. 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

The Curta calculator can also do multiplication and division

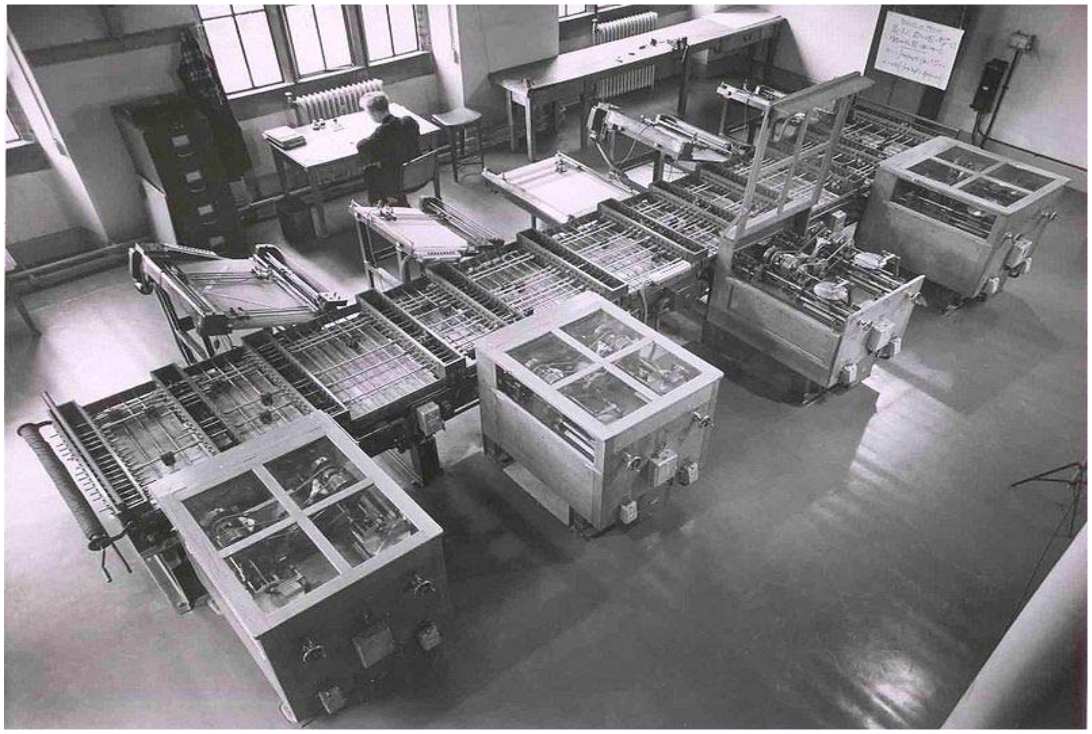
By the 20th century, earlier mechanical calculators, cash registers, accounting machines, and so on were redesigned to use electric motors, with gear position as the representation for the state of a variable. The word "computer" was a job title assigned to people who used these calculators to perform mathematical calculations. By the 1920s Lewis Fry Richardson's interest in weather prediction led him to propose human computers and numerical analysis to model the weather; to this day, the most powerful computers on Earth are needed to adequately model its weather using the Navier-Stokes equations.

Companies like Friden, Marchant Calculator and Monroe made desktop mechanical calculators from the 1930s that could add, subtract, multiply and divide. During the Manhattan project, future Nobel laureate Richard Feynman was the supervisor of the roomful of human computers, many of them female mathematicians, who understood the use of differential equations which were being solved for the war effort.

In 1948, the Curta was introduced. This was a small, portable, mechanical calculator that was about the size of a 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, which used a Nixie tube display and 177 subminiature 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, and introduced 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 produced the LOCI-2, a 10-digit transistorized desktop calculator that used a Nixie tube display and could compute logarithms.

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



Cambridge differential analyzer, 1938

Before World War II, mechanical and electrical analog computers 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 and currents as the analog quantities.

Centrally, these analog systems work by creating electrical *analogs* of other systems, allowing users to predict behavior of the systems of interest by observing the electrical analogs. 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 built in 1928; an electrical example is the Mallock machine built in 1941. A planimeter is a device which does integrals, using 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 and the fire-control systems, such as Arthur Pollen's Argo system for naval vessels. Some stayed in use for decades after WWII; the Mark I Fire Control Computer was deployed by the United States Navy on a variety of ships from destroyers to battleships. Other analog computers included the Heathkit EC-1, and the hydraulic MONIAC Computer which modeled econometric flows.

The art of mechanical analog computing reached its zenith with the differential analyzer, built by H. L. Hazen and Vannevar Bush at MIT starting in 1927, which in turn built on the mechanical integrators invented in 1876 by James Thomson and the torque amplifiers invented by 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's Moore School of Electrical Engineering, where the 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. But like all digital devices, the decimal precision of a digital device is a limitation, as compared to an analog device, in which the accuracy is a limitation. As electronics progressed during the 20th century, its problems of operation at low voltages while maintaining high signal-to-noise ratios were steadily addressed, as shown below, for a digital circuit is a specialized form of analog circuit, intended to operate at standardized settings (continuing in the same vein, logic gates can be realized as forms of digital circuits). But as digital computers have become faster and use larger memory (for example, RAM or internal storage), they have almost entirely displaced analog computers. Computer programming, or coding, has arisen as another human profession.

Electronic digital computation

Friden paper tape punch. 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, as electronic circuit elements replaced mechanical equivalents, and digital calculations replaced analog calculations. Machines such as the Z3, the Atanasoff–Berry Computer, the Colossus computers, and the ENIAC were built by hand using circuits containing relays or valves (vacuum tubes), and often used punched cards or punched paper 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.

Alan Turing's 1936 paper proved enormously influential in computing and computer science in two ways. Its main purpose was to prove that there were problems (namely the 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. Except for the limitations imposed by their finite memory stores, modern computers are said to be Turing-complete, which is to say, they have algorithm execution capability equivalent to a universal Turing machine.

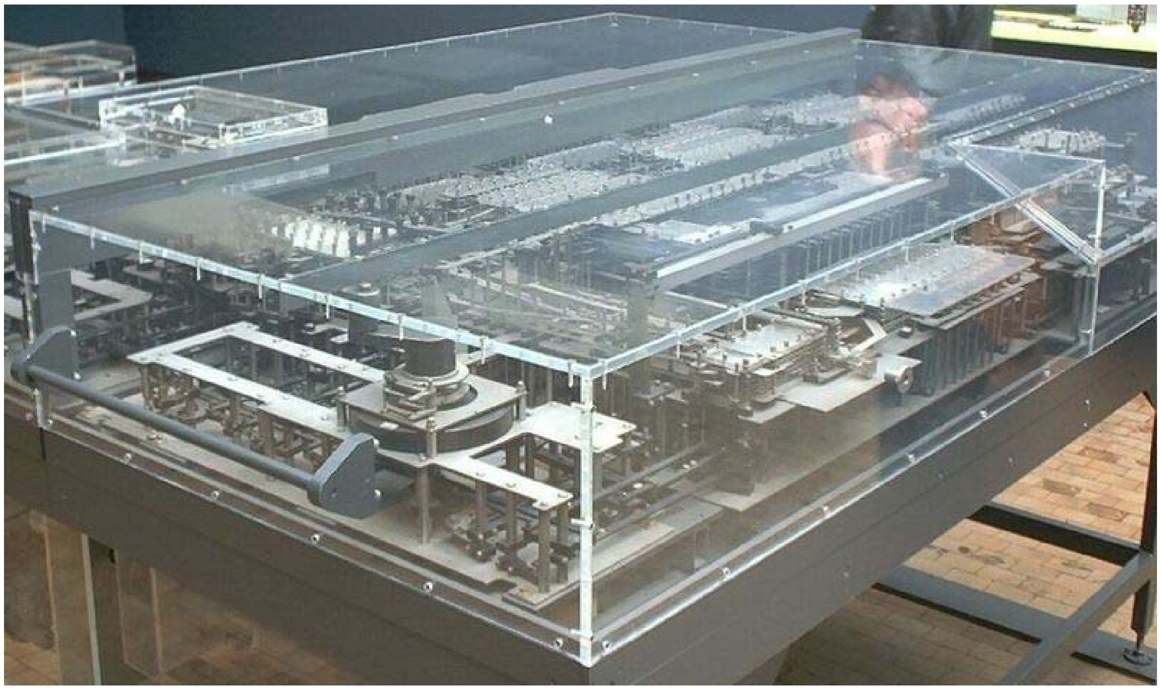
Nine-track magnetic tape

For a computing machine to be a practical general-purpose computer, there must be some convenient read-write mechanism, punched tape, for example. With a knowledge of Alan Turing's theoretical 'universal computing machine' John von Neumann defined an architecture which uses the same 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. 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. The third stream of computer development, Eckert and Mauchly's ENIAC and EDVAC, was widely publicized.

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.

Zuse



A reproduction of Zuse's Z1 computer

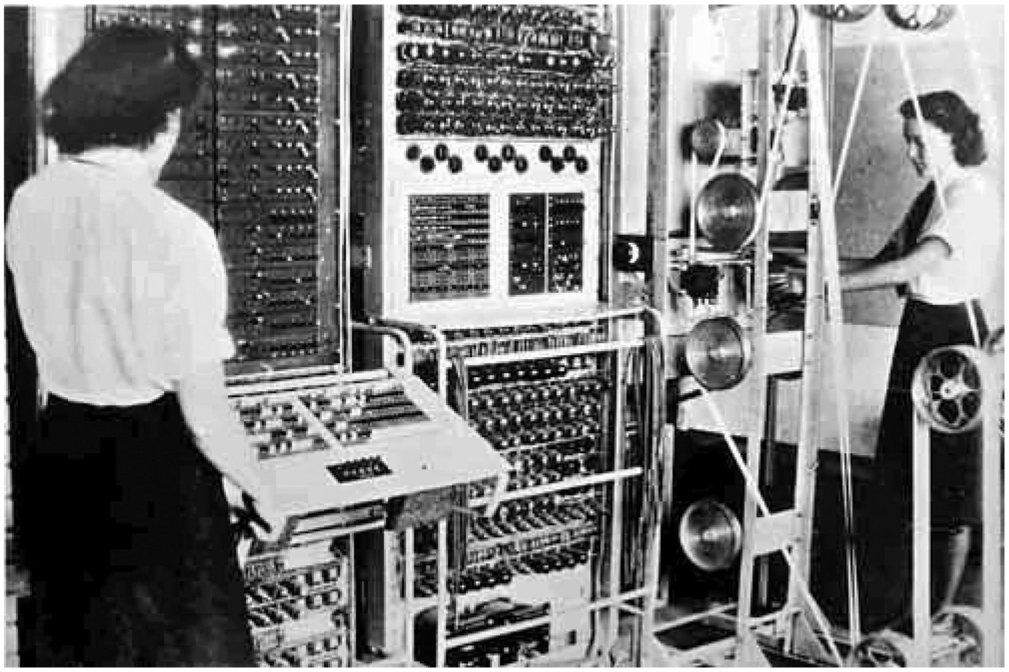
Working in isolation in Germany, Konrad Zuse started construction in 1936 of his first Z-series calculators featuring memory and (initially limited) programmability. Zuse's purely mechanical, but already binary Z1, finished in 1938, never worked reliably due to problems with the precision of parts.

Zuse's later machine, the Z3, was finished in 1941. It was based on telephone relays and did work satisfactorily. The Z3 thus became the 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. Replacement of the hard-to-implement decimal system (used in Charles Babbage's earlier design) by the simpler binary 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 (as always, ignoring physical storage limitations). In two 1936 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, first implemented in the British SSEM of 1948. Zuse also claimed to have designed the first higher-level programming language, which he named Plankalkül, in 1945 (published in 1948) although it was implemented for the first time in 2000 by a team around Raúl Rojas at the Free University of Berlin—five years after Zuse died.

Zuse suffered setbacks during World War II when some of his machines were destroyed in the course of Allied 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



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

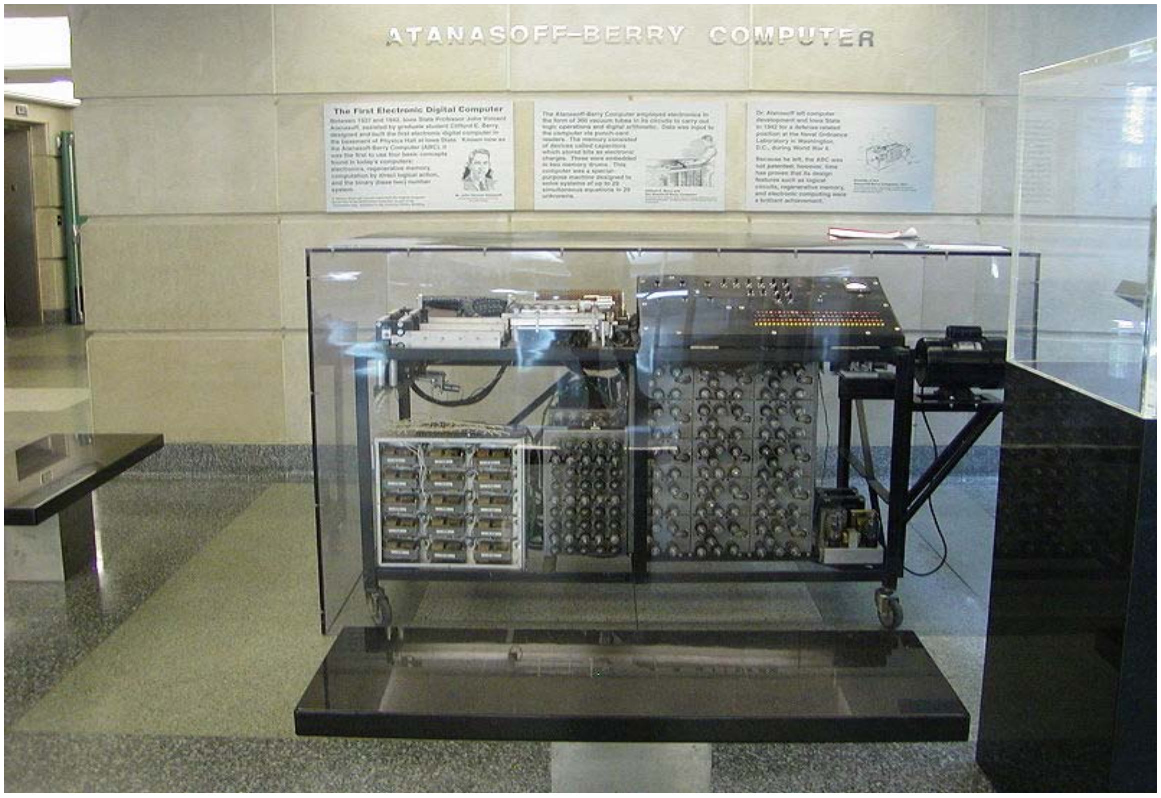
During World War II, the British at Bletchley Park (40 miles north of London) achieved a number of successes at breaking encrypted German military communications. The German encryption machine, Enigma, was attacked with the help of electro-mechanical machines called *bombes*. The bombe, designed by Alan Turing and Gordon Welchman, after the Polish cryptographic *bomba* by Marian Rejewski (1938), came into productive use in 1941. 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 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, Professor Max Newman and his colleagues helped specify the Colossus. The Mk I Colossus was built between March and December 1943 by Tommy Flowers and his colleagues at the Post Office Research Station at Dollis Hill in London and then shipped to Bletchley Park in January 1944.

Colossus was the world's first totally *electronic* programmable computing device. The Colossus 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 operations on its data, but it was not 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 personally issued an order for their destruction into pieces no larger than a man's hand, which was to ensure the fact the British were capable of cracking Lorenz was kept secret during the oncoming cold war. Due to this secrecy, the Colossi were not included in many histories of computing. A reconstructed copy of one of the Colossus machines is now on display at Bletchley Park.

American developments

In 1937, Claude Shannon showed there is a one-to-one correspondence between the concepts of Boolean logic and certain electrical circuits, now called logic gates, which are now ubiquitous in digital computers. In his master's thesis at MIT, for the first time in history, Shannon showed that electronic relays and switches can realize the expressions of Boolean algebra. Entitled *A Symbolic Analysis of Relay and Switching Circuits*, Shannon's thesis essentially founded practical digital circuit design. George Stibitz completed a relay-based computer he dubbed the "Model K" at Bell Labs in November 1937. Bell Labs authorized a full research program in late 1938 with Stibitz at the helm. Their *Complex Number Calculator*, completed January 8, 1940, was able to calculate complex numbers. In a demonstration to the American Mathematical Society conference at Dartmouth College on September 11, 1940, Stibitz was able to send the Complex Number Calculator remote commands over telephone lines by a 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, John Mauchly, and Norbert Wiener, who wrote about it in their memoirs.

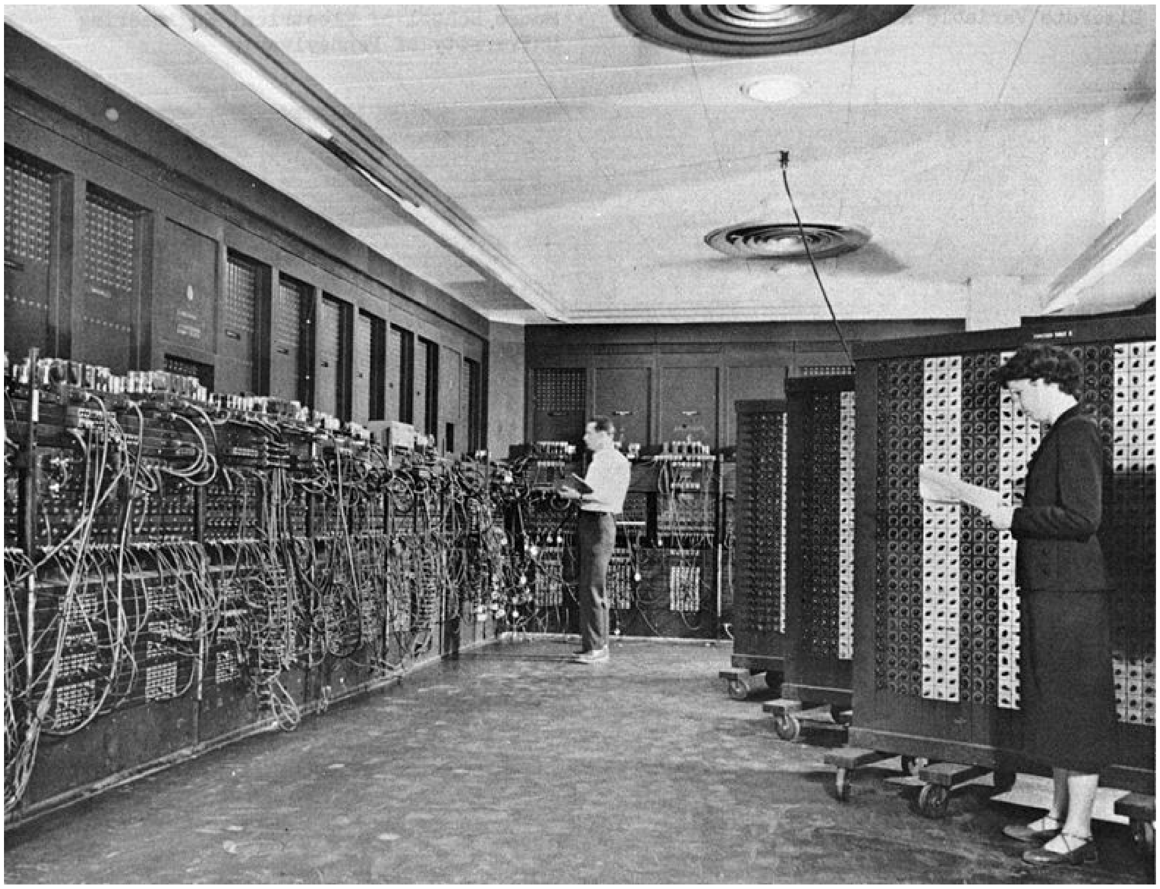


Atanasoff–Berry Computer replica at 1st floor of Durham Center, Iowa State University

In 1939, John Vincent Atanasoff and Clifford E. Berry of Iowa State University developed the Atanasoff–Berry Computer (ABC), The Atanasoff-Berry Computer was the world's first electronic digital computer. 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*, 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. Known officially as the Automatic Sequence Controlled Calculator, 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. 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 and began operation in May 1944.

ENIAC



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. (Colossus couldn't add). 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 and 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, and contained over 18,000 vacuum tubes. 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 electronic machines that evolved from it. Once a program was written, it had to be mechanically set into the machine. Six women did most of the programming of ENIAC. (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).

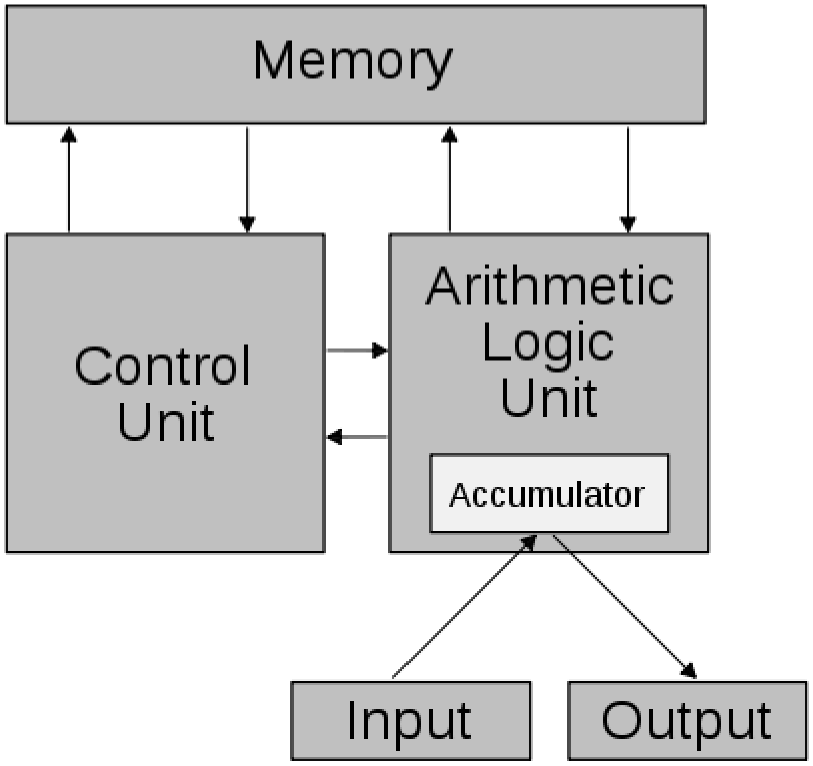
Early computer characteristics

Defining characteristics of some early digital computers of the 1940s (In the **history of computing hardware**)

Name	First operational	Numeral system	Computing mechanism	Programming	Turing complete
Zuse Z3 (Germany)	May 1941	Binary floating point	Electro-mechanical	Program-controlled by punched film stock (but no conditional branch)	Yes (1998)
Atanasoff–Berry Computer (US)	1942	Binary	Electronic	Not programmable—single purpose	No
Colossus Mark 1 (UK)	February 1944	Binary	Electronic	Program-controlled by patch cables and switches	No
Harvard Mark I – IBM ASCC (US)	May 1944	Decimal	Electro-mechanical	Program-controlled by 24-channel punched paper tape (but no conditional branch)	No
Colossus Mark 2 (UK)	June 1944	Binary	Electronic	Program-controlled by patch cables and switches	No
Zuse Z4 (Germany)	March 1945	Binary floating point	Electro-mechanical	Program-controlled by punched film stock	Yes
ENIAC (US)	July 1946	Decimal	Electronic	Program-controlled by patch cables and switches	Yes
Manchester Small-Scale Experimental Machine (Baby) (UK)	June 1948	Binary	Electronic	Stored-program in Williams cathode ray tube memory	Yes
Modified ENIAC (US)	September 1948	Decimal	Electronic	Program-controlled by patch cables and switches plus a primitive read-only stored programming mechanism using the Function Tables as program ROM	Yes

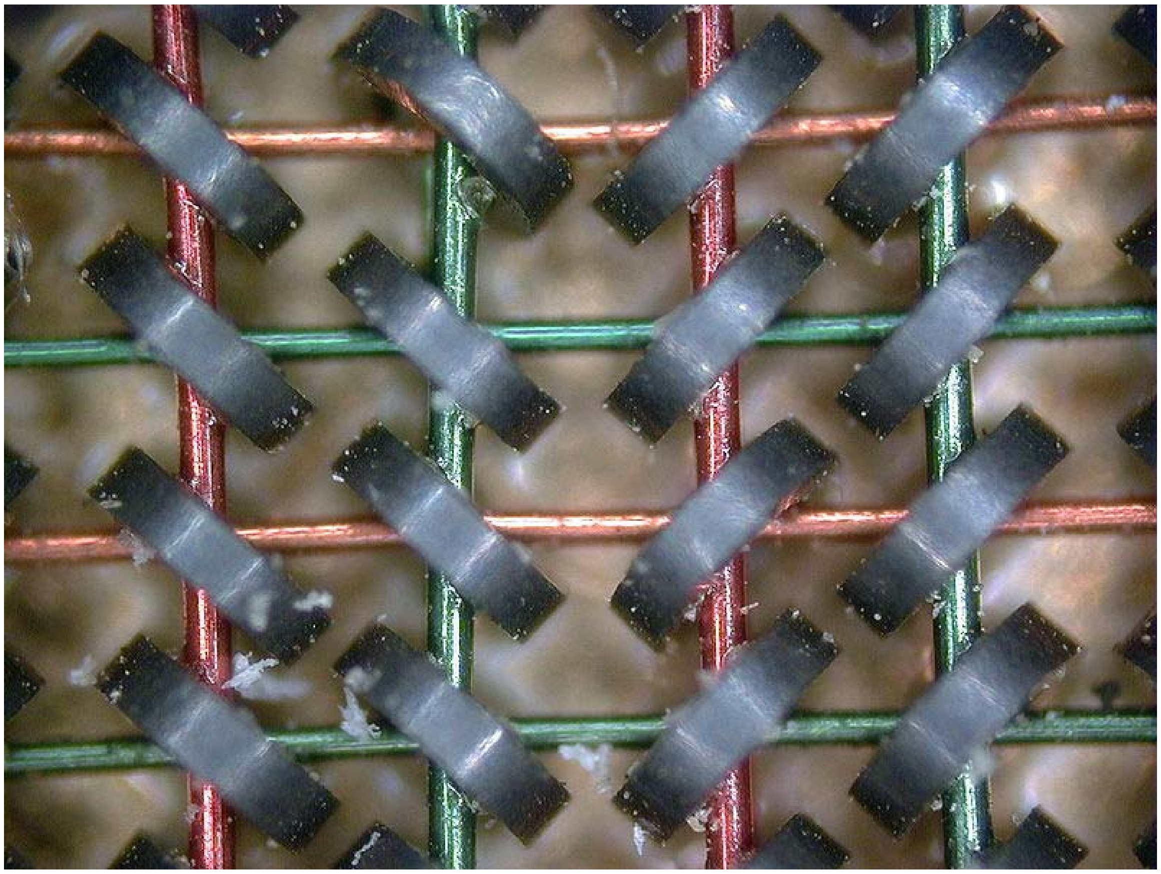
EDSAC (UK)	May 1949	Binary	Electronic	Stored-program in mercury delay line memory	Yes
Manchester Mark 1 (UK)	October 1949	Binary	Electronic	Stored-program in Williams cathode ray tube memory and magnetic drum memory	Yes
CSIRAC (Australia)	November 1949	Binary	Electronic	Stored-program in mercury delay line memory	Yes

First-generation machines



Design of the von Neumann architecture (1947)

Even before the ENIAC was finished, Eckert and Mauchly recognized its limitations and started the design of a stored-program computer, EDVAC. John von Neumann was credited with a widely circulated report describing the 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, would serve as the foundation for the worldwide development of ENIAC's successors. In this generation of equipment, temporary or working storage was provided by acoustic delay lines, which used the propagation time of sound through a medium such as liquid mercury (or through a wire) to briefly store data. A series of acoustic 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, 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 was rapidly displacing most other forms of temporary storage, and dominated the field through the mid-1970s.



Magnetic core memory. Each core is one bit.

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, developed by Frederic C. Williams and Tom Kilburn at the University of Manchester in 1948 as a test bed for the Williams tube; it was followed in 1949 by the Manchester Mark 1 computer, a complete system, using Williams tube and magnetic drum memory, and introducing index registers. The other contender for the title "first digital stored-program computer" had been EDSAC, designed and constructed at the 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. 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 from Kiev Institute of Electrotechnology, Soviet Union (now Ukraine). The computer 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, 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.

Commercial computers

The first commercial computer was the Ferranti Mark 1, which was delivered to the University of Manchester in February 1951. It was based on the Manchester Mark 1. The main improvements over the Manchester Mark 1 were in the size of the primary storage (using random access Williams tubes), secondary storage (using a magnetic drum), 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). A second machine was purchased by the University of Toronto, before the design was revised into the Mark 1 Star. At least seven of the these later machines were delivered between 1953 and 1957, one of them to Shell labs in Amsterdam.

In October 1947, the directors of J. Lyons & Company, 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 computer became operational in April 1951 and ran the world's first regular routine office computer job. 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 to go live on a stored program computer.

In June 1951, the UNIVAC I (Universal Automatic Computer) was delivered to the U.S. Census Bureau. Remington Rand eventually sold 46 machines at more than $1 million each ($8.46 million as of 2010). 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 mercury delay lines capable of storing 1,000 words of 11 decimal 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 media are still used in many computers. In 1952, IBM publicly announced the IBM 701 Electronic

Data Processing Machine, the first in its successful 700/7000 series and its first IBM mainframe computer. The 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, 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 was not implemented at that time.) A volunteer user group, which exists to this day, was founded in 1955 to share their software and experiences with the IBM 701.

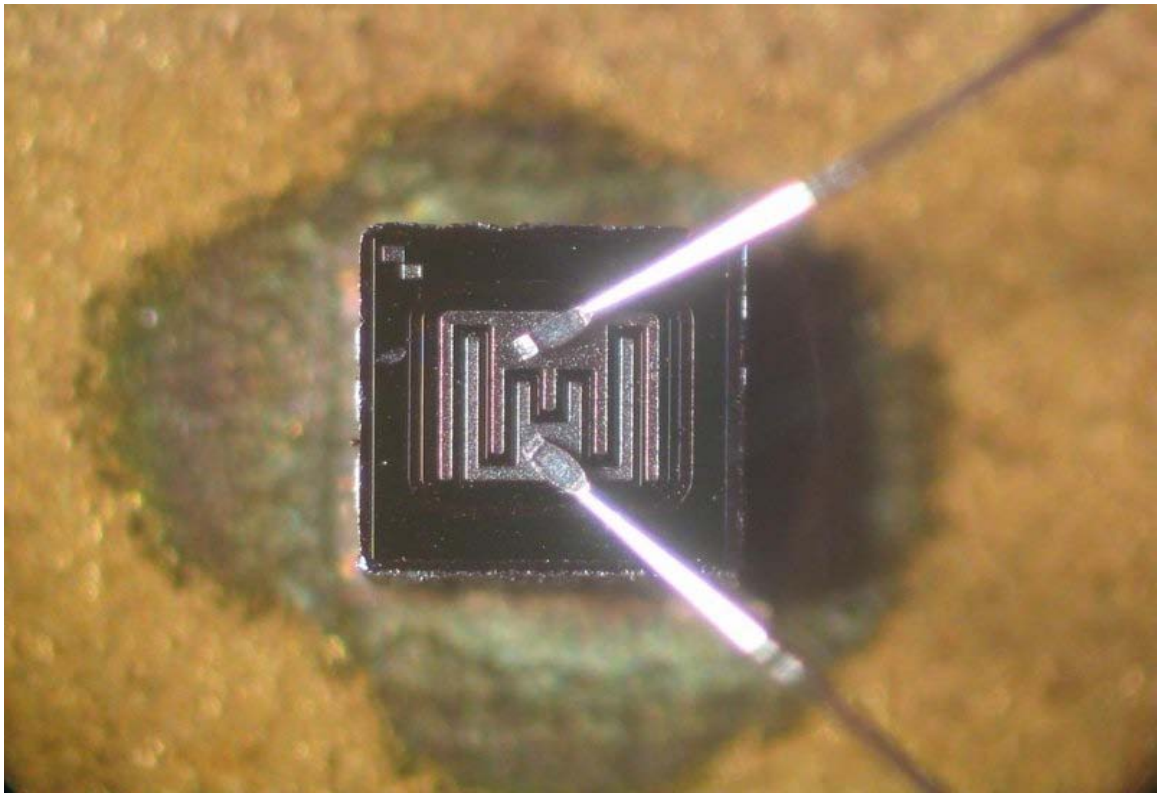
IBM 650 front panel

IBM introduced a smaller, more affordable computer in 1954 that proved very popular. The 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 ($4.09 million as of 2010) or could be leased for $3,500 a month ($30 thousand as of 2010). 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, 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.

In 1955, Maurice Wilkes invented microprogramming, which allows the base instruction set to be defined or extended by built-in programs (now called firmware or microcode). It was widely used in the CPUs and floating-point units of mainframe and other computers, such as the Manchester Atlas and the IBM 360 series.

IBM introduced its first magnetic disk system, RAMAC (Random Access Method of Accounting and Control) in 1956. Using fifty 24-inch (610 mm) metal disks, with 100 tracks per side, it was able to store 5 megabytes of data at a cost of $10,000 per megabyte ($80 thousand as of 2010).

Second generation: transistors

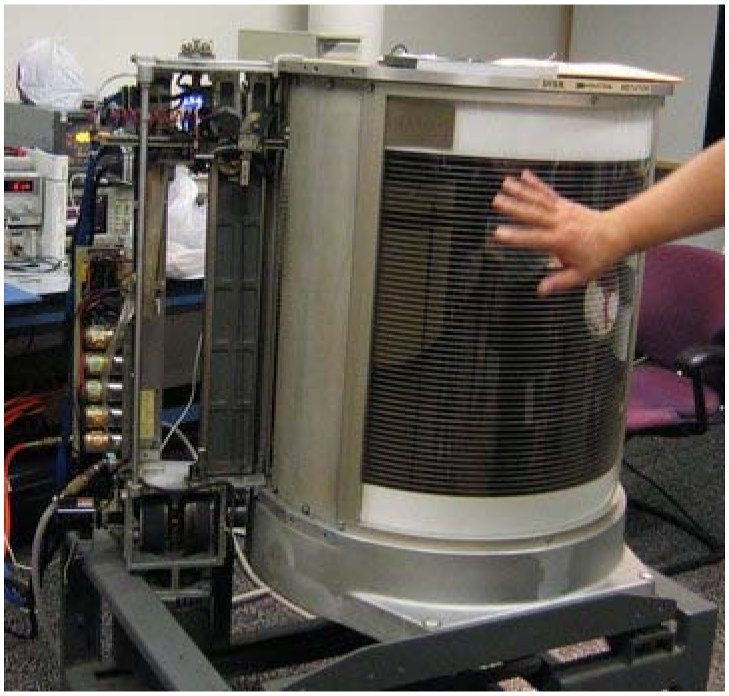


A bipolar junction transistor

The bipolar transistor was invented in 1947. From 1955 onwards transistors replaced vacuum tubes in computer designs, giving rise to the "second generation" of computers. Initially the only devices available were germanium point-contact transistors, which although less reliable than the vacuum tubes they replaced had the advantage of consuming far less power. The first transistorised computer was built at the University of Manchester and was operational by 1953; a second version was completed there in April 1955. The later machine used 200 transistors and 1,300 solid-state diodes 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, whereas the Harwell CADET operated without any valves by using a lower clock frequency, of 58 kHz when it became operational in February 1955. Problems with the reliability of early batches of point contact and alloyed junction transistors meant that the machine's mean time between failures was about 90 minutes, but this improved once the more reliable bipolar junction transistors became available.

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. Typically, second-generation computers were composed of large numbers of printed circuit boards such as the IBM Standard Modular System each carrying one to four logic gates or flip-flops.

A second generation computer, the IBM 1401, captured about one third of the world market. IBM installed more than one hundred thousand 1401s between 1960 and 1964.



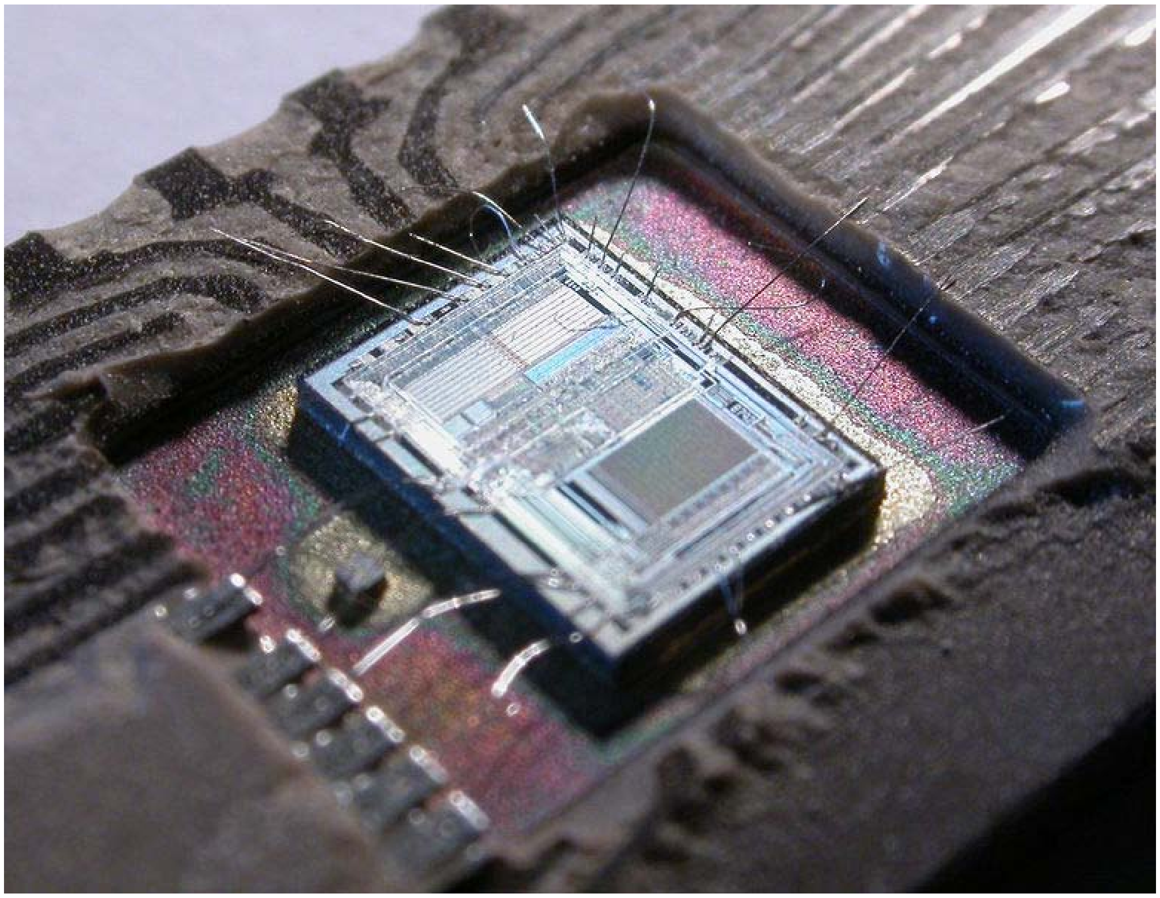
This RAMAC DASD is being restored at the Computer History Museum

Transistorized electronics improved not only the CPU (Central Processing Unit), but also the peripheral devices. The IBM 350 RAMAC was introduced in 1956 and was the world's first disk drive. The second generation disk data storage units were able to store tens of millions of letters and digits. Next to the fixed disk storage units, connected to the CPU via high-speed data transmission, were removable disk data storage units. A removable disk stack can be easily exchanged with another stack 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 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, the main CPU executed calculations and binary branch instructions. One databus would bear data between the main CPU and core memory at the CPU's fetch-execute cycle rate, and other databusses would typically serve the peripheral devices. On the 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 data fetch.

During the second generation remote terminal units (often in the form of teletype machines like a Friden Flexowriter) saw greatly increased use. 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*—the Internet.

Post-1960: third generation and beyond



Intel 8742 eight-bit microcontroller IC

The explosion in the use of computers began with "third-generation" computers, making use of Jack St. Clair Kilby's and Robert Noyce's independent invention of the integrated circuit (or microchip), which later led to the invention of the microprocessor, by Ted

Hoff, Federico Faggin, and Stanley Mazor at Intel. The integrated circuit in the image on the right, for example, an Intel 8742, is an 8-bit microcontroller that includes a CPU running at 12 MHz, 128 bytes of RAM, 2048 bytes of EPROM, and I/O in the same chip.

During the 1960s there was considerable overlap between second and third generation technologies. IBM implemented its IBM Solid Logic Technology modules in hybrid circuits 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 such as the B5000 were stack machines, which allowed for simpler programming. These pushdown automatons 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. It became possible to simulate analog circuits with the *simulation program with integrated circuit emphasis*, or SPICE (1971) on minicomputers, one of the programs for electronic design automation (EDA). The microprocessor led to the development of the 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. Steve Wozniak, co-founder of Apple Computer, is sometimes erroneously credited with developing the first mass-market home computers. However, his first computer, the Apple I, came out some time after the MOS Technology KIM-1 and Altair 8800, and the first Apple computer with graphic and sound capabilities came out well after the 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. 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 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 are the delivery platform. 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.

In the 21st century, multi-core CPUs became commercially available. Content-addressable memory (CAM) 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 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 gate only draws significant current during the 'transition' between logic states, except for leakage.

This has allowed computing to become a commodity which is now ubiquitous, embedded in many forms, from greeting cards and telephones to satellites. Computing hardware and its software have even become a metaphor for the operation of the universe. Although DNA-based computing and quantum qubit computing are years or decades in the future, the infrastructure is being laid today, for example, with DNA origami on photolithography. Fast digital circuits (including those based on Josephson junctions and rapid single flux quantum technology) are becoming more nearly realizable with the discovery of nanoscale superconductors.

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.

An indication of the rapidity of development of this field can be inferred by the history of the seminal article. 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.

Chapter 3

History of Computing Hardware (1960s–Present)

The **history of computing hardware** starting at 1960 is marked by the conversion from vacuum tube to solid state devices such as the transistor and later the integrated circuit. By 1959 discrete transistors were considered sufficiently reliable and economical that they made further vacuum tube computers uncompetitive. Computer main memory slowly moved away from magnetic core memory devices to solid-state static and dynamic semiconductor memory, which greatly reduced the cost, size and power consumption of computer devices. Eventually the cost of integrated circuit devices became low enough that home computers and personal computers became widespread.

Third generation

The mass increase in the use of computers accelerated with 'Third Generation' computers. These generally relied on Jack Kilby's invention of the integrated circuit (or microchip), starting around 1965. However, the IBM System/360 used hybrid circuits, which were solid-state devices interconnected on a substrate with discrete wires.

The first integrated circuit was produced in September 1958 but computers using them didn't begin to appear until 1963. Some of their early uses were in embedded systems, notably used by NASA for the Apollo Guidance Computer and by the military in the LGM-30 Minuteman intercontinental ballistic missile.

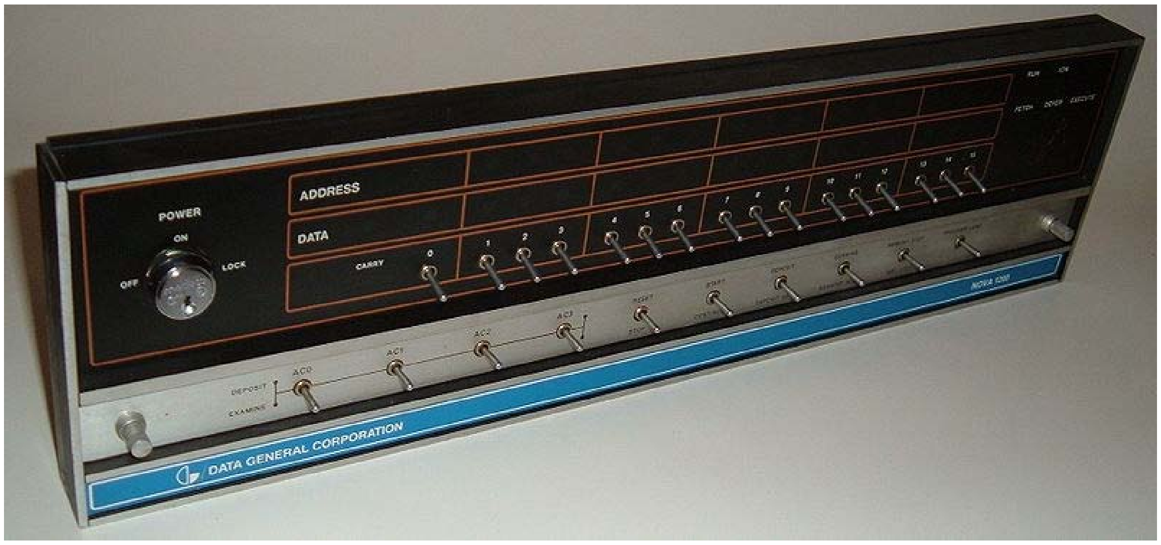
By 1971, the Illiac IV supercomputer, which was the fastest computer in the world for several years, used about a quarter-million small-scale ECL logic gate integrated circuits to make up sixty-four parallel data processors.

While large mainframe computers such as the System/360 increased storage and processing abilities, the integrated circuit also allowed development of much smaller computers. The minicomputer was a significant innovation in the 1960s and 1970s. It brought computing power to more people, not only through more convenient physical size but also through broadening the computer vendor field. Digital Equipment Corporation became the number two computer company behind IBM with their popular

PDP and VAX computer systems. Smaller, affordable hardware also brought about the development of important new operating systems like Unix.

Large-scale integration of circuits led to the development of very small processing units, an early example of this is the processor was the classified CADC used for analyzing flight data in the US Navy's F-14A Tomcat fighter jet. This processor was developed by Steve Geller, Ray Holt and a team from Garrett AiResearch and American Microsystems.

In 1966, Hewlett-Packard entered the general purpose computer business with its HP-2116, offering computing power formerly found only in much larger computers. It supported a wide variety of languages, among them BASIC, ALGOL, and FORTRAN.



1969 Data General Nova

In 1969, Data General shipped a total of 50,000 Novas at $8000 each. The Nova was one of the first 16-bit minicomputers and led the way toward word lengths that were multiples of the 8-bit byte. It was first to employ medium-scale integration (MSI) circuits from Fairchild Semiconductor, with subsequent models using large-scale integrated (LSI) circuits. Also notable was that the entire central processor was contained on one 15-inch printed circuit board.

In 1973, the TV Typewriter, designed by Don Lancaster, provided electronics hobbyists with a display of alphanumeric information on an ordinary television set. It used $120 worth of electronics components, as outlined in the September 1973 issue of Radio Electronics magazine. The original design included two memory boards and could generate and store 512 characters as 16 lines of 32 characters. A 90-minute cassette tape provided supplementary storage for about 100 pages of text. His design used minimalistic hardware to generate the timing of the various signals needed to create the TV signal. Clive Sinclair later used the same approach in his legendary Sinclair ZX80.

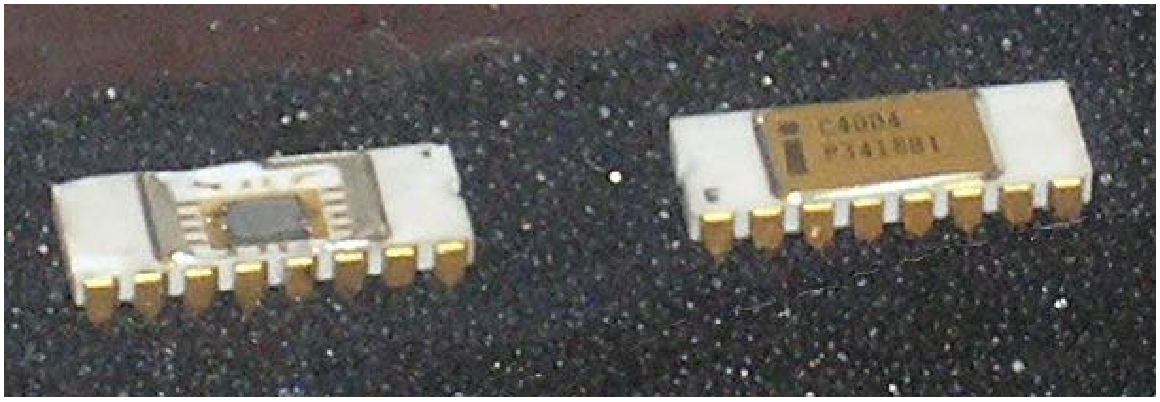
Fourth generation

The basis of the fourth generation was the invention of the microprocessor by a team at Intel.

Unlike third generation minicomputers, which were essentially scaled down versions of mainframe computers, the fourth generation's origins are fundamentally different. Microprocessor-based computers were originally very limited in their computational ability and speed, and were in no way an attempt to downsize the minicomputer. They were addressing an entirely different market.

Although processing power and storage capacities have grown beyond all recognition since the 1970s, the underlying technology of large-scale integration (LSI) or very-large-scale integration (VLSI) microchips has remained basically the same, so it is widely regarded that most of today's computers still belong to the fourth generation.

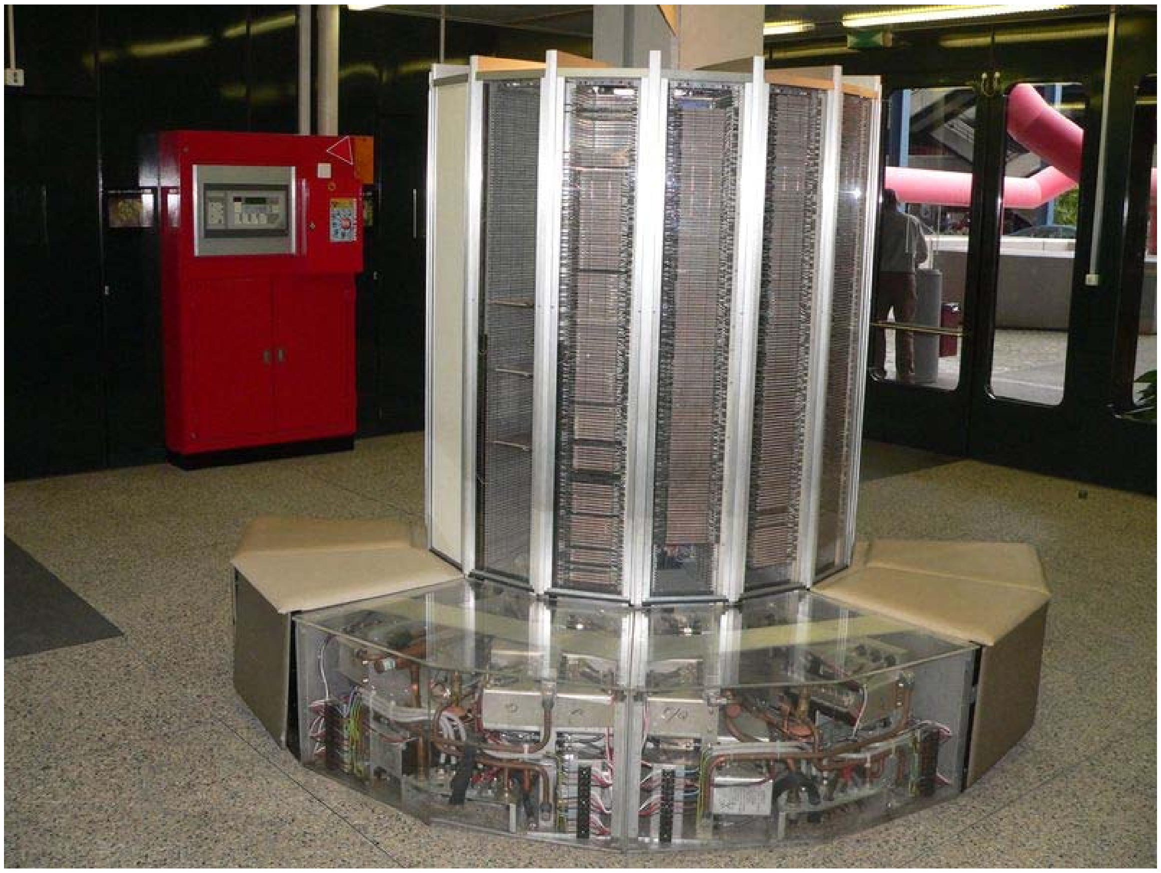
Microprocessors



1971: Intel 4004

On November 15, 1971, Intel released the world's first commercial microprocessor, the 4004. It was developed for a Japanese calculator company, Busicom, as an alternative to hardwired circuitry, but computers were developed around it, with much of their processing abilities provided by one small microprocessor chip. Coupled with one of Intel's other products - the RAM chip, based on an invention by Robert Dennard of IBM, (kilobits of memory on one chip) - the microprocessor allowed fourth generation computers to be smaller and faster than prior computers. The 4004 was only capable of 60,000 instructions per second, but its successors, the Intel 8008, 8080 (used in many computers using the CP/M operating system), and the 8086/8088 family (the IBM personal computer (PC) and compatibles use processors still backwards-compatible with the 8086) brought ever-growing speed and power to the computers. Other producers also made microprocessors which were widely used in microcomputers.

Supercomputers



1976: Cray-1 supercomputer

At the other end of the computing spectrum from the microcomputers, the powerful supercomputers of the era also used integrated circuit technology. In 1976 the Cray-1 was developed by Seymour Cray, who had left Control Data in 1972 to form his own company. This machine, the first supercomputer to make vector processing practical, had a characteristic horseshoe shape, to speed processing by shortening circuit paths. Vector processing, which uses one instruction to perform the same operation on many arguments, has been a fundamental supercomputer processing method ever since. The Cray-1 could calculate 150 million floating point operations per second (150 megaflops). 85 were shipped at a price of $5 million each. The Cray-1 had a CPU that was mostly constructed of SSI and MSI ECL ICs.

Mainframes and minicomputers

Time-sharing computer terminals connected to central computers, such as the TeleVideo ASCII character mode smart terminal pictured here, were sometimes used before the advent of the PC.

Before the introduction of the microprocessor in the early 1970s, computers were generally large, costly systems owned by large institutions: corporations, universities, government agencies, and the like. Users—who were experienced specialists—did not usually interact with the machine itself, but instead prepared tasks for the computer on off-line equipment, such as card punches. A number of assignments for the computer would be gathered up and processed in batch mode. After the jobs had completed, users could collect the output printouts and punched cards. In some organizations it could take hours or days between submitting a job to the computing center and receiving the output.

A more interactive form of computer use developed commercially by the middle 1960s. In a time-sharing system, multiple teletype terminals let many people share the use of one mainframe computer processor. This was common in business applications and in science and engineering.

A different model of computer use was foreshadowed by the way in which early, pre-commercial, experimental computers were used, where one user had exclusive use of a processor. Some of the first computers that might be called "personal" were early minicomputers such as the LINC and PDP-8, and later on VAX and larger minicomputers from Digital Equipment Corporation (DEC), Data General, Prime Computer, and others. They originated as peripheral processors for mainframe computers, taking on some routine tasks and freeing the processor for computation. By today's standards they were physically large (about the size of a refrigerator) and costly (typically tens of thousands of US dollars), and thus were rarely purchased by individuals. However, they were much smaller, less expensive, and generally simpler to operate than the mainframe computers of the time, and thus affordable by individual laboratories and research projects. Minicomputers largely freed these organizations from the batch processing and bureaucracy of a commercial or university computing center.

In addition, minicomputers were more interactive than mainframes, and soon had their own operating systems. The minicomputer Xerox Alto (1973) was a landmark step in the development of personal computers, because of its graphical user interface, bit-mapped high resolution screen, large internal and external memory storage, mouse, and special software.

Microprocessor and cost reduction

The Apple II, one of the "1977 Trinity". The drive shown is a model made for the Apple III.

The minicomputer ancestors of the modern personal computer used integrated circuit technology, which reduced size and cost compared to discrete transistors. Processing was carried out by circuits with large numbers of components arranged on multiple large printed circuit boards. Minicomputers were consequently physically large and expensive to produce compared with later microprocessor systems. After the "computer-on-a-chip" was commercialized, the cost to produce a computer system dropped dramatically. The arithmetic, logic, and control functions that previously occupied several costly circuit boards were now available in one integrated circuit which was very expensive to design but cheap to produce in large quantities. Concurrently, advances in developing solid state memory eliminated the bulky, costly, and power-hungry magnetic core memory used in prior generations of computers.

Altair 8800 and IMSAI 8080

Development of the single-chip microprocessor was an enormous catalyst to the popularization of cheap, easy to use, and truly personal computers. The Altair 8800, introduced in a *Popular Electronics* magazine article in the January 1975 issue, at the time set a new low price point for a computer, bringing computer ownership to an admittedly select market in the 1970s. This was followed by the IMSAI 8080 computer, with similar abilities and limitations. The Altair and IMSAI were essentially scaled-down minicomputers and were incomplete: to connect a keyboard or teletype to them required heavy, expensive "peripherals". These machines both featured a front panel with switches and lights, which communicated with the operator in binary. To program the machine after switching it on the bootstrap loader program had to be entered, without error, in binary, then a paper tape containing a BASIC interpreter loaded from a paper-tape reader. Keying the loader required setting a bank of eight switches up or down and pressing the "load" button, once for each byte of the program, which was typically hundreds of bytes long. The computer could run BASIC programs once the interpreter had been loaded.

1975: Altair 8800

The MITS Altair, the first commercially successful microprocessor kit, was featured on the cover of *Popular Electronics* magazine in January 1975. It was the world's first mass-produced personal computer kit, as well as the first computer to use an Intel 8080 processor. It was a commercial success with 10,000 Altairs being shipped. The Altair also inspired the software development efforts of Paul Allen and his high school friend Bill Gates who developed a BASIC interpreter for the Altair, and then formed Microsoft.

The MITS Altair 8800 effectively created a new industry of microcomputers and computer kits, with many others following, such as a wave of small business computers in the late 1970s based on the Intel 8080, Zilog Z80 and Intel 8085 microprocessor chips. Most ran the CP/M-80 operating system developed by Gary Kildall at Digital Research. CP/M-80 was the first popular microcomputer operating system to be used by many different hardware vendors, and many software packages were written for it, such as WordStar and dBase II.

Many hobbyists during the mid 1970s designed their own systems, with various degrees of success, and sometimes banded together to ease the job. Out of these house meetings the Homebrew Computer Club developed, where hobbyists met to talk about what they had done, exchange schematics and software, and demonstrate their systems. Many people built or assembled their own computers as per published designs. For example, many thousands of people built the Galaksija home computer later in the early 80s.

It was arguably the Altair computer that spawned the development of Apple, as well as Microsoft which produced and sold the Altair BASIC programming language interpreter, Microsoft's first product. The second generation of microcomputers, those that appeared in the late 1970s, sparked by the unexpected demand for the kit computers at the electronic hobbyist clubs, were usually known as home computers. For business use these systems were less capable and in some ways less versatile than the large business computers of the day. They were designed for fun and educational purposes, not so much for practical use. And although you could use some simple office/productivity applications on them, they were generally used by computer enthusiasts for learning to program and for running computer games, for which the personal computers of the period were less suitable and much too expensive. For the more technical hobbyists home computers were also used for electronics interfacing, such as controlling model railroads, and other general hobbyist pursuits.

Micral N

In France, the company R2E (Réalisations et Etudes Electroniques) formed by two former engineers of the Intertechnique company, André Truong Trong Thi and François Gernelle introduced in February 1973 a microcomputer, the Micral N based on the Intel 8008. Originally, the computer had been designed by Gernelle, Lacombe, Beckmann and Benchitrite for the Institut National de la Recherche Agronomique to automate hygrometric measurements. The Micral N cost a fifth of the price of a PDP-8, about 8500FF ($1300). The clock of the Intel 8008 was set at 500kHz, the memory was 16 kilobytes. A bus, called Pluribus was introduced and allowed connection of up to 14 boards. Different boards for digital I/O, analog I/O, memory, floppy disk were available from R2E. The Micral operating system was initially called Sysmic, and was later renamed Prologue. R2E was absorbed by Groupe Bull in 1978. Although Groupe Bull continued the production of Micral computers, it was not interested in the Personal Computer market. and Micral computers were mostly confined to highway toll gates (where they remained in service until 1992) and similar niche markets.

Microcomputer emerges

The advent of the microprocessor and solid-state memory made home computing affordable. Early hobby microcomputer systems such as the Altair 8800 and Apple I introduced around 1975 marked the release of low-cost 8-bit processor chips, which had sufficient computing power to be of interest to hobby and experimental users. By 1977 pre-assembled systems such as the Apple II, Commodore PET, and TRS-80 (later dubbed the "1977 Trinity" by *Byte* Magazine) began the era of mass-market home computers; much less effort was required to obtain an operating computer, and applications such as games, word processing, and spreadsheets began to proliferate. Distinct from computers used in homes, small business systems were typically based on CP/M, until IBM introduced the IBM-PC, which was quickly adopted. The PC was heavily cloned, leading to mass production and consequent cost reduction throughout the 1980s. This expanded the PCs presence in homes, replacing the home computer category during the 1990s and leading to the current monoculture of architecturally identical personal computers.

Chapter 4

History of Artificial Intelligence and History of Computer Science

History of artificial intelligence

The **history of artificial intelligence** began in antiquity, with myths, stories and rumors of artificial beings endowed with intelligence or consciousness by master craftsmen; as Pamela McCorduck writes, AI began with "an ancient wish to forge the gods."

The seeds of modern AI were planted by classical philosophers who attempted to describe the process of human thinking as the mechanical manipulation of symbols. This work culminated in the invention of the programmable digital computer in the 1940s, a machine based on the abstract essence of mathematical reasoning. This device and the ideas behind it inspired a handful of scientists to begin seriously discussing the possibility of building an electronic brain.

The field of artificial intelligence research was founded at a conference on the campus of Dartmouth College in the summer of 1956. Those who attended would become the leaders of AI research for decades. Many of them predicted that a machine as intelligent as a human being would exist in no more than a generation and they were given millions of dollars to make this vision come true.

Eventually it became obvious that they had grossly underestimated the difficulty of the project. In 1973, in response to the criticism of Sir James Lighthill and ongoing pressure from congress, the U.S. and British Governments stopped funding undirected research into artificial intelligence. Seven years later, a visionary initiative by the Japanese Government inspired governments and industry to provide AI with billions of dollars, but by the late 80s the investors became disillusioned and withdrew funding again. This cycle of boom and bust, of "AI winters" and summers, continues to haunt the field. Undaunted, there are those who make extraordinary predictions even now.

Progress in AI has continued, despite the rise and fall of its reputation in the eyes of government bureaucrats and venture capitalists. Problems that had begun to seem impossible in 1970 have been solved and the solutions are now used in successful commercial products. However, no machine has been built with a human level of intelligence, contrary to the optimistic predictions of the first generation of AI researchers. "We can only see a short distance ahead," admitted Alan Turing, in a famous 1950 paper that catalyzed the modern search for machines that think. "But," he added, "we can see much that must be done."

Precursors

McCorduck (2004) writes "artificial intelligence in one form or another is an idea that has pervaded Western intellectual history, a dream in urgent need of being realized," expressed in humanity's myths, legends, stories, speculation and clockwork automatons.

AI in myth, fiction and speculation

Mechanical men and artificial beings appear in Greek myths, such as the golden robots of Hephaestus and Pygmalion's Galatea. In the Middle Ages, there were rumors of secret mystical or alchemical means of placing mind into matter, such as Jābir ibn Hayyān's *Takwin*, Paracelsus' homunculus and Rabbi Judah Loew's Golem. By the 19th century, ideas about artificial men and thinking machines were developed in fiction, as in Mary Shelley's *Frankenstein* or Karel Čapek's *R.U.R. (Rossum's Universal Robots)*, and speculation, such as Samuel Butler's "Darwin among the Machines." AI has continued to be an important element of science fiction into the present.

Automatons

Al-Jazari's programmable automata (1206 CE)

Realistic humanoid automatons were built by craftsman from every civilization, including Yan Shi, Hero of Alexandria, Al-Jazari and Wolfgang von Kempelen. The oldest known automatons were the sacred statues of ancient Egypt and Greece. The faithful believed that craftsman had imbued these figures with very real minds, capable of wisdom and emotion—Hermes Trismegistus wrote that "by discovering the true nature of the gods, man has been able to reproduce it."

Formal reasoning

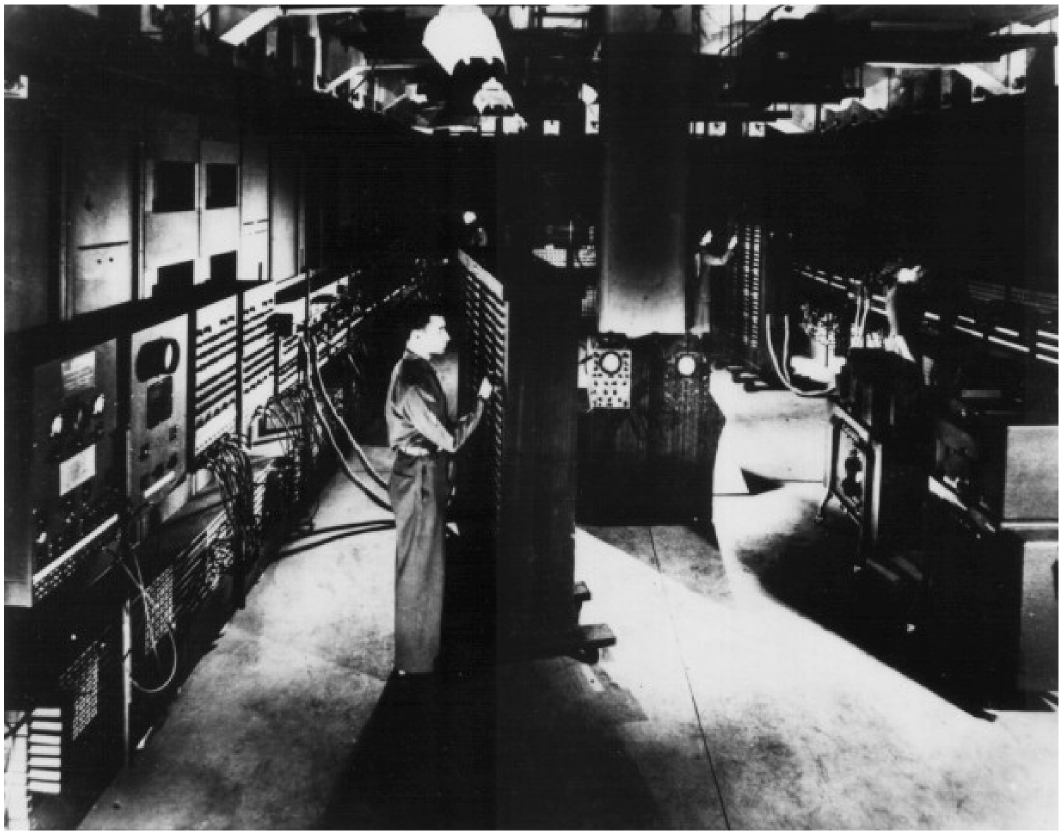
Artificial intelligence is based on the assumption that the process of human thought can be mechanized. The study of mechanical—or "formal"—reasoning has a long history. Chinese, Indian and Greek philosophers all developed structured methods of formal deduction in the first millennium BCE. Their ideas were developed over the centuries by philosophers such as Aristotle (who gave a formal analysis of the syllogism), Euclid (whose *Elements* was a model of formal reasoning), al-Khwārizmī (who developed algebra and gave his name to "algorithm") and European scholastic philosophers such as William of Ockham and Duns Scotus.

Majorcan philosopher Ramon Llull (1232–1315) developed several *logical machines* devoted to the production of knowledge by logical means; Llull described his machines as mechanical entities that could combine basic and undeniable truths by simple logical operations, produced by the machine by mechanical meanings, in such ways as to produce all the possible knowledge. Llull's work had a great influence on Gottfried Leibniz, who redeveloped his ideas.

Gottfried Leibniz, who speculated that human reason could be reduced to mechanical calculation

In the 17th century, Leibniz, Thomas Hobbes and René Descartes explored the possibility that all rational thought could be made as systematic as algebra or geometry. Hobbes famously wrote in *Leviathan*: "reason is nothing but reckoning". Leibniz envisioned a universal language of reasoning (his *characteristica universalis*) which would reduce argumentation to calculation, so that "there would be no more need of disputation between two philosophers than between two accountants. For it would suffice to take their pencils in hand, down to their slates, and to say each other (with a friend as witness, if they liked): *Let us calculate*." These philosophers had begun to articulate the physical symbol system hypothesis that would become the guiding faith of AI research.

In the 20th century, the study of mathematical logic provided the essential breakthrough that made artificial intelligence seem plausible. The foundations had been set by such works as Boole's *The Laws of Thought* and Frege's *Begriffsschrift*. Building on Frege's system, Russell and Whitehead presented a formal treatment of the foundations of mathematics in their masterpiece, the *Principia Mathematica* in 1913. Inspired by Russell's success, David Hilbert challenged mathematicians of the 1920s and 30s to answer this fundamental question: "can all of mathematical reasoning be formalized?" His question was answered by Gödel's incompleteness proof, Turing's machine and Church's Lambda calculus. Their answer was surprising in two ways. First, they proved that there were, in fact, limits to what mathematical logic could accomplish.



The ENIAC, at the Moore School of Electrical Engineering

But second (and more important for AI) their work suggested that, within these limits, *any* form of mathematical reasoning could be mechanized. The Church-Turing thesis implied that a mechanical device, shuffling symbols as simple as 0 and 1, could imitate any conceivable process of mathematical deduction. The key insight was the Turing machine—a simple theoretical construct that captured the essence of abstract symbol manipulation. This invention would inspire a handful of scientists to begin discussing the possibility of thinking machines.

The birth of artificial intelligence 1943−1956

The IBM 702: a computer used by the first generation of AI researchers

In the 1940s and 50s, a handful of scientists from a variety fields (mathematics, psychology, engineering, economics and political science) began to discuss the possibility of creating an artificial brain. The field of artificial intelligence research was founded as an academic discipline in 1956.

Cybernetics and early neural networks

The earliest research into thinking machines was inspired by a confluence of ideas that became prevalent in the late 30s, 40s and early 50s. Recent research in neurology had shown that the brain was an electrical network of neurons that fired in all-or-nothing pulses. Norbert Wiener's cybernetics described control and stability in electrical networks. Claude Shannon's information theory described digital signals (i.e., all-or-nothing signals). Alan Turing's theory of computation showed any form computation could be described digitally. The close relationship between these ideas suggested that it might be possible to construct an electronic brain.

Examples of work in this vein includes robots such as W. Grey Walter's turtles and the Johns Hopkins Beast. These machines did not use computers, digital electronics or symbolic reasoning; they were controlled entirely by analog circuitry.

Walter Pitts and Warren McCulloch analyzed networks of idealized artificial neurons and showed how they might perform simple logical functions. They were the first to describe what later researchers would call a neural network. One of the students inspired by Pitts and McCulloch was a young Marvin Minsky, then a 24 year old graduate student. In 1951 (with Dean Edmonds) he built the first neural net machine, the SNARC. Minsky was to become one of the most important leaders and innovators in AI for the next 50 years.

Game AI

In 1951, using the Ferranti Mark 1 machine of the University of Manchester, Christopher Strachey wrote a checkers program and Dietrich Prinz wrote one for chess. Arthur Samuel's checkers program, developed in the middle 50s and early 60s, eventually achieved sufficient skill to challenge a respectable amateur. Game AI would continue to be used as a measure of progress in AI throughout its history.

Turing's test

In 1950 Alan Turing published a landmark paper in which he speculated about the possibility of creating machines with true intelligence. He noted that "intelligence" is difficult to define and devised his famous Turing Test. If a machine could carry on a conversation (over a teletype) that was indistinguishable from a conversation with a human being, then the machine could be called "intelligent." This simplified version of the problem allowed Turing to argue convincingly that a "thinking machine" was at least *plausible* and the paper answered all the most common objections to the proposition. The Turing Test was the first serious proposal in the philosophy of artificial intelligence.

Symbolic reasoning and the Logic Theorist

When access to digital computers became possible in the middle fifties, a few scientists instinctively recognized that a machine that could manipulate numbers could also manipulate symbols and that the manipulation of symbols could well be the essence of human thought. This was a new approach to creating thinking machines.

In 1955, Allen Newell and (future Nobel Laureate) Herbert Simon created the "Logic Theorist" (with help from J. C. Shaw). The program would eventually prove 38 of the first 52 theorems in Russell and Whitehead's *Principia Mathematica*, and find new and more elegant proofs for some. Simon said that they had "solved the venerable mind/body problem, explaining how a system composed of matter can have the properties of mind." (This was an early statement of the philosophical position John Searle would later call "Strong AI": that machines can contain minds just as human bodies do.)

Dartmouth Conference 1956: the birth of AI

The Dartmouth Conference of 1956 was organized by Marvin Minsky, John McCarthy and two senior scientists: Claude Shannon and Nathan Rochester of IBM. The proposal for the conference included this assertion: "every aspect of learning or any other feature of intelligence can be so precisely described that a machine can be made to simulate it". The participants included Ray Solomonoff, Oliver Selfridge, Trenchard More, Arthur

Samuel, Allen Newell and Herbert Simon, all of whom would create important programs during the first decades of AI research. At the conference Newell and Simon debuted the "Logic Theorist" and McCarthy persuaded the attendees to accept "Artificial Intelligence" as the name of the field. The 1956 Dartmouth conference was the moment that AI gained its name, its mission, its first success and its major players, and is widely considered the birth of AI.

The golden years 1956−1974

The years after the Dartmouth conference were an era of discovery, of sprinting across new ground. The programs that were developed during this time were, to most people, simply "astonishing": computers were solving algebra word problems, proving theorems in geometry and learning to speak English. Few at the time would have believed that such "intelligent" behavior by machines was possible at all. Researchers expressed an intense optimism in private and in print, predicting that a fully intelligent machine would be built in less than 20 years. Government agencies like ARPA poured money into the new field.

The work

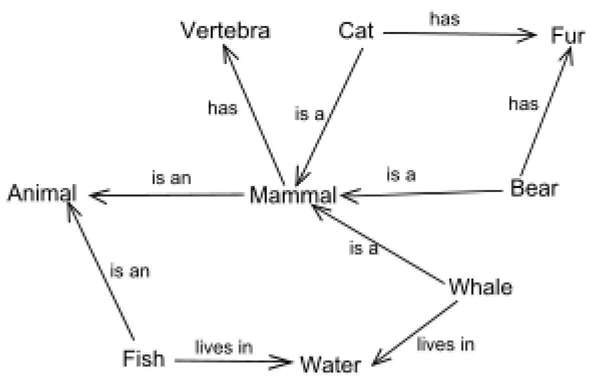
There were many successful programs and new directions in the late 50s and 1960s. Among the most influential were these:

Reasoning as search

Many early AI programs used the same basic algorithm. To achieve some goal (like winning a game or proving a theorem), they proceeded step by step towards it (by making a move or a deduction) as if searching through a maze, backtracking whenever they reached a dead end. This paradigm was called "reasoning as search".

The principal difficulty was that, for many problems, the number of possible paths through the "maze" was simply astronomical (a situation known as a "combinatorial explosion"). Researchers would reduce the search space by using heuristics or "rules of thumb" that would eliminate those paths that were unlikely to lead to a solution.

Newell and Simon tried to capture a general version of this algorithm in a program called the "General Problem Solver". Other "searching" programs were able to accomplish impressive tasks like solving problems in geometry and algebra, such as Herbert Gelernter's Geometry Theorem Prover (1958) and SAINT, written by Minsky's student James Slagle (1961). Other programs searched through goals and subgoals to plan actions, like the STRIPS system developed at Stanford to control the behavior of their robot Shakey.



An example of a semantic network

Natural language

An important goal of AI research is to allow computers to communicate in natural languages like English. An early success was Daniel Bobrow's program STUDENT, which could solve high school algebra word problems.

A semantic net represents concepts (e.g. "house","door") as nodes and relations among concepts (e.g. "has-a") as links between the nodes. The first AI program to use a semantic net was written by Ross Quillian and the most successful (and controversial) version was Roger Schank's Conceptual Dependency.

Perhaps the most interesting English speaking computer program was Joseph Weizenbaum's ELIZA, the first chatterbot. ELIZA could carry out conversations that were so realistic that users occasionally were fooled into thinking they were communicating with a human being and not a program. But in fact, ELIZA had no idea what she was talking about. She simply gave a canned response or repeated back what was said to her, rephrasing her response with a few grammar rules.

Micro-worlds

In the late 60s, Marvin Minsky and Seymour Papert of the MIT AI Laboratory proposed that AI research should focus on artificially simple situations known as micro-worlds. They pointed out that in successful sciences like physics, basic principles were often best understood using simplified models like frictionless planes or perfectly rigid bodies. Much of the research focused on a "blocks world," which consists of colored blocks of various shapes and sizes arrayed on a flat surface.

This paradigm led to innovative work in machine vision by Gerald Sussman (who led the team), Adolfo Guzman, David Waltz (who invented "constraint propagation"), and especially Patrick Winston. At the same time, Minsky and Papert built a robot arm that could stack blocks, bringing the blocks world to life. The crowning achievement of the micro-world program was Terry Winograd's SHRDLU. It could communicate in ordinary English sentences, plan operations and execute them.

The optimism

The first generation of AI researchers made these predictions about their work:

- 1958, H. A. Simon and Allen Newell: "within ten years a digital computer will be the world's chess champion" and "within ten years a digital computer will discover and prove an important new mathematical theorem."
- 1965, H. A. Simon: "machines will be capable, within twenty years, of doing any work a man can do."
- 1967, Marvin Minsky: "Within a generation ... the problem of creating 'artificial intelligence' will substantially be solved."
- 1970, Marvin Minsky (in *Life* Magazine): "In from three to eight years we will have a machine with the general intelligence of an average human being."

The money

In June 1963 MIT received a $2.2 million grant from the newly created Advanced Research Projects Agency (later known as DARPA). The money was used to fund project MAC which subsumed the "AI Group" founded by Minsky and McCarthy five years earlier. ARPA continued to provide three million dollars a year until the 70s. ARPA made similar grants to Newell and Simon's program at CMU and to the Stanford AI Project (founded by John McCarthy in 1963). Another important AI laboratory was established at Edinburgh University by Donald Michie in 1965. These four institutions would continue to be the main centers of AI research (and funding) in academia for many years.

The money was proffered with few strings attached: J. C. R. Licklider, then the director of ARPA, believed that his organization should "fund people, not projects!" and allowed researchers to pursue whatever directions might interest them. This created a freewheeling atmosphere at MIT that gave birth to the hacker culture, but this "hands off" approach would not last.

The first AI winter 1974−1980

In the 70s, AI was subject to critiques and financial setbacks. AI researchers had failed to appreciate the difficulty of the problems they faced. Their tremendous optimism had raised expectations impossibly high, and when the promised results failed to materialize, funding for AI disappeared. At the same time, the field of connectionism (or neural nets) was shut down almost completely for 10 years by Marvin Minsky's devastating criticism of perceptrons. Despite the difficulties with public perception of AI in the late 70s, new ideas were explored in logic programming, commonsense reasoning and many other areas.

The problems

In the early seventies, the capabilities of AI programs were limited. Even the most impressive could only handle trivial versions of the problems they were supposed to solve; all the programs were, in some sense, "toys". AI researchers had begun to run into several fundamental limits that could not be overcome in the 1970s. Although some of these limits would be conquered in later decades, others still stymie the field to this day.

- **Limited computer power**: There was not enough memory or processing speed to accomplish anything truly useful. For example, Ross Quillian's successful work on natural language was demonstrated with a vocabulary of only *twenty* words, because that was all that would fit in memory. Hans Moravec argued in 1976 that computers were still millions of times too weak to exhibit intelligence. He suggested an analogy: artificial intelligence requires computer power in the same way that aircraft require power. Below a certain threshold, it's impossible, but, as power increases, eventually it could become easy.
- **Intractability and the combinatorial explosion**. In 1972 Richard Karp (building on Stephen Cook's 1971 theorem) showed there are many problems that can probably only be solved in exponential time (in the size of the inputs). Finding optimal solutions to these problems requires unimaginable amounts of computer time except when the problems are trivial. This almost certainly meant that many of the "toy" solutions used by AI would probably never scale up into useful systems.
- **Commonsense knowledge and reasoning**. Many important artificial intelligence applications like vision or natural language require simply enormous amounts of information about the world: the program needs to have some idea of what it might be looking at or what it is talking about. This requires that the program know most of the same things about the world that a child does. Researchers soon discovered that this was a truly *vast* amount of information. No one in 1970 could build a database so large and no one knew how a program might learn so much information.
- **Moravec's paradox**: Proving theorems and solving geometry problems is comparatively easy for computers, but a supposedly simple task like recognizing a face or crossing a room without bumping into anything is extremely difficult. This helps explain why research into vision and robotics had made so little progress by the middle 1970s.
- **The frame and qualification problems**. AI researchers (like John McCarthy) who used logic discovered that they could not represent ordinary deductions that involved planning or default reasoning without making changes to the structure of logic itself. They developed new logics (like non-monotonic logics and modal logics) to try to solve the problems.

The end of funding

The agencies which funded AI research (such as the British government, DARPA and NRC) became frustrated with the lack of progress and eventually cut off almost all funding for undirected research into AI. The pattern began as early as 1966 when the ALPAC report appeared criticizing machine translation efforts. After spending 20 million dollars, the NRC ended all support. In 1973, the Lighthill report on the state of AI research in England criticized the utter failure of AI to achieve its "grandiose objectives" and led to the dismantling of AI research in that country. (The report specifically mentioned the combinatorial explosion problem as a reason for AI's failings.) DARPA was deeply disappointed with researchers working on the Speech Understanding Research program at CMU and canceled an annual grant of three million dollars. By 1974, funding for AI projects was hard to find.

Hans Moravec blamed the crisis on the unrealistic predictions of his colleagues. "Many researchers were caught up in a web of increasing exaggeration." However, there was another issue: since the passage of the Mansfield Amendment in 1969, DARPA had been under increasing pressure to fund "mission-oriented direct research, rather than basic undirected research". Funding for the creative, freewheeling exploration that had gone on in the 60s would not come from DARPA. Instead, the money was directed at specific projects with clear objectives, such as autonomous tanks and battle management systems.

Critiques from across campus

Several philosophers had strong objections to the claims being made by AI researchers. One of the earliest was John Lucas, who argued that Gödel's incompleteness theorem showed that a formal system (such as a computer program) could never see the truth of certain statements, while a human being could. Hubert Dreyfus ridiculed the broken promises of the 60s and critiqued the assumptions of AI, arguing that human reasoning actually involved very little "symbol processing" and a great deal of embodied, instinctive, unconscious "know how". John Searle's Chinese Room argument, presented in 1980, attempted to show that a program could not be said to "understand" the symbols that it uses (a quality called "intentionality"). If the symbols have no meaning for the machine, Searle argued, then the machine can not be described as "thinking".

These critiques were not taken seriously by AI researchers, often because they seemed so far off the point. Problems like intractability and commonsense knowledge seemed much more immediate and serious. It was unclear what difference "know how" or "intentionality" made to an actual computer program. Minsky said of Dreyfus and Searle "they misunderstand, and should be ignored." Dreyfus, who taught at MIT, was given a cold shoulder: he later said that AI researchers "dared not be seen having lunch with me." Joseph Weizenbaum, the author of ELIZA, felt his colleagues' treatment of Dreyfus was unprofessional and childish. Although he was an outspoken critic of Dreyfus' positions, he "deliberately made it plain that theirs was not the way to treat a human being."

Weizenbaum began to have serious ethical doubts about AI when Kenneth Colby wrote DOCTOR, a chatterbot therapist. Weizenbaum was disturbed that Colby saw his mindless program as a serious therapeutic tool. A feud began, and the situation was not helped when Colby did not credit Weizenbaum for his contribution to the program. In

1976, Weizenbaum published *Computer Power and Human Reason* which argued that the misuse of artificial intelligence has the potential to devalue human life.

Perceptrons and the dark age of connectionism

A perceptron was a form of neural network introduced in 1958 by Frank Rosenblatt, who had been a schoolmate of Marvin Minsky at the Bronx High School of Science. Like most AI researchers, he was optimistic about their power, predicting that "perceptron may eventually be able to learn, make decisions, and translate languages." An active research program into the paradigm was carried out throughout the 60s but came to a sudden halt with the publication of Minsky and Papert's 1969 book *Perceptrons*. It suggested that there were severe limitations to what perceptrons could do and that Frank Rosenblatt's predictions had been grossly exaggerated. The effect of the book was devastating: virtually no research at all was done in connectionism for 10 years. Eventually, a new generation of researchers would revive the field and thereafter it would become a vital and useful part of artificial intelligence. Rosenblatt would not live to see this, as he died in a boating accident shortly after the book was published.

The neats: logic, Prolog and expert systems

Logic was introduced into AI research as early as 1958, by John McCarthy in his Advice Taker proposal. In 1963, J. Alan Robinson had discovered a simple method to implement deduction on computers, the resolution and unification algorithm. However, straightforward implementations, like those attempted by McCarthy and his students in the late 60s, were especially intractable: the programs required astronomical numbers of steps to prove simple theorems. A more fruitful approach to logic was developed in the 70s by Robert Kowalski at the University of Edinburgh, and soon this led to the collaboration with French researchers Alain Colmerauer and Phillipe Roussel who created the successful logic programming language Prolog. Prolog uses a subset of logic (Horn clauses, closely related to "rules" and "production rules") that permit tractable computation. Rules would continue to be influential, providing a foundation for Edward Feigenbaum's expert systems and the continuing work by Alan Newell and Herbert Simon that would lead to Soar and their unified theories of cognition.

Critics of the logical approach noted, as Dreyfus had, that human beings rarely used logic when they solved problems. Experiments by psychologists like Peter Wason, Eleanor Rosch, Amos Tversky, Daniel Kahneman and others provided proof. McCarthy responded that what people do is irrelevant. He argued that what is really needed are machines that can solve problems—not machines that think as people do.

The scruffies: frames and scripts

Among the critics of McCarthy's approach were his colleagues across the country at MIT. Marvin Minsky, Seymour Papert and Roger Schank were trying to solve problems like "story understanding" and "object recognition" that *required* a machine to think like a person. In order to use ordinary concepts like "chair" or "restaurant" they had to make all the same illogical assumptions that people normally made. Unfortunately, imprecise concepts like these are hard to represent in logic. Gerald Sussman observed that "using precise language to describe essentially imprecise concepts doesn't make them any more precise." Schank described their "anti-logic" approaches as "scruffy", as opposed to the "neat" paradigms used by McCarthy, Kowalski, Feigenbaum, Newell and Simon.

In 1975, in a seminal paper, Minsky noted that many of his fellow "scruffy" researchers were using the same kind of tool: a framework that captures all our common sense assumptions about something. For example, if we use the concept of a bird, there is a constellation of facts that immediately come to mind: we might assume that it flies, eats worms and so on. We know these facts are not always true and that deductions using these facts will not be "logical," but these structured sets of assumptions are part of the *context* of everything we say and think. He called these structures "frames". Schank used a version of frames he called "scripts" to successfully answer questions about short stories in English. Many years later object-oriented programming would adopt the essential idea of "inheritance" from AI research on frames.

Boom 1980–1987

In the 1980s a form of AI program called "expert systems" was adopted by corporations around the world and knowledge became the focus of mainstream AI research. In those same years, the Japanese government aggressively funded AI with its fifth generation computer project. Another encouraging event in the early 1980s was the revival of connectionism in the work of John Hopfield and David Rumelhart. Once again, AI had achieved success.

The rise of expert systems

An expert system is a program that answers questions or solves problems about a specific domain of knowledge, using logical rules that are derived from the knowledge of experts. The earliest examples were developed by Edward Feigenbaum and his students. Dendral, begun in 1965, identified compounds from spectrometer readings. MYCIN, developed in 1972, diagnosed infectious blood diseases. They demonstrated the feasibility of the approach.

Expert systems restricted themselves to a small domain of specific knowledge (thus avoiding the commonsense knowledge problem) and their simple design made it relatively easy for programs to be built and then modified once they were in place. All in all, the programs proved to be *useful*: something that AI had not been able to achieve up to this point.

In 1980, an expert system called XCON was completed at CMU for the Digital Equipment Corporation. It was an enormous success: it was saving the company 40 million dollars annually by 1986. Corporations around the world began to develop and deploy expert systems and by 1985 they were spending over a billion dollars on AI, most of it to in-house AI departments. An industry grew up to support them, including hardware companies like Symbolics and Lisp Machines and software companies such as IntelliCorp and Aion.

The knowledge revolution

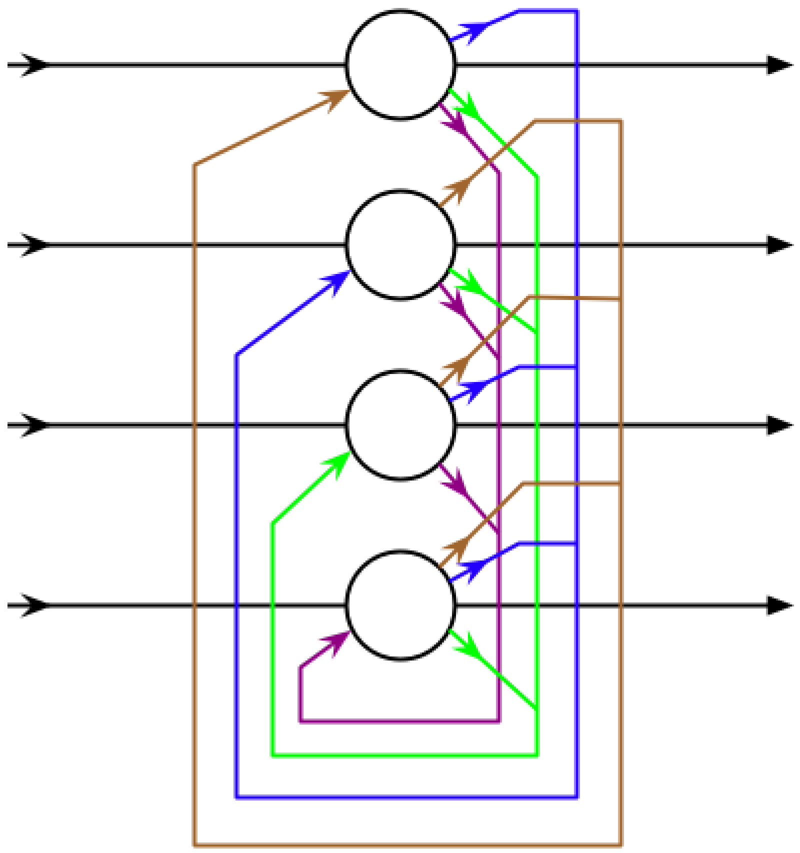
The power of expert systems came from the expert knowledge they contained. They were part of a new direction in AI research that had been gaining ground throughout the 70s. "AI researchers were beginning to suspect—reluctantly, for it violated the scientific canon of parsimony—that intelligence might very well be based on the ability to use large amounts of diverse knowledge in different ways," writes Pamela McCorduck. "[T]he great lesson from the 1970s was that intelligent behavior depended very much on dealing with knowledge, sometimes quite detailed knowledge, of a domain where a given task lay". Knowledge based systems and knowledge engineering became a major focus of AI research in the 1980s.

The 1980s also saw the birth of Cyc, the first attempt to attack the commonsense knowledge problem directly, by creating a massive database that would contain all the mundane facts that the average person knows. Douglas Lenat, who started and led the project, argued that there is no shortcut ― the only way for machines to know the meaning of human concepts is to teach them, one concept at a time, by hand. The project was not expected to be completed for many decades.

The money returns: the fifth generation project

In 1981, the Japanese Ministry of International Trade and Industry set aside $850 million dollars for the Fifth generation computer project. Their objectives were to write programs and build machines that could carry on conversations, translate languages, interpret pictures, and reason like human beings. Much to the chagrin of scruffies, they chose Prolog as the primary computer language for the project.

Other countries responded with new programs of their own. The UK began the ₤350 million Alvey project. A consortium of American companies formed the Microelectronics and Computer Technology Corporation (or "MCC") to fund large scale projects in AI and information technology. DARPA responded as well, founding the Strategic Computing Initiative and tripling its investment in AI between 1984 and 1988.



A Hopfield net with four nodes

The revival of connectionism

In 1982, physicist John Hopfield was able to prove that a form of neural network (now called a "Hopfield net") could learn and process information in a completely new way. Around the same time, David Rumelhart popularized a new method for training neural networks called "backpropagation" (discovered years earlier by Paul Werbos). These two discoveries revived the field of connectionism which had been largely abandoned since 1970.

The new field was unified and inspired by the appearance of *Parallel Distributed Processing* in 1986—a two volume collection of papers edited by Rumelhart and psychologist James McClelland. Neural networks would become commercially successful in the 1990s, when they began to be used as the engines driving programs like optical character recognition and speech recognition.

Bust: the second AI winter 1987−1993

The business community's fascination with AI rose and fell in the 80s in the classic pattern of an economic bubble. The collapse was in the *perception* of AI by government agencies and investors — the field continued to make advances despite the criticism.

Rodney Brooks and Hans Moravec, researchers from the related field of robotics, argued for an entirely new approach to artificial intelligence.

AI winter

The term "AI winter" was coined by researchers who had survived the funding cuts of 1974 when they became concerned that enthusiasm for expert systems had spiraled out of control and that disappointment would certainly follow. Their fears were well founded: in the late 80s and early 90s, AI suffered a series of financial setbacks.

The first indication of a change in weather was the sudden collapse of the market for specialized AI hardware in 1987. Desktop computers from Apple and IBM had been steadily gaining speed and power and in 1987 they became more powerful than the more expensive Lisp machines made by Symbolics and others. There was no longer a good reason to buy them. An entire industry worth half a billion dollars was demolished overnight.

Eventually the earliest successful expert systems, such as XCON, proved too expensive to maintain. They were difficult to update, they could not learn, they were "brittle" (i.e., they could make grotesque mistakes when given unusual inputs), and they fell prey to problems (such as the qualification problem) that had been identified years earlier. Expert systems proved useful, but only in a few special contexts.

In the late 80s, the Strategic Computing Initiative cut funding to AI "deeply and brutally." New leadership at DARPA had decided that AI was not "the next wave" and directed funds towards projects that seemed more likely to produce immediate results.

By 1991, the impressive list of goals penned in 1981 for Japan's Fifth Generation Project had not been met. Indeed, some of them, like "carry on a casual conversation" had not been met by 2010. As with other AI projects, expectations had run much higher than what was actually possible.

The importance of having a body: Nouvelle AI and embodied reason

In the late 80s, several researchers advocated a completely new approach to artificial intelligence, based on robotics. They believed that, to show real intelligence, a machine needs to have a *body* — it needs to perceive, move, survive and deal with the world. They argued that these sensorimotor skills are essential to higher level skills like commonsense reasoning and that abstract reasoning was actually the *least* interesting or important human skill. They advocated building intelligence "from the bottom up."

The approach revived ideas from cybernetics and control theory that had been unpopular since the sixties. Another precursor was David Marr, who had come to MIT in the late 70s from a successful background in theoretical neuroscience to lead the group studying vision. He rejected all symbolic approaches (*both* McCarthy's logic and Minsky's frames), arguing that AI needed to understand the physical machinery of vision from the bottom up before any symbolic processing took place. (Marr's work would be cut short by leukemia in 1980.)

In a 1990 paper Elephants Don't Play Chess, robotics researcher Rodney Brooks took direct aim at the physical symbol system hypothesis, arguing that symbols are not always necessary since "the world is its own best model. It is always exactly up to date. It always has every detail there is to be known. The trick is to sense it appropriately and often enough." In the 80s and 90s, many cognitive scientists also rejected the symbol processing model of the mind and argued that the body was essential for reasoning, a theory called the embodied mind thesis.

AI 1993−present

The field of AI, now more than a half a century old, finally achieved some of its oldest goals. It began to be used successfully throughout the technology industry, although somewhat behind the scenes. Some of the success was due to increasing computer power and some was achieved by focusing on specific isolated problems and pursuing them with the highest standards of scientific accountability. Still, the reputation of AI, in the business world at least, was less than pristine. Inside the field there was little agreement on the reasons for AI's failure to fulfill the dream of human level intelligence that had captured the imagination of the world in the 1960s. Together, all these factors helped to fragment AI into competing subfields focused on particular problems or approaches, sometimes even under new names that disguised the tarnished pedigree of "artificial intelligence". AI was both more cautious and more successful than it had ever been.

Milestones and Moore's Law

On 11 May 1997, Deep Blue became the first computer chess-playing system to beat a reigning world chess champion, Garry Kasparov. In 2005, a Stanford robot won the DARPA Grand Challenge by driving autonomously for 131 miles along an unrehearsed desert trail. In 2009, the Blue Brain Project announced that they had successfully simulated parts of a rat's brain.

These successes were not due to some revolutionary new paradigm, but mostly on the tedious application of engineering skill and on the tremendous power of computers today. In fact, Deep Blue's computer was 10 million times faster than the Ferranti Mark 1 that Christopher Strachey taught to play chess in 1951. This dramatic increase is measured by Moore's law, which predicts that the speed and memory capacity of computers doubles every two years. The fundamental problem of "raw computer power" was slowly being overcome.

Intelligent agents

A new paradigm called "intelligent agents" became widely accepted during the 90s. Although earlier researchers had proposed modular "divide and conquer" approaches to AI, the intelligent agent did not reach its modern form until Judea Pearl, Alan Newell and others brought concepts from decision theory and economics into the study of AI. When the economist's definition of a rational agent was married to computer science's definition of an object or module, the intelligent agent paradigm was complete.

An intelligent agent is a system that perceives its environment and takes actions which maximize its chances of success. The simplest intelligent agents are programs that solve specific problems. The most complicated intelligent agents known are rational, thinking human beings. The intelligent agent paradigm defines AI research as "the study of intelligent agents". This is a generalization of some earlier definitions of AI: it goes beyond studying human intelligence; it studies all kinds of intelligence.

The paradigm gave researchers license to study isolated problems and find solutions that were both verifiable and useful. It provided a common language to describe problems and share their solutions with each other, and with other fields that also used concepts of abstract agents, like economics and control theory. It was hoped that a complete agent architecture (like Newell's SOAR) would one day allow researchers to build more versatile and intelligent systems out of interacting intelligent agents.

Victory of the neats

AI researchers began to develop and use sophisticated mathematical tools more than they ever had in the past. There was a widespread realization that many of the problems that AI needed to solve were already being worked on by researchers in fields like mathematics, economics or operations research. The shared mathematical language allowed both a higher level of collaboration with more established and successful fields and the achievement of results which were measurable and provable; AI had become a more rigorous "scientific" discipline. Russell & Norvig (2003) describe this as nothing less than a "revolution" and "the victory of the neats".

Judea Pearl's highly influential 1988 book brought probability and decision theory into AI. Among the many new tools in use were Bayesian networks, hidden Markov models, information theory, stochastic modeling and classical optimization. Precise mathematical descriptions were also developed for "computational intelligence" paradigms like neural networks and evolutionary algorithms.

AI behind the scenes

Algorithms originally developed by AI researchers began to appear as parts of larger systems. AI had solved a lot of very difficult problems and their solutions proved to be useful throughout the technology industry, such as data mining, industrial robotics, logistics, speech recognition, banking software, medical diagnosis and Google's search engine.

The field of AI receives little or no credit for these successes. Many of AI's greatest innovations have been reduced to the status of just another item in the tool chest of computer science.

Nick Bostrom explains "A lot of cutting edge AI has filtered into general applications, often without being called AI because once something becomes useful enough and common enough it's not labeled AI anymore."

Many researchers in AI today deliberately call their work by other names, such as informatics, knowledge-based systems, cognitive systems or computational intelligence. In part, this may be because they considered their field to be fundamentally different from AI, but also the new names help to procure funding. In the commercial world at least, the failed promises of the AI Winter continue to haunt AI research, as the New York Times reported in 2005: "Computer scientists and software engineers avoided the term artificial intelligence for fear of being viewed as wild-eyed dreamers."

Where is HAL 9000?

In 1968, Arthur C. Clarke and Stanley Kubrick had imagined that by the year 2001, a machine would exist with an intelligence that matched or exceeded the capability of human beings. The character they created, HAL 9000, was based on hard science: many leading AI researchers also believed that such a machine would exist by the year 2001.

Marvin Minsky asks "So the question is why didn't we get HAL in 2001?" Minsky believes that the answer is that the central problems, like commonsense reasoning, were being neglected, while most researchers pursued things like commercial applications of neural nets or genetic algorithms. John McCarthy, on the other hand, still blames the qualification problem. For Ray Kurzweil, the issue is computer power and, using Moore's Law, he predicts that machines with human-level intelligence will appear by 2029. Jeff Hawkins argues that neural net research ignores the essential properties of the human cortex, preferring simple models that have been successful at solving simple problems. There are many other explanations and for each there is a corresponding research program underway.

History of computer science

The **history of computer science** began long before the modern discipline of computer science that emerged in the twentieth century, and hinted at in the centuries prior. The progression, from mechanical inventions and mathematical theories towards the modern concepts and machines, formed a major academic field and the basis of a massive worldwide industry.

Early history

Early computation

The earliest known tool for use in computation was the abacus, and it was thought to have been invented in Babylon circa 2400 BCE. Its original style of usage was by lines drawn in sand with pebbles. This was the first known computer and most advanced system of calculation known to date - preceding Greek methods by 2,000 years. Abaci of a more modern design are still used as calculation tools today.

The Antikythera mechanism is believed to be the earliest known mechanical analog computer. It was designed to calculate astronomical positions. It was discovered in 1901 in the Antikythera wreck off the Greek island of Antikythera, between Kythera and Crete, and has been dated to *circa* 100 BC. Technological artifacts of similar complexity did not reappear until the 14th century, when mechanical astronomical clocks appeared in Europe.

In the 3rd century CE the South Pointing Chariot was invented in ancient China. It was the first known geared mechanism to use a differential gear, which was later used in analog computers. The Chinese also invented a more sophisticated abacus from around the 2nd century BCE, known as the Chinese abacus.

Mechanical analog computing devices appeared again a thousand years later in the medieval Islamic world. Examples of devices from this period include the equatorium by Arzachel, the mechanical geared astrolabe by Abū Rayhān al-Bīrūnī, and the torquetum by Jabir ibn Aflah. Muslim engineers built a number of Automata, including some musical automata that could be 'programmed' to play different musical patterns. These devices were developed by the Banū Mūsā brothers and Al-Jazari Muslim mathematicians also made important advances in cryptography, such as the development of cryptanalysis and frequency analysis by Alkindus.

When John Napier discovered logarithms for computational purposes in the early 17th century, there followed a period of considerable progress by inventors and scientists in making calculating tools. In 1623 Wilhelm Schickard designed a calculating machine, but abandoned the project, when the prototype he had started building was destroyed by a fire in 1624. Around 1640, Blaise Pascal, a leading French mathematician, constructed the first mechanical adding device based on a design described by Greek mathematician Hero of Alexandria. Then in 1672 Gottfried Wilhelm Leibniz invented the Stepped Reckoner which he completed in 1694.

None of the early computational devices were really computers in the modern sense, and it took considerable advancement in mathematics and theory before the first modern computers could be designed.

Algorithms

In the 7th century, Indian mathematician Brahmagupta gave the first explanation of the Hindu-Arabic numeral system and the use of zero as both a placeholder and a decimal digit.

Approximately around the year 825, Persian mathematician Al-Khwarizmi wrote a book, *On the Calculation with Hindu Numerals*, that was principally responsible for the diffusion of the Indian system of numeration in the Middle East and then Europe. Around the 12th century, there was translation of this book written into Latin: *Algoritmi de numero Indorum*. These books presented newer concepts to perform a series of steps in order to accomplish a task such as the systematic application of arithmetic to algebra. By derivation from his name, we have the term algorithm.

Binary logic

Around the 3rd century BC, Indian mathematician Pingala discovered the binary numeral system. In this system, still used today in all modern computers, a sequence of ones and zeros can represent any number.

In 1703, Gottfried Leibnitz developed logic in a formal, mathematical sense with his writings on the binary numeral system. In his system, the ones and zeros also represent *true* and *false* values or *on* and *off* states. But it took more than a century before George Boole published his Boolean algebra in 1854 with a complete system that allowed computational processes to be mathematically modeled.

By this time, the first mechanical devices driven by a binary pattern had been invented. The industrial revolution had driven forward the mechanization of many tasks, and this included weaving. Punched cards controlled Joseph Marie Jacquard's loom in 1801, where a hole punched in the card indicated a binary *one* and an unpunched spot indicated a binary *zero*. Jacquard's loom was far from being a computer, but it did illustrate that machines could be driven by binary systems.

Birth of computer science

Before the 1920s, *computers* (sometimes *computors*) were human clerks that performed computations. They were usually under the lead of a physicist. Many thousands of computers were employed in commerce, government, and research establishments. Most of these computers were women, and they were known to have a degree in calculus. Some performed astronomical calculations for calendars.

After the 1920s, the expression *computing machine* referred to any machine that performed the work of a human computer, especially those in accordance with effective methods of the Church-Turing thesis. The thesis states that a mathematical method is effective if it could be set out as a list of instructions able to be followed by a human clerk with paper and pencil, for as long as necessary, and without ingenuity or insight.

Machines that computed with continuous values became known as the *analog* kind. They used machinery that represented continuous numeric quantities, like the angle of a shaft rotation or difference in electrical potential.

Digital machinery, in contrast to analog, were able to render a state of a numeric value and store each individual digit. Digital machinery used difference engines or relays before the invention of faster memory devices.

The phrase *computing machine* gradually gave away, after the late 1940s, to just *computer* as the onset of electronic digital machinery became common. These computers were able to perform the calculations that were performed by the previous human clerks.

Since the values stored by digital machines were not bound to physical properties like analog devices, a logical computer, based on digital equipment, was able to do anything that could be described "purely mechanical." The theoretical Turing Machine, created by Alan Turing, is a hypothetical device theorized in order to study the properties of such hardware.

Emergence of a discipline

The theoretical groundwork

The mathematical foundations of modern computer science began to be laid by Kurt Gödel with his incompleteness theorem (1931). In this theorem, he showed that there were limits to what could be proved and disproved within a formal system. This led to work by Gödel and others to define and describe these formal systems, including concepts such as mu-recursive functions and lambda-definable functions.

1936 was a key year for computer science. Alan Turing and Alonzo Church independently, and also together, introduced the formalization of an algorithm, with limits on what can be computed, and a "purely mechanical" model for computing.

These topics are covered by what is now called the Church–Turing thesis, a hypothesis about the nature of mechanical calculation devices, such as electronic computers. The thesis claims that any calculation that is possible can be performed by an algorithm running on a computer, provided that sufficient time and storage space are available.

Turing also included with the thesis a description of the Turing machine. A Turing machine has an infinitely long tape and a read/write head that can move along the tape, changing the values along the way. Clearly such a machine could never be built, but nonetheless, the model can simulate the computation of any algorithm which can be performed on a modern computer.

Turing is so important to computer science that his name is also featured on the Turing Award and the Turing test. He contributed greatly to British code-breaking successes in the Second World War, and continued to design computers and software through the 1940s, but committed suicide in 1954.

At a symposium on large-scale digital machinery in Cambridge, Turing said, "We are trying to build a machine to do all kinds of different things simply by programming rather than by the addition of extra apparatus".

In 1948, the first practical computer that could run stored programs, based on the Turing machine model, had been built - the Manchester Baby.

In 1950, Britain's National Physical Laboratory completed Pilot ACE, a small scale programmable computer, based on Turing's philosophy.

Shannon and information theory

Up to and during the 1930s, electrical engineers were able to build electronic circuits to solve mathematical and logic problems, but most did so in an *ad hoc* manner, lacking any theoretical rigor. This changed with Claude Elwood Shannon's publication of his 1937 master's thesis, A Symbolic Analysis of Relay and Switching Circuits. While taking an undergraduate philosophy class, Shannon had been exposed to Boole's work, and recognized that it could be used to arrange electromechanical relays (then used in telephone routing switches) to solve logic problems. This concept, of utilizing the properties of electrical switches to do logic, is the basic concept that underlies all electronic digital computers, and his thesis became the foundation of practical digital circuit design when it became widely known among the electrical engineering community during and after World War II.

Shannon went on to found the field of information theory with his 1948 paper titled A Mathematical Theory of Communication, which applied probability theory to the problem of how to best encode the information a sender wants to transmit. This work is one of the theoretical foundations for many areas of study, including data compression and cryptography.

Wiener and Cybernetics

From experiments with anti-aircraft systems that interpreted radar images to detect enemy planes, Norbert Wiener coined the term cybernetics from the Greek word for "steersman." He published "Cybernetics" in 1948, which influenced artificial intelligence.

Wiener also compared computation, computing machinery, memory devices, and other cognitive similarities with his analysis of brain waves.

The first actual computer bug was a moth. It was stuck in between the relays on the Harvard Mark II. While the invention of the term 'bug' is often but erroneously attributed to Grace Hopper, a future rear admiral in the U.S. Navy, who supposedly logged the "bug" on September 9, 1945, most other accounts conflict at least with these details. According to these accounts, the actual date was September 9, 1947 when operators filed this 'incident' — along with the insect and the notation "First actual case of bug being found".

Chapter 5

History of Operating Systems and History of Programming Languages

History of operating systems

The **history of computer operating systems** recapitulates to a degree the recent history of computer hardware.

Operating systems (OSes) provide a set of functions needed and used by most application programs on a computer, and the linkages needed to control and synchronize computer hardware. On the first computers, with no operating system, every program needed the full hardware specification to run correctly and perform standard tasks, and its own drivers for peripheral devices like printers and punched paper card readers. The growing complexity of hardware and application programs eventually made operating systems a necessity.

Background

The earliest computers lacked any form of operating system. Each user had sole use of the machine and would arrive armed with program and data, often on punched paper cards and magnetic or paper 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 control 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.

Symbolic languages, assemblers, and compilers were developed for programmers to translate symbolic program-code into binary code that previously would have been hand-encoded. Later machines came with libraries of support code on punched cards or magnetic tape, which would be linked to the user's program to assist in operations such as input and output. 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.

As machines became more powerful, the time to run programs diminished and the time to hand off the equipment to the next user became very large by comparison. Accounting for and paying for machine usage moved on from checking the wall clock to automatic logging by the computer. Run queues evolved from a literal queue of people at the door, to a heap of media on a jobs-waiting table, or batches of punch-cards stacked one on top of the other in the reader, until the machine itself was able to select and sequence which magnetic tape drives were online. Where program developers had originally had access to run their own jobs on the machine, they were supplanted by dedicated machine operators who looked after the well-being and maintenance of the machine and were less and less concerned with implementing tasks manually. When commercially available computer centers were faced with the implications of data lost through tampering or operational errors, equipment vendors were put under pressure to enhance the runtime libraries to prevent misuse of system resources. Automated monitoring was needed not just for CPU usage but for counting pages printed, cards punched, cards read, disk storage used and for signaling when operator intervention was required by jobs such as changing magnetic tapes. Security features were added to operating systems to record audit trails of which programs were accessing which files and to prevent access to a production payroll file by an engineering program, for example.

All these features were building up towards the repertoire of a fully capable operating system. Eventually the runtime libraries became an amalgamated program that was started before the first customer job and could read in the customer job, control its execution, record its usage, reassign hardware resources after the job ended, and immediately go on to process the next job. These resident background programs, capable of managing multistep processes, were often called monitors or monitor-programs before the term OS established itself.

An underlying program offering basic hardware-management, software-scheduling and resource-monitoring may seem a remote ancestor to the user-oriented OSes of the personal computing era. But there has been a shift in the meaning of OS. Just as early automobiles lacked speedometers, radios, and air-conditioners which later became standard, more and more optional software features became standard features in every OS package, although some applications such as data base management systems and spreadsheets remain optional and separately priced. This has lead to the perception of an OS as a complete user-system with an integrated graphical user interface, utilities, some applications such as text editors and file managers, and configuration tools.

The true descendant of the early operating systems is what is now called the "kernel". In technical and development circles the old restricted sense of an OS persists because of the continued active development of embedded operating systems for all kinds of devices with a data-processing component, from hand-held gadgets up to industrial robots and real-time control-systems, which do not run user applications at the front-end. An embedded OS in a device today is not so far removed as one might think from its ancestor of the 1950s.

The broader categories of systems and application software are discussed in the computer software article.

The mainframe era

It is generally thought that the first operating system used for real work was GM-NAA I/O, produced in 1956 by General Motors' Research division for its IBM 704. Most other early operating systems for IBM mainframes were also produced by customers.

Early operating systems were very diverse, with each vendor or customer producing one or more operating systems specific to their particular mainframe computer. Every operating system, even from the same vendor, could have radically different models of commands, operating procedures, and such facilities as debugging aids. Typically, each time the manufacturer brought out a new machine, there would be a new operating system, and most applications would have to be manually adjusted, recompiled, and retested.

Systems on IBM hardware

The state of affairs continued until the 1960s when IBM, already a leading hardware vendor, stopped work on existing systems and put all their effort into developing the System/360 series of machines, all of which used the *same* instruction and input/output architecture. IBM intended to develop a single operating system for the new hardware, the OS/360. The problems encountered in the development of the OS/360 are legendary, and are described by Fred Brooks in *The Mythical Man-Month*—a book that has become a classic of software engineering. Because of performance differences across the hardware range and delays with software development, a whole family of operating systems were introduced instead of a single OS/360.

IBM wound up releasing a series of stop-gaps followed by two longer-lived operating systems:

- OS/360 for mid-range and large systems. This was available in three System generation options:
    - PCP for early users and for those without the resources for multiprogramming.
    - MFT for mid-range systems. This had one successor, OS/VS1, which was discontinued in the 1980s.
    - MVT for large systems. This was similar in most ways to MFT (programs could be ported between the two without being re-compiled), but has more sophisticated memory management and a time-sharing facility, TSO. MVT had several successors including the current z/OS.

- DOS/360 for small System/360 models had several successors including the current z/VSE. It was significantly different from OS/360.

IBM maintained full compatibility with the past, so that programs developed in the sixties can still run under z/VSE (if developed for DOS/360) or z/OS (if developed for MFT or MVT) with no change.

IBM also developed, but never officially released, TSS/360, a time-sharing system for the S/360 Model 67.

Several operating systems for the IBM S/360 and S/370 architectures were developed by third parties, including the Michigan Terminal System (MTS) and MUSIC/SP.

Other mainframe operating systems

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 DTSS time sharing system, one of the pioneering efforts in timesharing and programming languages.

In the late 1970s, Control Data and the University of Illinois developed the PLATO system, which used plasma panel displays and long-distance time sharing networks. PLATO was remarkably innovative for its time; the shared memory model of PLATO's TUTOR programming language allowed applications such as real-time chat and multi-user graphical games.

UNIVAC, the first commercial computer manufacturer, produced a series of EXEC operating systems. Like all early main-frame systems, this was a batch-oriented system that 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 BASIC system.

Burroughs Corporation introduced the B5000 in 1961 with the MCP (Master Control Program) operating system. The B5000 was a stack machine designed to exclusively support high-level languages, with no software, not even at the lowest level of the operating system, being written directly in machine language or assembly language; the MCP was the first OS to be written entirely in a high-level language - ESPOL, a dialect of ALGOL - although ESPOL had specialized statements for each "syllable" (opcode) in the B5000 instruction set. MCP also introduced many other ground-breaking innovations, such as being one of the first commercial implementations of virtual memory. The rewrite of MCP for the B6500 is still in use today in the Unisys ClearPath/MCP line of computers.

GE introduced the GE 600 series with the General Electric Comprehensive Operating Supervisor (GECOS) operating system. After Honeywell acquired GE's computer business, it was renamed to General Comprehensive Operating System (GCOS).

Project MAC at MIT, working with GE and BTL, developed Multics, which introduced the concept of ringed security privilege levels.

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. Before the widespread use of Unix, TOPS-10 was a particularly popular system in universities, and in the early ARPANET community.

In 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 architecture to appear to be the same as others in a series. In fact most 360's after the 360/40 (except the 360/165 and 360/168) were microprogrammed implementations. But soon other means of achieving application compatibility were proven to be more significant.

Minicomputers and the rise of Unix

The beginnings of the Unix operating system was developed at AT&T Bell Laboratories in the late 1960s. Because it was essentially free in early editions, easily obtainable, and easily modified, it achieved wide acceptance. It also became a requirement within the Bell systems operating companies. Since it was written in the C language, when that language was ported to a new machine architecture, Unix was also able to be ported. This portability permitted it to become the choice for a second generation of minicomputers and the first generation of workstations. By widespread use it exemplified the idea of an operating system that was conceptually the same across various hardware platforms. It still was owned by AT&T and that limited its use to groups or corporations who could afford to license it. It became one of the roots of the open source movement.

Other than that, Digital Equipment Corporation created several operating systems for its 16-bit PDP-11 class machines, including the simple RT-11 system, the time-sharing RSTS operating systems, and the RSX-11 family of real-time operating systems, and the VMS system for the 32-bit VAX computer.

Another system which evolved in this time frame was the Pick operating system. The Pick system was developed and sold by Microdata Corporation who created the precursors of the system. The system is an example of a system which started as a database application support program and graduated to system work.

The case of 8-bit home computers and game consoles

Home computers

Most small 8-bit home computers of the 1980s, such as the Commodore 64, Apple II, the Atari 8-bit, the Amstrad CPC, ZX Spectrum series and others could use a disk-loading operating system, such as CP/M or GEOS, but they could also generally work without one. Most, if not all, of these computers shipped with a built-in BASIC interpreter on ROM, which also served as a crude operating system, allowing minimal file managing operations (such as deleting, copying, etc.) to be performed and sometimes disk formatting, and application loading and executing, which sometimes needed a non-trivial command sequence, as with the Commodore 64.

True operating systems were unneeded in part because most such machines were used for entertainment and education, and seldom used for more serious business or science purposes.

Another reason is that they were usually single-task and single-user machines and shipped with minimal amounts of computer memory, usually between 4 and 256 kilobytes, with 64 and 128 being common figures, and 8-bit processors, so an operating system's overhead would likely compromise the performance of the machine without really being needed.

Even the available word processor and integrated software applications were mostly self-contained programs which took over the machine completely, as also did video games.

Game consoles and video games

Since virtually all video game consoles and arcade cabinets designed and built after 1980 were true digital machines (unlike the analog *Pong* clones and derivatives), some of them carried a minimal form of BIOS or built-in game, such as the ColecoVision, the Sega Master System and the SNK Neo Geo. There were however successful designs where a BIOS was not needed, such as the original Nintendo Entertainment System and its clones.

Modern day game consoles and videogames, starting with the PC-Engine, all have a minimal BIOS that also provides some interactive utilities such as memory card management, audio or video CD playback, copy protection and sometimes carry libraries for developers to use etc. Few of these cases, however, would qualify as a true operating system.

The most notable exceptions are probably the Dreamcast game console which includes a minimal BIOS, like the PlayStation, but can load the Windows CE operating system from the game disk allowing easily porting of games from the PC world, and the Xbox game console, which is little more than a disguised Intel-based PC running a secret, modified version of Microsoft Windows in the background. Furthermore, there are Linux versions that will run on a Dreamcast and later game consoles as well.

Long before that, Sony had released a kind of development kit called the Net Yaroze for its first PlayStation platform, which provided a series of programming and developing tools to be used with a normal PC and a specially modified "Black PlayStation" that could be interfaced with a PC and download programs from it. These operations require in general a functional OS on both platforms involved.

In general, it can be said that videogame consoles and arcade coin operated machines used at most a built-in BIOS during the 1970s, 1980s and most of the 1990s, while from the PlayStation era and beyond they started getting more and more sophisticated, to the point of requiring a generic or custom-built OS for aiding in development and expandability.

The personal computer era: Apple, Amiga, PC/MS/DR-DOS and beyond

The development of microprocessors made inexpensive computing available for the small business and hobbyist, which in turn led to the widespread use of interchangeable hardware components using a common interconnection (such as the S-100, SS-50, Apple II, ISA, and PCI buses), and an increasing need for "standard" operating systems to control them. The most important of the early OSes on these machines was Digital Research's CP/M-80 for the 8080 / 8085 / Z-80 CPUs. It was based on several Digital Equipment Corporation operating systems, mostly for the PDP-11 architecture. Microsoft's first operating system, M-DOS, was designed along many of the PDP-11 features, but for microprocessor based systems. MS-DOS, or PC-DOS when supplied by IBM, was based originally on CP/M-80. Each of these machines had a small boot program in ROM which loaded the OS itself from disk. The BIOS on the IBM-PC class machines was an extension of this idea and has accreted more features and functions in the 20 years since the first IBM-PC was introduced in 1981.

The decreasing cost of display equipment and processors made it practical to provide graphical user interfaces for many operating systems, such as the generic X Window System that is provided with many Unix systems, or other graphical systems such as Microsoft Windows, the RadioShack Color Computer's OS-9 Level II/MultiVue, Commodore's AmigaOS, Apple's Mac OS, or even IBM's OS/2. The original GUI was developed at Xerox Palo Alto Research Center in the early '70s (the Alto computer system) and imitated by many vendors.

The rise of virtualization

Operating systems originally ran directly on the hardware itself and provided services to applications. With CP-67 on the IBM System/360 Model 67 and Virtual Machine Facility/370 (**VM/370**) on System/370, IBM introduced the notion of virtual machine, where the operating system itself runs under the control of a hypervisor, instead of being in direct control of the hardware. VMware popularized this technology on personal computers. Over time, the line between virtual machines, monitors, and operating systems was blurred:

- Hypervisors grew more complex, gaining their own application programming interface, memory management or file system.
- Virtualization becomes a key feature of operating systems, as exemplified by Hyper-V in Windows Server 2008 or HP Integrity Virtual Machines in HP-UX.
- In some systems, such as POWER5 and POWER6-based servers from IBM, the hypervisor is no longer optional.
- Applications have been re-designed to run directly on a virtual machine monitor.

In many ways, virtual machine software today plays the role formerly held by the operating system, including managing the hardware resources (processor, memory, I/O devices), applying scheduling policies, or allowing system administrators to manage systems.

History of programming languages

Before 1940

The first programming languages predate the modern computer. At first, the languages were codes.

The Jacquard loom, invented in 1801, used holes in punched cards to represent sewing loom arm movements in order to generate decorative patterns automatically.

During a nine-month period in 1842-1843, Ada Lovelace translated the memoir of Italian mathematician Luigi Menabrea about Charles Babbage's newest proposed machine, the Analytical Engine. With the article, she appended a set of notes which specified in complete detail a method for calculating Bernoulli numbers with the Engine, recognized by some historians as the world's first computer program.

Herman Hollerith realized that he could encode information on punch cards when he observed that train conductors encode the appearance of the ticket holders on the train tickets using the position of punched holes on the tickets. Hollerith then encoded the 1890 census data on punch cards.

The first computer codes were specialized for their applications. In the first decades of the 20th century, numerical calculations were based on decimal numbers. Eventually it was realized that logic could be represented with numbers, not only with words. For example, Alonzo Church was able to express the lambda calculus in a formulaic way. The Turing machine was an abstraction of the operation of a tape-marking machine, for example, in use at the telephone companies. Turing machines set the basis for storage of programs as data in the von Neumann architecture of computers by representing a machine through a finite number. However, unlike the lambda calculus, Turing's code does not serve well as a basis for higher-level languages—its principal use is in rigorous analyses of algorithmic complexity.

Like many "firsts" in history, the first modern programming language is hard to identify. From the start, the restrictions of the hardware defined the language. Punch cards allowed 80 columns, but some of the columns had to be used for a sorting number on each card. FORTRAN included some keywords which were the same as English words, such as "IF", "GOTO" (go to) and "CONTINUE". The use of a magnetic drum for memory meant that computer programs also had to be interleaved with the rotations of the drum. Thus the programs were more hardware-dependent.

To some people, what was the first modern programming language depends on how much power and human-readability is required before the status of "programming language" is granted. Jacquard looms and Charles Babbage's Difference Engine both had simple, extremely limited languages for describing the actions that these machines should perform. One can even regard the punch holes on a player piano scroll as a limited domain-specific language, albeit not designed for human consumption.

The 1940s

In the 1940s, the first recognizably modern, electrically powered computers were created. The limited speed and memory capacity forced programmers to write hand tuned assembly language programs. It was soon discovered that programming in assembly language required a great deal of intellectual effort and was error-prone.

In 1948, Konrad Zuse published a paper about his programming language Plankalkül. However, it was not implemented in his lifetime and his original contributions were isolated from other developments.

Some important languages that were developed in this period include:

- 1943 - Plankalkül (Konrad Zuse), designed, but unimplemented for a half-century
- 1943 - ENIAC coding system, machine-specific codeset appearing in 1948
- 1949 - 1954 — a series of machine-specific mnemonic instruction sets, like ENIAC's, beginning in 1949 with C-10 for BINAC (which later evolved into UNIVAC). Each codeset, or instruction set, was tailored to a specific manufacturer.

The 1950s and 1960s

In the 1950s, the first three modern programming languages whose descendants are still in widespread use today were designed:

- FORTRAN (1955), the "**FOR**mula **TRAN**slator", invented by John Backus et al.;
- LISP [1958], the "**LIS**t **P**rocessor", invented by John McCarthy et al.;
- COBOL, the **CO**mmon **B**usiness **O**riented **L**anguage, created by the Short Range Committee, heavily influenced by Grace Hopper.

Another milestone in the late 1950s was the publication, by a committee of American and European computer scientists, of "a new language for algorithms"; the *ALGOL 60 Report* (the "**ALGO**rithmic **L**anguage"). This report consolidated many ideas circulating at the time and featured two key language innovations:

- nested block structure: code sequences and associated declarations could be grouped into blocks without having to be turned into separate, explicitly named procedures;
- lexical scoping: a block could have its own private variables, procedures and functions, invisible to code outside that block, i.e. information hiding.

Another innovation, related to this, was in how the language was described:

- a mathematically exact notation, Backus-Naur Form (BNF), was used to describe the language's syntax. Nearly all subsequent programming languages have used a variant of BNF to describe the context-free portion of their syntax.

Algol 60 was particularly influential in the design of later languages, some of which soon became more popular. The Burroughs large systems were designed to be programmed in an extended subset of Algol.

Algol's key ideas were continued, producing ALGOL 68:

- syntax and semantics became even more orthogonal, with anonymous routines, a recursive typing system with higher-order functions, etc.;
- not only the context-free part, but the full language syntax and semantics were defined formally, in terms of Van Wijngaarden grammar, a formalism designed specifically for this purpose.

Algol 68's many little-used language features (e.g. concurrent and parallel blocks) and its complex system of syntactic shortcuts and automatic type coercions made it unpopular with implementers and gained it a reputation of being *difficult*. Niklaus Wirth actually walked out of the design committee to create the simpler Pascal language.

Some important languages that were developed in this period include:

- 1951 - Regional Assembly Language
- 1952 - Autocode
- 1954 - FORTRAN
- 1954 - IPL (forerunner to LISP)
- 1955 - FLOW-MATIC (forerunner to COBOL)

- 1957 - COMTRAN (forerunner to COBOL)
- 1958 - LISP
- 1958 - ALGOL 58
- 1959 - FACT (forerunner to COBOL)
- 1959 - COBOL
- 1962 - APL
- 1962 - Simula
- 1962 - SNOBOL
- 1963 - CPL (forerunner to C)
- 1964 - BASIC
- 1964 - PL/I
- 1967 - BCPL (forerunner to C)

1967-1978: establishing fundamental paradigms

The period from the late 1960s to the late 1970s brought a major flowering of programming languages. Most of the major language paradigms now in use were invented in this period:

- **Simula**, invented in the late 1960s by Nygaard and Dahl as a superset of Algol 60, was the first language designed to support object-oriented programming.
- **C**, an early systems programming language, was developed by Dennis Ritchie and Ken Thompson at Bell Labs between 1969 and 1973.
- **Smalltalk** (mid 1970s) provided a complete ground-up design of an object-oriented language.
- **Prolog**, designed in 1972 by Colmerauer, Roussel, and Kowalski, was the first logic programming language.
- **ML** built a polymorphic type system (invented by Robin Milner in 1973) on top of Lisp, pioneering statically typed functional programming languages.

Each of these languages spawned an entire family of descendants, and most modern languages count at least one of them in their ancestry.

The 1960s and 1970s also saw considerable debate over the merits of "structured programming", which essentially meant programming without the use of Goto. This debate was closely related to language design: some languages did not include GOTO, which forced structured programming on the programmer. Although the debate raged hotly at the time, nearly all programmers now agree that, even in languages that provide GOTO, it is bad programming style to use it except in rare circumstances. As a result, later generations of language designers have found the structured programming debate tedious and even bewildering.

Some important languages that were developed in this period include:

- 1968 - Logo
- 1970 - Pascal

- 1970 - Forth
- 1972 - C
- 1972 - Smalltalk
- 1972 - Prolog
- 1973 - ML
- 1975 - Scheme
- 1978 - SQL (initially only a query language, later extended with programming constructs)

The 1980s: consolidation, modules, performance

The 1980s were years of relative consolidation in imperative languages. Rather than inventing new paradigms, all of these movements elaborated upon the ideas invented in the previous decade. C++ combined object-oriented and systems programming. The United States government standardized Ada, a systems programming language intended for use by defense contractors. In Japan and elsewhere, vast sums were spent investigating so-called fifth-generation programming languages that incorporated logic programming constructs. The functional languages community moved to standardize ML and Lisp. Research in Miranda, a functional language with lazy evaluation, began to take hold in this decade.

One important new trend in language design was an increased focus on programming for large-scale systems through the use of *modules*, or large-scale organizational units of code. Modula, Ada, and ML all developed notable module systems in the 1980s. Module systems were often wedded to generic programming constructs---generics being, in essence, parameterized modules.

Although major new paradigms for imperative programming languages did not appear, many researchers expanded on the ideas of prior languages and adapted them to new contexts. For example, the languages of the Argus and Emerald systems adapted object-oriented programming to distributed systems.

The 1980s also brought advances in programming language implementation. The RISC movement in computer architecture postulated that hardware should be designed for compilers rather than for human assembly programmers. Aided by processor speed improvements that enabled increasingly aggressive compilation techniques, the RISC movement sparked greater interest in compilation technology for high-level languages.

Language technology continued along these lines well into the 1990s.

Some important languages that were developed in this period include:

- 1980 - C++ (as C with classes, name changed in July 1983)
- 1983 - Objective-C
- 1983 - Ada
- 1984 - Common Lisp

- 1985 - Eiffel
- 1986 - Erlang
- 1987 - Perl
- 1988 - Tcl
- 1989 - FL (Backus)

The 1990s: the Internet age

The 1990s saw no fundamental novelty in imperative languages, but much recombination and maturation of old ideas. This era began the spread of functional languages. A big driving philosophy was programmer productivity. Many "rapid application development" (RAD) languages emerged, which usually came with an IDE, garbage collection, and were descendants of older languages. All such languages were object-oriented. These included Object Pascal, Visual Basic, and Java. Java in particular received much attention. More radical and innovative than the RAD languages were the new scripting languages. These did not directly descend from other languages and featured new syntaxes and more liberal incorporation of features. Many consider these scripting languages to be more productive than even the RAD languages, but often because of choices that make small programs simpler but large programs more difficult to write and maintain. Nevertheless, scripting languages came to be the most prominent ones used in connection with the Web.

Some important languages that were developed in this period include:

- 1990 - Haskell
- 1991 - Python
- 1991 - Visual Basic
- 1993 - Ruby
- 1993 - Lua
- 1994 - CLOS (part of ANSI Common Lisp)
- 1995 - Java
- 1995 - Delphi (Object Pascal)
- 1995 - JavaScript
- 1995 - PHP
- 1997 - Rebol
- 1999 - D

Current trends

Programming language evolution continues, in both industry and research. Some of the current trends include:

- Mechanisms for adding security and reliability verification to the language: extended static checking, information flow control, static thread safety.
- Alternative mechanisms for modularity: mixins, delegates, aspects.

- Component-oriented software development.
- Constructs to support concurrent and distributed programming.
- Metaprogramming, reflection or access to the abstract syntax tree
- Increased emphasis on distribution and mobility.
- Integration with databases, including XML and relational databases.
- Support for Unicode so that source code (program text) is not restricted to those characters contained in the ASCII character set; allowing, for example, use of non-Latin-based scripts or extended punctuation.
- XML for graphical interface (XUL, XAML).

Some important languages developed during this period include:

- 2001 - C#
- 2001 - Visual Basic .NET
- 2002 - F#
- 2003 - Scala
- 2003 - Factor
- 2006 - Windows Power Shell
- 2007 - Clojure
- 2007 - Groovy
- 2009 - Go

Chapter 6

Introduction to History of the Internet

The **history of the Internet** starts in the 1950s and 1960s with the development of computers. This began with point-to-point communication between mainframe computers and terminals, expanded to point-to-point connections between computers and then early research into packet switching. Packet switched networks such as ARPANET, Mark I at NPL in the UK, CYCLADES, Merit Network, Tymnet, and Telenet, were developed in the late 1960s and early 1970s using a variety of protocols. The ARPANET in particular lead to the development of protocols for internetworking, where multiple separate networks could be joined together into a network of networks.

In 1982 the Internet Protocol Suite (TCP/IP) was standardized and the concept of a world-wide network of fully interconnected TCP/IP networks called the Internet was introduced. Access to the ARPANET was expanded in 1981 when the National Science Foundation (NSF) developed the Computer Science Network (CSNET) and again in 1986 when NSFNET provided access to supercomputer sites in the United States from research and education organizations. The ARPANET was decommissioned in 1990. Commercial internet service providers (ISPs) began to emerge in the late 1980s and 1990s and the Internet was commercialized in 1995 when NSFNET was decommissioned, removing the last restrictions on the use of the Internet to carry commercial traffic.

Since the mid-1990s the Internet has had a drastic impact on culture and commerce, including the rise of near instant communication by electronic mail, text-based discussion forums and the World Wide Web. The research and education community continues to use advanced networks such as NSF's very high speed Backbone Network Service (vBNS) and Internet2. Increasing amounts of data are transmitted at higher and higher speeds over fiber optic networks operating at 1-Gbps, 10-Gbps, or more. The Internet continues to grow, driven by ever greater amounts of online information and knowledge, by commerce and entertainment, and by social networking.

Precursors

The Internet has precursors that date back to the 19th century, especially the telegraph system, more than a century before the digital Internet became widely used in the second half of the 1990s. The concept of data communication – transmitting data between two different places, connected via some kind of electromagnetic medium, such as radio or an electrical wire – predates the introduction of the first computers. Such communication systems were typically limited to point to point communication between two end devices. Telegraph systems and telex machines can be considered early precursors of this kind of communication.

Early computers used the technology available at the time to allow communication between the central processing unit and remote terminals. As the technology evolved, new systems were devised to allow communication over longer distances (for terminals) or with higher speed (for interconnection of local devices) that were necessary for the mainframe computer model. Using these technologies it was possible to exchange data (such as files) between remote computers. However, the point to point communication model was limited, as it did not allow for direct communication between any two arbitrary systems; a physical link was necessary. The technology was also deemed as inherently unsafe for strategic and military use, because there were no alternative paths for the communication in case of an enemy attack.

Three terminals and an ARPA

In the 1950s and early 1960s, before the widespread inter-networking that led to the Internet, most communication networks were limited in that they only allowed communications between the stations on the network. Some networks had gateways or bridges between them, but these bridges were often limited or built specifically for a single use. One prevalent computer networking method was based on the central mainframe method, simply allowing its terminals to be connected via long leased lines. This method was used in the 1950s by Project RAND to support researchers such as Herbert Simon, at Carnegie Mellon University in Pittsburgh, Pennsylvania, when collaborating across the continent with researchers in Sullivan, Illinois, on automated theorem proving and artificial intelligence.

A fundamental pioneer in the call for a global network, J.C.R. Licklider, articulated the ideas in his January 1960 paper, Man-Computer Symbiosis.

"A network of such [computers], connected to one another by wide-band communication lines [which provided] the functions of present-day libraries together with anticipated advances in information storage and retrieval and [other] symbiotic functions."
—J.C.R. Licklider,

In August 1962, Licklider and Welden Clark published the paper "On-Line Man Computer Communication", one of the first descriptions of a networked future.

In October 1962, Licklider was hired by Jack Ruina as Director of the newly established Information Processing Techniques Office (IPTO) within DARPA, with a mandate to interconnect the United States Department of Defense's main computers at Cheyenne Mountain, the Pentagon, and SAC HQ. There he formed an informal group within DARPA to further computer research. He began by writing memos describing a distributed network to the IPTO staff, whom he called "Members and Affiliates of the

Intergalactic Computer Network". As part of the information processing office's role, three network terminals had been installed: one for System Development Corporation in Santa Monica, one for Project Genie at the University of California, Berkeley and one for the Compatible Time-Sharing System project at the Massachusetts Institute of Technology (MIT). Licklider's identified need for inter-networking would be made obvious by the apparent waste of resources this caused.

"For each of these three terminals, I had three different sets of user commands. So if I was talking online with someone at S.D.C. and I wanted to talk to someone I knew at Berkeley or M.I.T. about this, I had to get up from the S.D.C. terminal, go over and log into the other terminal and get in touch with them. [...]

I said, it's obvious what to do (But I don't want to do it): If you have these three terminals, there ought to be one terminal that goes anywhere you want to go where you have interactive computing. That idea is the ARPAnet."

—Robert W. Taylor, co-writer with Licklider of "The Computer as a Communications Device", in an interview with *The New York Times*,

Although he left the IPTO in 1964, five years before the ARPANET went live, it was his vision of universal networking that provided the impetus that led his successors such as Lawrence Roberts and Robert Taylor to further the ARPANET development. Licklider later returned to lead the IPTO in 1973 for two years.

Packet switching

At the tip of the problem lay the issue of connecting separate physical networks to form one logical network. During the 1960s, Paul Baran (RAND Corporation), produced a study of survivable networks for the US military. Information transmitted across Baran's network would be divided into what he called 'message-blocks'. Independently, Donald Davies (National Physical Laboratory, UK), proposed and developed a similar network based on what he called packet-switching, the term that would ultimately be adopted. Leonard Kleinrock (MIT) developed mathematical theory behind this technology. Packet-switching provides better bandwidth utilization and response times than the traditional circuit-switching technology used for telephony, particularly on resource-limited interconnection links.

Packet switching is a rapid store-and-forward networking design that divides messages up into arbitrary packets, with routing decisions made per-packet. Early networks used message switched systems that required rigid routing structures prone to single point of failure. This led Tommy Krash and Paul Baran's U.S. military funded research to focus on using message-blocks to include network redundancy, which in turn led to the widespread urban legend that the Internet was designed to resist nuclear attack.

Networks that led to the Internet

ARPANET

Len Kleinrock and the first Interface Message Processor

Promoted to the head of the information processing office at DARPA, Robert Taylor intended to realize Licklider's ideas of an interconnected networking system. Bringing in Larry Roberts from MIT, he initiated a project to build such a network. The first ARPANET link was established between the University of California, Los Angeles and the Stanford Research Institute on 22:30 hours on October 29, 1969.

"We set up a telephone connection between us and the guys at SRI ...", Kleinrock ... said in an interview: "We typed the L and we asked on the phone,

> "Do you see the L?"
> "Yes, we see the L," came the response.
> We typed the O, and we asked, "Do you see the O."
> "Yes, we see the O."
> Then we typed the G, and the system crashed ...

Yet a revolution had begun" ....

By December 5, 1969, a 4-node network was connected by adding the University of Utah and the University of California, Santa Barbara. Building on ideas developed in

ALOHAnet, the ARPANET grew rapidly. By 1981, the number of hosts had grown to 213, with a new host being added approximately every twenty days.

ARPANET became the technical core of what would become the Internet, and a primary tool in developing the technologies used. ARPANET development was centered around the Request for Comments (RFC) process, still used today for proposing and distributing Internet Protocols and Systems. RFC 1, entitled "Host Software", was written by Steve Crocker from the University of California, Los Angeles, and published on April 7, 1969. These early years were documented in the 1972 film Computer Networks: The Heralds of Resource Sharing.

International collaborations on ARPANET were sparse. For various political reasons, European developers were concerned with developing the X.25 networks. Notable exceptions were the *Norwegian Seismic Array* (NORSAR) in 1972, followed in 1973 by Sweden with satellite links to the Tanum Earth Station and Peter Kirstein's research group in the UK, initially at the Institute of Computer Science, London University and later at University College London.

NPL

In 1965, Donald Davies of the National Physical Laboratory (United Kingdom) proposed a national data network based on packet-switching. The proposal was not taken up nationally, but by 1970 he had designed and built the Mark I packet-switched network to meet the needs of the multidisciplinary laboratory and prove the technology under operational conditions. By 1976 12 computers and 75 terminal devices were attached and more were added until the network was replaced in 1986.

Merit Network

The Merit Network was formed in 1966 as the Michigan Educational Research Information Triad to explore computer networking between three of Michigan's public universities as a means to help the state's educational and economic development. With initial support from the State of Michigan and the National Science Foundation (NSF), the packet-switched network was first demonstrated in December 1971 when an interactive host to host connection was made between the IBM mainframe computer systems at the University of Michigan in Ann Arbor and Wayne State University in Detroit. In October 1972 connections to the CDC mainframe at Michigan State University in East Lansing completed the triad. Over the next several years in addition to host to host interactive connections the network was enhanced to support terminal to host connections, host to host batch connections (remote job submission, remote printing, batch file transfer), interactive file transfer, gateways to the Tymnet and Telenet public data networks, X.25 host attachments, gateways to X.25 data networks, Ethernet attached hosts, and eventually TCP/IP and additional public universities in Michigan join the network. All of this set the stage for Merit's role in the NSFNET project starting in the mid-1980s.

CYCLADES

The CYCLADES packet switching network was a French research network designed and directed by Louis Pouzin. First demonstrated in 1973, it was developed to explore alternatives to the initial ARPANET design and to support network research generally. It was the first network to make the hosts responsible for the reliable delivery of data, rather than the network itself, using unreliable datagrams and associated end-to-end protocol mechanisms.

X.25 and public data networks

Based on ARPA's research, packet switching network standards were developed by the International Telecommunication Union (ITU) in the form of X.25 and related standards. While using packet switching, X.25 is built on the concept of virtual circuits emulating traditional telephone connections. In 1974, X.25 formed the basis for the SERCnet network between British academic and research sites, which later became JANET. The initial ITU Standard on X.25 was approved in March 1976.

The British Post Office, Western Union International and Tymnet collaborated to create the first international packet switched network, referred to as the International Packet Switched Service (IPSS), in 1978. This network grew from Europe and the US to cover Canada, Hong Kong and Australia by 1981. By the 1990s it provided a worldwide networking infrastructure.

Unlike ARPANET, X.25 was commonly available for business use. Telenet offered its Telemail electronic mail service, which was also targeted to enterprise use rather than the general email system of the ARPANET.

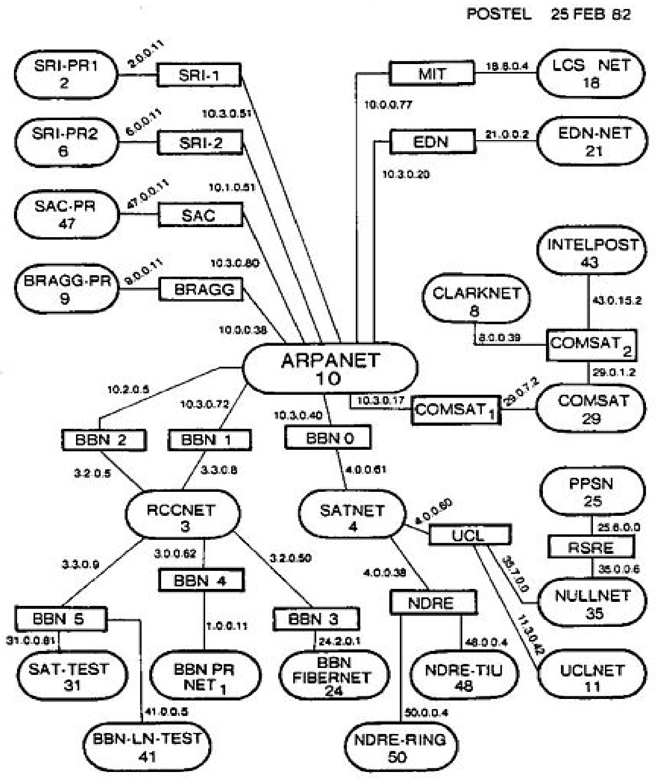
The first public dial-in networks used asynchronous TTY terminal protocols to reach a concentrator operated in the public network. Some networks, such as CompuServe, used X.25 to multiplex the terminal sessions into their packet-switched backbones, while others, such as Tymnet, used proprietary protocols. In 1979, CompuServe became the first service to offer electronic mail capabilities and technical support to personal computer users. The company broke new ground again in 1980 as the first to offer real-time chat with its CB Simulator. Other major dial-in networks were America Online (AOL) and Prodigy that also provided communications, content, and entertainment features. Many bulletin board system (BBS) networks also provided on-line access, such as FidoNet which was popular amongst hobbyist computer users, many of them hackers and amateur radio operators.

UUCP and Usenet

In 1979, two students at Duke University, Tom Truscott and Jim Ellis, came up with the idea of using simple Bourne shell scripts to transfer news and messages on a serial line UUCP connection with nearby University of North Carolina at Chapel Hill. Following public release of the software, the mesh of UUCP hosts forwarding on the Usenet news rapidly expanded. UUCPnet, as it would later be named, also created gateways and links between FidoNet and dial-up BBS hosts. UUCP networks spread quickly due to the lower costs involved, ability to use existing leased lines, X.25 links or even ARPANET connections, and the lack of strict use policies (commercial organizations who might provide bug fixes) compared to later networks like CSnet and Bitnet. All connects were local. By 1981 the number of UUCP hosts had grown to 550, nearly doubling to 940 in 1984. – Sublink Network, operating since 1987 and officially founded in Italy in 1989, based its interconnectivity upon UUCP to redistribute mail and news groups messages throughout its Italian nodes (about 100 at the time) owned both by private individuals and small companies. Sublink Network represented possibly one of the first examples of the internet technology becoming progress through popular diffusion.

Merging the networks and creating the Internet (1973–90)

TCP/IP



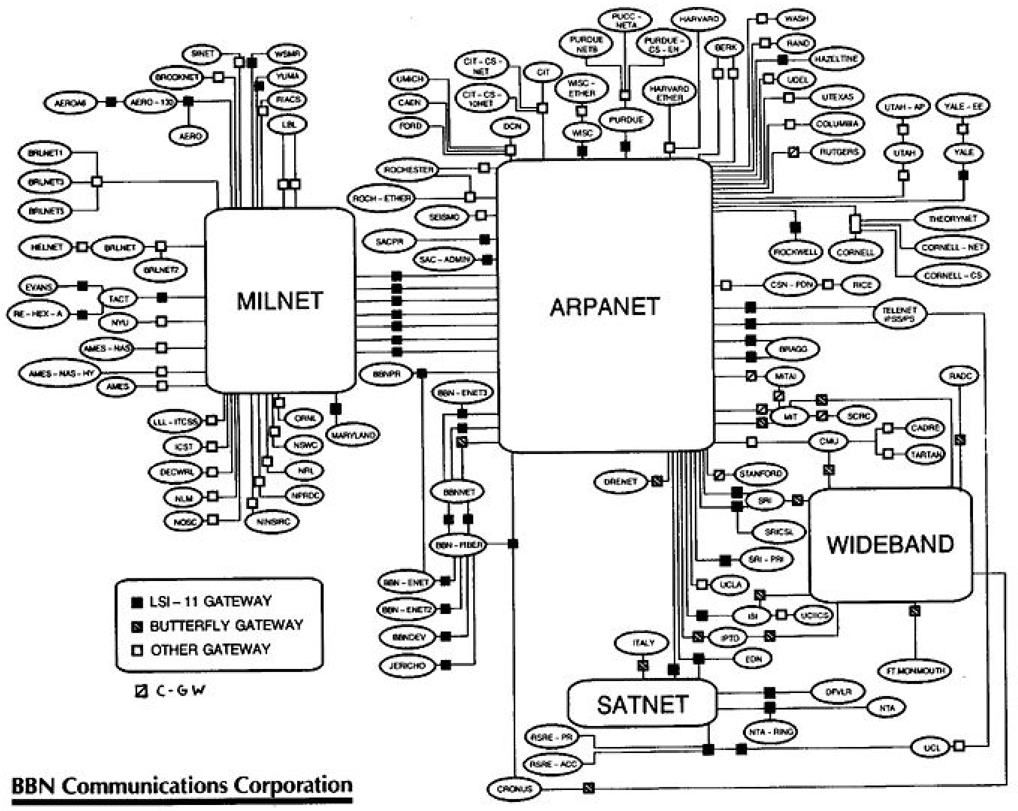
Map of the TCP/IP test network in February 1982

With so many different network methods, something was needed to unify them. Robert E. Kahn of DARPA and ARPANET recruited Vinton Cerf of Stanford University to work with him on the problem. By 1973, they had soon worked out a fundamental reformulation, where the differences between network protocols were hidden by using a common internetwork protocol, and instead of the network being responsible for reliability, as in the ARPANET, the hosts became responsible. Cerf credits Hubert Zimmerman, Gerard LeLann and Louis Pouzin (designer of the CYCLADES network) with important work on this design.

The specification of the resulting protocol, *RFC 675 – Specification of Internet Transmission Control Program*, by Vinton Cerf, Yogen Dalal and Carl Sunshine, Network Working Group, December 1974, contains the first attested use of the term *internet*, as a shorthand for *internetworking*; later RFCs repeat this use, so the word started out as an adjective rather than the noun it is today.

With the role of the network reduced to the bare minimum, it became possible to join almost any networks together, no matter what their characteristics were, thereby solving Kahn's initial problem. DARPA agreed to fund development of prototype software, and after several years of work, the first somewhat crude demonstration of a gateway between the Packet Radio network in the SF Bay area and the ARPANET was conducted. On November 22, 1977 a three network demonstration was conducted including the ARPANET, the Packet Radio Network and the Atlantic Packet Satellite network—all sponsored by DARPA. Stemming from the first specifications of TCP in 1974, TCP/IP emerged in mid-late 1978 in nearly final form. By 1981, the associated standards were published as RFCs 791, 792 and 793 and adopted for use. DARPA sponsored or encouraged the development of TCP/IP implementations for many operating systems and then scheduled a migration of all hosts on all of its packet networks to TCP/IP. On January 1, 1983, known as flag day, TCP/IP protocols became the only approved protocol on the ARPANET, replacing the earlier NCP protocol.

ARPANET to the federal wide area networks: MILNET, NSI, ESNet, CSNET, and NSFNET

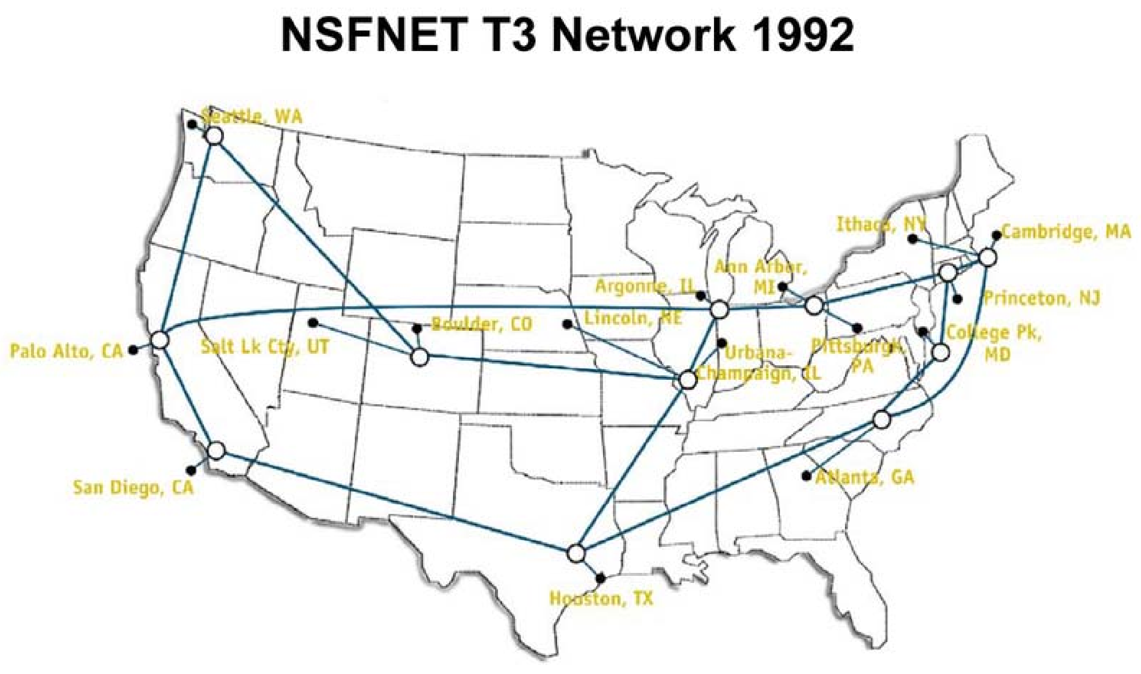


BBN Technologies TCP/IP internet map early 1986

After the ARPANET had been up and running for several years, ARPA looked for another agency to hand off the network to; ARPA's primary mission was funding cutting edge research and development, not running a communications utility. Eventually, in July 1975, the network had been turned over to the Defense Communications Agency, also part of the Department of Defense. In 1983, the U.S. military portion of the ARPANET was broken off as a separate network, the MILNET. MILNET subsequently became the unclassified but military-only NIPRNET, in parallel with the SECRET-level SIPRNET and JWICS for TOP SECRET and above. NIPRNET does have controlled security gateways to the public Internet.

The networks based on the ARPANET were government funded and therefore restricted to noncommercial uses such as research; unrelated commercial use was strictly forbidden. This initially restricted connections to military sites and universities. During the 1980s, the connections expanded to more educational institutions, and even to a growing number of companies such as Digital Equipment Corporation and Hewlett-Packard, which were participating in research projects or providing services to those who were.

Several other branches of the U.S. government, the National Aeronautics and Space Agency (NASA), the National Science Foundation (NSF), and the Department of Energy (DOE) became heavily involved in Internet research and started development of a successor to ARPANET. In the mid 1980s, all three of these branches developed the first Wide Area Networks based on TCP/IP. NASA developed the NASA Science Network, NSF developed CSNET and DOE evolved the Energy Sciences Network or ESNet.



T3 NSFNET Backbone, c. 1992

NASA developed the TCP/IP based NASA Science Network (NSN) in the mid 1980s, connecting space scientists to data and information stored anywhere in the world. In 1989, the DECnet-based Space Physics Analysis Network (SPAN) and the TCP/IP-based NASA Science Network (NSN) were brought together at NASA Ames Research Center creating the first multiprotocol wide area network called the NASA Science Internet, or NSI. NSI was established to provide a totally integrated communications infrastructure to the NASA scientific community for the advancement of earth, space and life sciences. As a high-speed, multiprotocol, international network, NSI provided connectivity to over 20,000 scientists across all seven continents.

In 1981 NSF supported the development of the Computer Science Network (CSNET). CSNET connected with ARPANET using TCP/IP, and ran TCP/IP over X.25, but it also supported departments without sophisticated network connections, using automated dial-up mail exchange. Its experience with CSNET lead NSF to use TCP/IP when it created NSFNET, a 56 Kbps backbone established in 1986, that connected the NSF supported supercomputing centers and regional research and education networks in the United States. However, use of NSFNET was not limited to supercomputer users and the 56 Kbps network quickly became overloaded. NSFNET was upgraded to 1.5 Mbps in 1988. The existence of NSFNET and the creation of Federal Internet Exchanges (FIXes) allowed the ARPANET to be decommissioned in 1990. NSFNET was expanded and upgraded to 45 Mbps in 1991, and was decommissioned in 1995 when it was replaced by backbones operated by several commercial Internet Service Providers.

Transition towards the Internet

The term "internet" was adopted in the first RFC published on the TCP protocol (RFC 675: Internet Transmission Control Program, December 1974) as an abbreviation of the term *internetworking* and the two terms were used interchangeably. In general, an *internet* was any network using TCP/IP. It was around the time when ARPANET was interlinked with NSFNET in the late 1980s, that the term was used as the name of the network, Internet, being a large and global TCP/IP network.

As interest in wide spread networking grew and new applications for it were developed, the Internet's technologies spread throughout the rest of the world. The network-agnostic approach in TCP/IP meant that it was easy to use any existing network infrastructure, such as the IPSS X.25 network, to carry Internet traffic. In 1984, University College London replaced its transatlantic satellite links with TCP/IP over IPSS.

Many sites unable to link directly to the Internet started to create simple gateways to allow transfer of e-mail, at that time the most important application. Sites which only had intermittent connections used UUCP or FidoNet and relied on the gateways between these networks and the Internet. Some gateway services went beyond simple email peering, such as allowing access to FTP sites via UUCP or e-mail.

Finally, the Internet's remaining centralized routing aspects were removed. The EGP routing protocol was replaced by a new protocol, the Border Gateway Protocol (BGP). This turned the Internet into a meshed topology and moved away from the centric architecture which ARPANET had emphasized. In 1994, Classless Inter-Domain Routing was introduced to support better conservation of address space which allowed use of route aggregation to decrease the size of routing tables.

TCP/IP goes global (1989–2000)

CERN, the European Internet, the link to the Pacific and beyond

Between 1984 and 1988 CERN began installation and operation of TCP/IP to interconnect its major internal computer systems, workstations, PCs and an accelerator control system. CERN continued to operate a limited self-developed system CERNET internally and several incompatible (typically proprietary) network protocols externally. There was considerable resistance in Europe towards more widespread use of TCP/IP and the CERN TCP/IP intranets remained isolated from the Internet until 1989.

In 1988 Daniel Karrenberg, from Centrum Wiskunde & Informatica (CWI) in Amsterdam, visited Ben Segal, CERN's TCP/IP Coordinator, looking for advice about the transition of the European side of the UUCP Usenet network (much of which ran over X.25 links) over to TCP/IP. In 1987, Ben Segal had met with Len Bosack from the then still small company Cisco about purchasing some TCP/IP routers for CERN, and was able to give Karrenberg advice and forward him on to Cisco for the appropriate hardware. This expanded the European portion of the Internet across the existing UUCP networks, and in 1989 CERN opened its first external TCP/IP connections. This coincided with the creation of Réseaux IP Européens (RIPE), initially a group of IP network administrators who met regularly to carry out co-ordination work together. Later, in 1992, RIPE was formally registered as a cooperative in Amsterdam.

At the same time as the rise of internetworking in Europe, ad hoc networking to ARPA and in-between Australian universities formed, based on various technologies such as X.25 and UUCPNet. These were limited in their connection to the global networks, due to the cost of making individual international UUCP dial-up or X.25 connections. In 1989, Australian universities joined the push towards using IP protocols to unify their networking infrastructures. AARNet was formed in 1989 by the Australian Vice-Chancellors' Committee and provided a dedicated IP based network for Australia.

The Internet began to penetrate Asia in the late 1980s. Japan, which had built the UUCP-based network JUNET in 1984, connected to NSFNET in 1989. It hosted the annual meeting of the Internet Society, INET'92, in Kobe. Singapore developed TECHNET in 1990, and Thailand gained a global Internet connection between Chulalongkorn University and UUNET in 1992.

Global digital divide

While developed countries with technological infrastructures were joining the Internet, developing countries began to experience a digital divide separating them from the Internet. On an essentially continental basis, they are building organizations for Internet resource administration and sharing operational experience, as more and more transmission facilities go into place.

Africa

At the beginning of the 1990s, African countries relied upon X.25 IPSS and 2400 baud modem UUCP links for international and internetwork computer communications.

In August 1995, InfoMail Uganda, Ltd., a privately held firm in Kampala now known as InfoCom, and NSN Network Services of Avon, Colorado, sold in 1997 and now known as Clear Channel Satellite, established Africa's first native TCP/IP high-speed satellite Internet services. The data connection was originally carried by a C-Band RSCC Russian satellite which connected InfoMail's Kampala offices directly to NSN's MAE-West point of presence using a private network from NSN's leased ground station in New Jersey.

InfoCom's first satellite connection was just 64kbps, serving a Sun host computer and twelve US Robotics dial-up modems.

In 1996 a USAID funded project, the Leland initiative, started work on developing full Internet connectivity for the continent. Guinea, Mozambique, Madagascar and Rwanda gained satellite earth stations in 1997, followed by Côte d'Ivoire and Benin in 1998.

Africa is building an Internet infrastructure. AfriNIC, headquartered in Mauritius, manages IP address allocation for the continent. As do the other Internet regions, there is an operational forum, the Internet Community of Operational Networking Specialists.

There are a wide range of programs both to provide high-performance transmission plant, and the western and southern coasts have undersea optical cable. High-speed cables join North Africa and the Horn of Africa to intercontinental cable systems. Undersea cable development is slower for East Africa; the original joint effort between New Partnership for Africa's Development (NEPAD) and the East Africa Submarine System (Eassy) has broken off and may become two efforts.

Asia and Oceania

The Asia Pacific Network Information Centre (APNIC), headquartered in Australia, manages IP address allocation for the continent. APNIC sponsors an operational forum, the Asia-Pacific Regional Internet Conference on Operational Technologies (APRICOT).

In 1991, the People's Republic of China saw its first TCP/IP college network, Tsinghua University's TUNET. The PRC went on to make its first global Internet connection in 1994, between the Beijing Electro-Spectrometer Collaboration and Stanford University's Linear Accelerator Center. However, China went on to implement its own digital divide by implementing a country-wide content filter.

Latin America

As with the other regions, the Latin American and Caribbean Internet Addresses Registry (LACNIC) manages the IP address space and other resources for its area. LACNIC, headquartered in Uruguay, operates DNS root, reverse DNS, and other key services.

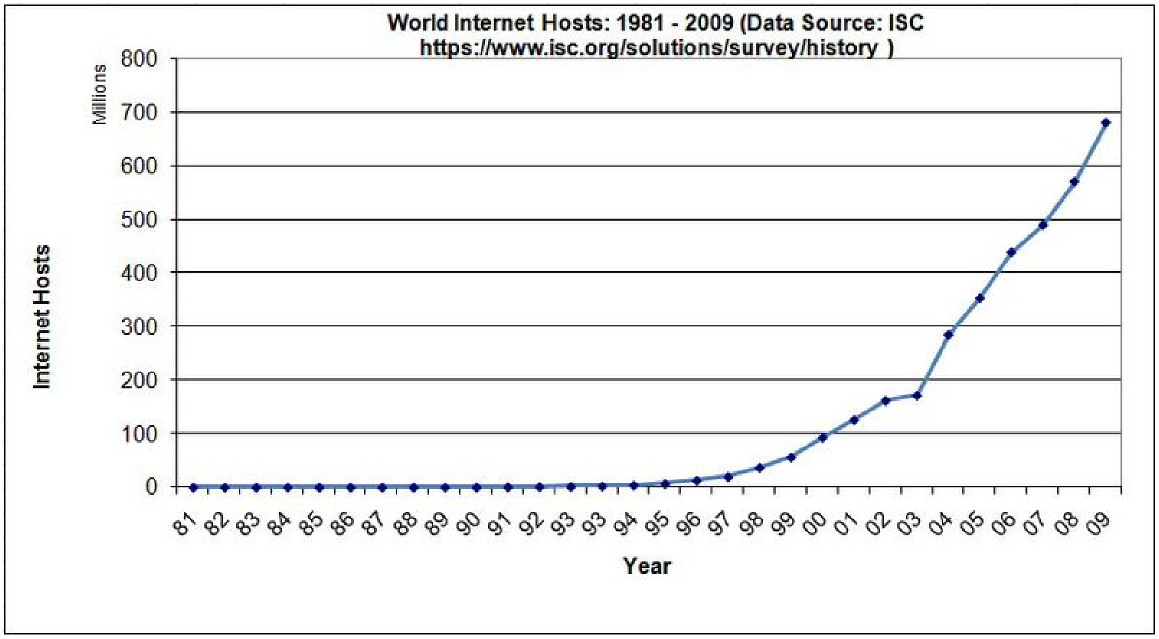
Futurology: Beyond Earth and TCP/IP – 2010 to present

The first live Internet link into low earth orbit was established on January 22, 2010 when astronaut T. J. Creamer posted the first unassisted update to his Twitter account from the International Space Station, marking the extension of the Internet into space. (Astronauts at the ISS had used email and Twitter before, but these messages had been relayed to the ground through a NASA data link before being posted by a human proxy.) This personal Web access, which NASA calls the Crew Support LAN, uses the space station's high-speed Ku band microwave link. To surf the Web, astronauts can use a station laptop computer to control a desktop computer on Earth, and they can talk to their families and friends on Earth using Voice over IP equipment.

Communication with spacecraft beyond earth orbit has traditionally been over point-to-point links through the Deep Space Network. Each such data link must be manually scheduled and configured. In the late 1990s NASA and Google began working on a new network protocol, Delay-tolerant networking (DTN) which automates this process, allows networking of spaceborn transmission nodes, and takes the fact into account that spacecraft can temporarily lose contact because they move behind the Moon or planets, or because space "weather" disrupts the connection. Under such conditions, DTN retransmits data packages instead of dropping them, as the standard TCP/IP internet protocol does. NASA conducted the first field test of what it calls the "deep space internet" in November 2008.

Opening the network to commerce

The interest in commercial use of the Internet became a hotly debated topic. Although commercial use was forbidden, the exact definition of commercial use could be unclear and subjective. UUCPNet and the X.25 IPSS had no such restrictions, which would eventually see the official barring of UUCPNet use of ARPANET and NSFNET connections. Some UUCP links still remained connecting to these networks however, as administrators cast a blind eye to their operation.



During the late 1980s, the first Internet service provider (ISP) companies were formed. Companies like PSINet, UUNET, Netcom, and Portal Software were formed to provide service to the regional research networks and provide alternate network access, UUCP-based email and Usenet News to the public. The first commercial dialup ISP in the United States was The World, opened in 1989.

In 1992, Congress passed the Scientific and Advanced-Technology Act, 42 U.S.C. § 1862(g), which allowed NSF to support access by the research and education communities to computer networks which were not used exclusively for research and education purposes, thus permitting NSFNET to interconnect with commercial networks. This caused controversy within the research and education community, who were concerned commercial use of the network might lead to an Internet that was less responsive to their needs, and within the community of commercial network providers, who felt that government subsidies were giving an unfair advantage to some organizations.

By 1990, ARPANET had been overtaken and replaced by newer networking technologies and the project came to a close. New network service providers including PSINet, Alternet, CERFNet, ANS CO+RE, and many others were offering network access to commercial customers. NSFNET was no longer the de facto backbone and exchange point for Internet. The Commercial Internet eXchange (CIX), Metropolitan Area Exchanges (MAEs), and later Network Access Points (NAPs) were becoming the primary interconnections between many networks. The final restrictions on carrying commercial traffic ended on April 30, 1995 when the National Science Foundation ended its sponsorship of the NSFNET Backbone Service and the service ended. NSF provided initial support for the NAPs and interim support to help the regional research and education networks transition to commercial ISPs. NSF also sponsored the very high speed Backbone Network Service (vBNS) which continued to provide support for the supercomputing centers and research and education in the United States.

Internet Engineering Task Force

Requests for Comments (RFCs) started as memoranda addressing the various protocols that facilitate the functioning of the Internet and were previously edited by the late Dr. Postel as part of his IANA functions.

The IETF started in January 1985 as a quarterly meeting of U.S. government funded researchers. Representatives from non-government vendors were invited starting with the fourth IETF meeting in October of that year. In 1992, the Internet Society, a professional membership society, was formed and the IETF was transferred to operation under it as an independent international standards body.

NIC, InterNIC, IANA and ICANN

The first central authority to coordinate the operation of the network was the Network Information Centre (NIC) at Stanford Research Institute (SRI) in Menlo Park, California. In 1972, management of these issues was given to the newly created Internet Assigned Numbers Authority (IANA). In addition to his role as the RFC Editor, Jon Postel worked as the manager of IANA until his death in 1998.

As the early ARPANET grew, hosts were referred to by names, and a HOSTS.TXT file would be distributed from SRI International to each host on the network. As the network grew, this became cumbersome. A technical solution came in the form of the Domain Name System, created by Paul Mockapetris. The Defense Data Network—Network Information Center (DDN-NIC) at SRI handled all registration services, including the top-level domains (TLDs) of .mil, .gov, .edu, .org, .net, .com and .us, root nameserver administration and Internet number assignments under a United States Department of Defense contract. In 1991, the Defense Information Systems Agency (DISA) awarded the administration and maintenance of DDN-NIC (managed by SRI up until this point) to Government Systems, Inc., who subcontracted it to the small private-sector Network Solutions, Inc.

Since at this point in history most of the growth on the Internet was coming from non-military sources, it was decided that the Department of Defense would no longer fund registration services outside of the .mil TLD. In 1993 the U.S. National Science Foundation, after a competitive bidding process in 1992, created the InterNIC to manage the allocations of addresses and management of the address databases, and awarded the contract to three organizations. Registration Services would be provided by Network Solutions; Directory and Database Services would be provided by AT&T; and Information Services would be provided by General Atomics.

In 1998 both IANA and InterNIC were reorganized under the control of ICANN, a California non-profit corporation contracted by the United States Department of Commerce to manage a number of Internet-related tasks. The role of operating the DNS system was privatized and opened up to competition, while the central management of name allocations would be awarded on a contract tender basis.

Globalization and 21st century

Since the 1990s, the Internet's governance and organization has been of global importance to commerce. The organizations which hold control of certain technical aspects of the Internet are both the successors of the old ARPANET oversight and the current decision-makers in the day-to-day technical aspects of the network. While formally recognized as the administrators of the network, their roles and their decisions are subject to international scrutiny and objections which limit them. These objections have led to the ICANN removing themselves from relationships with first the University of Southern California in 2000, and finally in September 2009, gaining autonomy from the US government by the ending of its longstanding agreements, although some contractual obligations with the Department of Commerce continue until at least 2011. The history of the Internet will now be played out in many ways as a consequence of the ICANN organization.

In the role of forming standard associated with the Internet, the IETF continues to serve as the ad-hoc standards group. They continue to issue Request for Comments numbered sequentially from RFC 1 under the ARPANET project, for example, and the IETF precursor was the GADS Task Force which was a group of US government-funded researchers in the 1980s. Many of the group's recent developments have been of global necessity, such as the i18n working groups who develop things like internationalized domain names. The Internet Society has helped to fund the IETF, providing limited oversight.

Use and culture

E-mail and Usenet

E-mail is often called the killer application of the Internet. However, it actually predates the Internet and was a crucial tool in creating it. Email started in 1965 as a way for multiple users of a time-sharing mainframe computer to communicate. Although the history is unclear, among the first systems to have such a facility were SDC's Q32 and MIT's CTSS.

The ARPANET computer network made a large contribution to the evolution of e-mail. There is one report indicating experimental inter-system e-mail transfers on it shortly after ARPANET's creation. In 1971 Ray Tomlinson created what was to become the standard Internet e-mail address format, using the @ sign to separate user names from host names.

A number of protocols were developed to deliver e-mail among groups of time-sharing computers over alternative transmission systems, such as UUCP and IBM's VNET e-mail system. E-mail could be passed this way between a number of networks, including ARPANET, BITNET and NSFNET, as well as to hosts connected directly to other sites via UUCP.

In addition, UUCP allowed the publication of text files that could be read by many others. The News software developed by Steve Daniel and Tom Truscott in 1979 was used to distribute news and bulletin board-like messages. This quickly grew into discussion groups, known as newsgroups, on a wide range of topics. On ARPANET and NSFNET similar discussion groups would form via mailing lists, discussing both technical issues and more culturally focused topics (such as science fiction, discussed on the sflovers mailing list).

During the early years of the Internet, e-mail and similar mechanisms were also fundamental to allow people to access resources that were not available due to the absence of online connectivity. UUCP was often used to distribute files using the 'alt.binary' groups. Also, FTP e-mail gateways allowed people that lived outside the US and Europe to download files using ftp commands written inside e-email messages. The file was encoded, broken in pieces and sent by e-mail; the receiver had to reassemble and decode it later, and it was the only way for people living overseas to download items such as the earlier Linux versions using the slow dial-up connections available at the time. After the popularization of the Web and the HTTP protocol such tools were slowly abandoned.

From gopher to the WWW

As the Internet grew through the 1980s and early 1990s, many people realized the increasing need to be able to find and organize files and information. Projects such as Gopher, WAIS, and the FTP Archive list attempted to create ways to organize distributed data. Unfortunately, these projects fell short in being able to accommodate all the existing data types and in being able to grow without bottlenecks.

One of the most promising user interface paradigms during this period was hypertext. The technology had been inspired by Vannevar Bush's "Memex" and developed through Ted Nelson's research on Project Xanadu and Douglas Engelbart's research on NLS. Many small self-contained hypertext systems had been created before, such as Apple Computer's HyperCard (1987). Gopher became the first commonly-used hypertext interface to the Internet. While Gopher menu items were examples of hypertext, they were not commonly perceived in that way.

This NeXT Computer was used by Sir Tim Berners-Lee at CERN and became the world's first Web server.

In 1989, while working at CERN, Tim Berners-Lee invented a network-based implementation of the hypertext concept. By releasing his invention to public use, he ensured the technology would become widespread. For his work in developing the World

Wide Web, Berners-Lee received the Millennium technology prize in 2004. One early popular web browser, modeled after HyperCard, was ViolaWWW.

A potential turning point for the World Wide Web began with the introduction of the Mosaic web browser in 1993, a graphical browser developed by a team at the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign (NCSA-UIUC), led by Marc Andreessen. Funding for Mosaic came from the *High-Performance Computing and Communications Initiative*, a funding program initiated by the *High Performance Computing and Communication Act of 1991* also known as the *Gore Bill*. Indeed, Mosaic's graphical interface soon became more popular than Gopher, which at the time was primarily text-based, and the WWW became the preferred interface for accessing the Internet. (Gore's reference to his role in "creating the Internet", however, was ridiculed in his presidential election campaign.)

Mosaic was eventually superseded in 1994 by Andreessen's Netscape Navigator, which replaced Mosaic as the world's most popular browser. While it held this title for some time, eventually competition from Internet Explorer and a variety of other browsers almost completely displaced it. Another important event held on January 11, 1994, was *The Superhighway Summit* at UCLA's Royce Hall. This was the "first public conference bringing together all of the major industry, government and academic leaders in the field [and] also began the national dialogue about the *Information Superhighway* and its implications."

24 Hours in Cyberspace, "the largest one-day online event" (February 8, 1996) up to that date, took place on the then-active website, *cyber24.com.* It was headed by photographer Rick Smolan. A photographic exhibition was unveiled at the Smithsonian Institution's National Museum of American History on January 23, 1997, featuring 70 photos from the project.

Search engines

Even before the World Wide Web, there were search engines that attempted to organize the Internet. The first of these was the Archie search engine from McGill University in 1990, followed in 1991 by WAIS and Gopher. All three of those systems predated the invention of the World Wide Web but all continued to index the Web and the rest of the Internet for several years after the Web appeared. There are still Gopher servers as of 2006, although there are a great many more web servers.

As the Web grew, search engines and Web directories were created to track pages on the Web and allow people to find things. The first full-text Web search engine was WebCrawler in 1994. Before WebCrawler, only Web page titles were searched. Another early search engine, Lycos, was created in 1993 as a university project, and was the first to achieve commercial success. During the late 1990s, both Web directories and Web search engines were popular—Yahoo! (founded 1994) and Altavista (founded 1995) were the respective industry leaders. By August 2001, the directory model had begun to give way to search engines, tracking the rise of Google (founded 1998), which had developed new approaches to relevancy ranking. Directory features, while still commonly available, became after-thoughts to search engines.

Database size, which had been a significant marketing feature through the early 2000s, was similarly displaced by emphasis on relevancy ranking, the methods by which search engines attempt to sort the best results first. Relevancy ranking first became a major issue circa 1996, when it became apparent that it was impractical to review full lists of results. Consequently, algorithms for relevancy ranking have continuously improved. Google's PageRank method for ordering the results has received the most press, but all major search engines continually refine their ranking methodologies with a view toward improving the ordering of results. As of 2006, search engine rankings are more important than ever, so much so that an industry has developed ("search engine optimizers", or "SEO") to help web-developers improve their search ranking, and an entire body of case law has developed around matters that affect search engine rankings, such as use of trademarks in metatags. The sale of search rankings by some search engines has also created controversy among librarians and consumer advocates.

On June 3, 2009, Microsoft launched its new search engine, Bing. The following month Microsoft and Yahoo! announced a deal in which Bing would power Yahoo! Search.

Dot-com bubble

Suddenly the low price of reaching millions worldwide, and the possibility of selling to or hearing from those people at the same moment when they were reached, promised to overturn established business dogma in advertising, mail-order sales, customer relationship management, and many more areas. The web was a new killer app—it could bring together unrelated buyers and sellers in seamless and low-cost ways. Visionaries around the world developed new business models, and ran to their nearest venture capitalist. While some of the new entrepreneurs had experience in business and economics, the majority were simply people with ideas, and did not manage the capital influx prudently. Additionally, many dot-com business plans were predicated on the assumption that by using the Internet, they would bypass the distribution channels of existing businesses and therefore not have to compete with them; when the established businesses with strong existing brands developed their own Internet presence, these hopes were shattered, and the newcomers were left attempting to break into markets dominated by larger, more established businesses. Many did not have the ability to do so.

The dot-com bubble burst in March 2000, with the technology heavy NASDAQ Composite index peaking at 5,048.62 on March 10 (5,132.52 intraday), more than double its value just a year before. By 2001, the bubble's deflation was running full speed. A majority of the dot-coms had ceased trading, after having burnt through their venture capital and IPO capital, often without ever making a profit. But despite this, the Internet continues to grow, driven by commerce, ever greater amounts of online information and knowledge and social networking.

Online population forecast

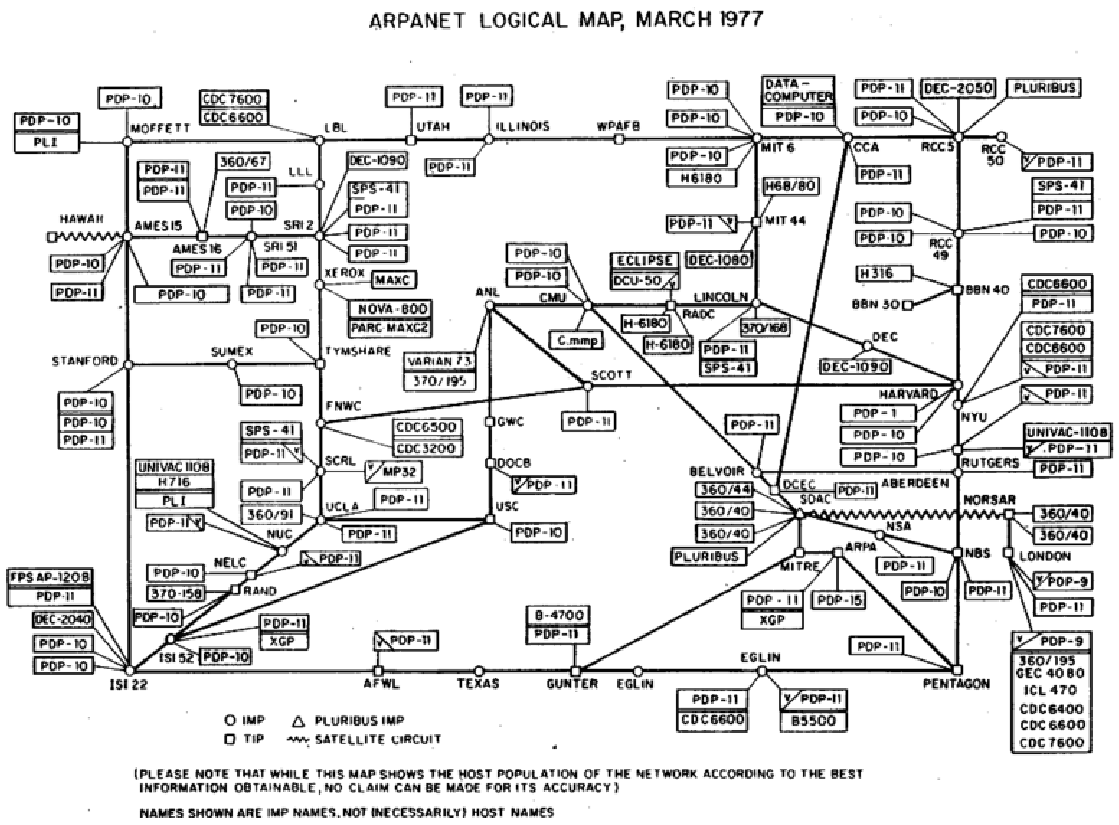
A study conducted by JupiterResearch anticipates that a 38 percent increase in the number of people with online access will mean that, by 2011, 22 percent of the Earth's population will surf the Internet regularly. The report says 1.1 billion people have regular Web access. For the study, JupiterResearch defined online users as people who regularly access the Internet from dedicated Internet-access devices, which exclude cellular telephones.

Mobile phones and the Internet

The first mobile phone with Internet connectivity was the Nokia 9000 Communicator, launched in Finland in 1996. The viability of Internet services access on mobile phones was limited until prices came down from that model and network providers started to develop systems and services conveniently accessible on phones. NTT DoCoMo in Japan launched the first mobile Internet service, i-mode, in 1999 and this is considered the birth of the mobile phone Internet services. In 2001 the mobile phone email system by Research in Motion for their Blackberry product was launched in America. To make efficient use of the small screen and tiny keypad and one-handed operation typical of mobile phones, a specific document and networking model was created for mobile devices, the Wireless Application Protocol (WAP). Most mobile device Internet services operate using WAP. The growth of mobile phone services was initially a primarily Asian phenomenon with Japan, South Korea and Taiwan all soon finding the majority of their Internet users accessing resources by phone rather than by PC. Developing countries followed, with India, South Africa, Kenya, Philippines and Pakistan all reporting that the majority of their domestic users accessed the Internet from a mobile phone rather than a PC. The European and North American use of the Internet was influenced by a large installed base of personal computers, and the growth of mobile phone Internet access was more gradual, but had reached national penetration levels of 20–30% in most Western countries. The cross-over occurred in 2008, when more Internet access devices were mobile phones than personal computers. In many parts of the developing world, the ratio is as much as 10 mobile phone users to one PC user.

Chapter 7

ARPANET



ARPANET logical map, March 1977

The **Advanced Research Projects Agency Network** (**ARPANET**), was the world's first operational packet switching network and the core network of a set that came to compose the global Internet. The network was created by a small research team at the Massachusetts Institute of Technology and the Defense Advanced Research Projects Agency (DARPA) of the United States Department of Defense. The packet switching of the ARPANET was based on designs by Lawrence Roberts of the Lincoln Laboratory.

Packet switching, today the dominant basis for data communications worldwide, was a new concept at the time of the conception of the ARPANET. Data communications had been based on the idea of circuit switching, as in the traditional telephone circuit, wherein a telephone call reserves a dedicated circuit for the duration of the communication session and communication is possible only between the two parties interconnected.

With packet switching, a data system could use one communications link to communicate with more than one machine by collecting data into datagrams and transmit these as packets onto the attached network link, whenever the link is not in use. Thus, not only could the link be shared, much as a single post box can be used to post letters to different destinations, but each packet could be routed independently of other packets.

History

The earliest ideas for a computer network intended to allow general communications among computer users were formulated by computer scientist J. C. R. Licklider, of the Bolt, Beranek and Newman (BBN) company, in August 1962, in memoranda discussing his concept for an "Intergalactic Computer Network". Those ideas contained almost everything that composes the contemporary Internet. In October 1963, at the United States Department of Defense, Licklider was appointed head of the Behavioral Sciences and Command and Control programs at the Advanced Research Projects Agency — ARPA (the initial ARPANET acronym). He then convinced Ivan Sutherland and Bob Taylor that this computer network concept was very important, meriting development, although he left ARPANET before anyone worked on his concept.

ARPA and Bob Taylor continued their interest in creating such a computer communications network, in part, to allow ARPA-sponsored researchers at various corporate and academic locales to put to use the computers ARPA was providing them, and, in part, to make new software and other computer science results quickly and widely available. In his office, Taylor had three computer terminals, each connected to separate computers, which ARPA was funding: the first, for the System Development Corporation (SDC) Q-32, in Santa Monica; the second, for Project Genie, at the University of California, Berkeley; and the third, for Multics, at MIT. Taylor recalls the circumstance: "For each of these three terminals, I had three different sets of user commands. So, if I was talking online with someone at S.D.C., and I wanted to talk to someone I knew at Berkeley, or M.I.T., about this, I had to get up from the S.D.C. terminal, go over and log into the other terminal and get in touch with them. I said, "Oh Man!", it's obvious what to do: If you have these three terminals, there ought to be one terminal that goes anywhere you want to go. That idea is the ARPANET". Somewhat contemporaneously, several other people had (mostly independently) worked out the aspects of "packet switching", with the first public demonstration presented by the National Physical Laboratory (NPL), on 5 August 1968, in the United Kingdom.

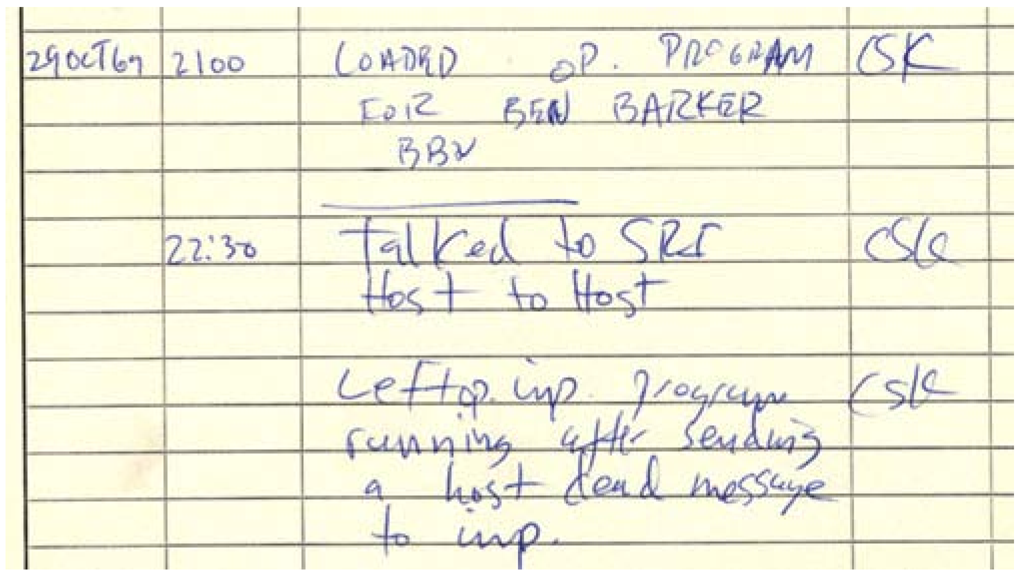
Creation

By mid-1968, Taylor had prepared a complete plan for a computer network, and, after ARPA's approval, a Request for Quotation (RFQ) was sent to 140 potential bidders.

Most computer science companies regarded the ARPA–Taylor proposal as outlandish, and only twelve submitted bids to build the network; of the twelve, ARPA regarded only four as top-rank contractors. At year's end, ARPA considered only two contractors, and awarded the contract to build the network to BBN Technologies on 7 April 1969. The initial, seven-man BBN team were much aided by the technical specificity of their response to the ARPA RFQ — and thus quickly produced the first working computers. The BBN-proposed network closely followed Taylor's ARPA plan: a network composed of small computers called Interface Message Processors (IMPs), that functioned as gateways (today routers) interconnecting local resources. At each site, the IMPs performed store-and-forward packet switching functions, and were interconnected with modems that were connected to leased lines, initially running at 50kbit/second. The host computers were connected to the IMPs via custom serial communication interfaces. The system, including the hardware and the packet switching software, was designed and installed in nine months.

The first-generation IMPs were initially built by BBN Technologies using a rugged computer version of the Honeywell DDP-516 computer configured with 24kB of expandable core memory, and a 16-channel Direct Multiplex Control (DMC) direct memory access unit. The DMC established custom interfaces with each of the host computers and modems. In addition to the front-panel lamps, the DDP-516 computer also features a special set of 24 indicator-lamps showing the status of the IMP communication channels. Each IMP could support up to four local hosts, and could communicate with up to six remote IMPs via leased lines.

ARPANET deployed



29OCT69	2100	LOADED OP. PROGRAM FOR BEN BARKER BBN	CSK
	22:30	Talked to SRI Host to Host	CSK
		Left op. imp program running after sending a host dead message to imp.	CSK

Historical document: First ARPANET IMP log: the first message ever sent via the ARPANET, 10:30 PM, 29 October 1969. This IMP Log excerpt, kept at UCLA, describes setting up a message transmission from the UCLA SDS Sigma 8 Host computer to the SRI SDS 940 Host computer

The initial ARPANET consisted of four IMPs:

- University of California, Los Angeles (UCLA), where Leonard Kleinrock had established a Network Measurement Center, with an SDS Sigma 7 being the first computer attached to it;
- The Stanford Research Institute's Augmentation Research Center, where Douglas Engelbart had created the ground-breaking NLS system, a very important early hypertext system (with the SDS 940 that ran NLS, named "Genie", being the first host attached);
- University of California, Santa Barbara (UCSB), with the Culler-Fried Interactive Mathematics Centre's IBM 360/75, running OS/MVT being the machine attached;
- The University of Utah's Computer Science Department, where Ivan Sutherland had moved, running a DEC PDP-10 running TENEX.

The first message on the ARPANET was sent by UCLA student programmer Charley Kline, at 10:30 p.m, on October 29, 1969. Supervised by Prof. Leonard Kleinrock, Kline transmitted from the university's SDS Sigma 7 Host computer to the Stanford Research Institute's SDS 940 Host computer. The message text was the word "login"; the "l" and the "o" letters were transmitted, but the system then crashed. Hence, the literal first message over the ARPANET was "lo". About an hour later, having recovered from the crash, the SDS Sigma 7 computer effected a full "login". The first permanent ARPANET link was established on November 21, 1969, between the IMP at UCLA and the IMP at the Stanford Research Institute. By December 5, 1969, the entire four-node network was established.

The contents of the first email transmission in 1971 have been forgotten; in the Frequently Asked Questions section of his Web site, the sender, Ray Tomlinson, who sent the message between two computers sitting side-by-side, claims that the contents were "entirely forgettable, and I have, therefore, forgotten them", and speculates that the message likely was "QWERTYUIOP" or some such.

Software and protocols

The starting point for host-to-host communication on the ARPANET was the 1822 protocol BBN Report 1822, which defined the transmission of messages to an IMP. The message format was designed to work unambiguously with a broad range of computer architectures. An 1822 message essentially consisted of a message type, a numeric host address, and a data field. To send a data message to another host, the transmitting host formatted a data message containing the destination host's address and the data message being sent, and then transmitted the message through the 1822 hardware interface. The IMP then delivered the message to its destination address, either by delivering it to a locally connected host, or by delivering it to another IMP. When the message was ultimately delivered to the destination host, the receiving IMP would transmit a *Ready for Next Message* (RFNM) acknowledgement to the sending, host IMP.

Unlike modern Internet datagrams, the ARPANET was designed to reliably transmit 1822 messages, and to inform the host computer when it loses a message; the contemporary IP is unreliable, whereas the TCP is reliable. Nonetheless, the 1822 protocol proved inadequate for handling multiple connections among different applications residing in a host computer. This problem was addressed with the Network Control Program (NCP), which provided a standard method to establish reliable, flow-controlled, bidirectional communications links among different processes in different host computers. The NCP interface allowed application software to connect across the ARPANET by implementing higher-level communication protocols, an early example of the *protocol layering* concept incorporated to the OSI model. In 1983, TCP/IP protocols replaced NCP as the ARPANET's principal protocol, and the ARPANET then became one component of the early Internet.

Network applications

NCP provided a standard set of network services that could be shared by several applications running on a single host computer. This led to the evolution of *application protocols* that operated, more or less, independently of the underlying network service. When the ARPANET migrated to the Internet protocols in 1983, the major application protocols migrated with it.

- E-mail: In 1971, Ray Tomlinson, of the BBN company sent the first network e-mail. By 1973, e-mail constituted 75 percent of ARPANET traffic.
- File transfer: By 1973, the File Transfer Protocol (FTP) specification had been defined and implemented, enabling file transfers over the ARPANET.
- Voice traffic: The Network Voice Protocol (NVP) specifications were defined in (RFC 741), then implemented, but, because of technical shortcomings, conference calls over the ARPANET never worked well; the contemporary Voice over Internet Protocol (packet voice) was decades away.

Growth

In March, 1970, the ARPANET reached the east coast of the United States, when a BBN company IMP was connected to the network. Thereafter, the ARPANET grew: 9 IMPs by June 1970 and 13 IMPs by December 1970, then 18 by September 1971 (when the network included 23 university and government hosts); 29 IMPs by August 1972, and 40 by September, 1973. By June 1974, there were 46 IMPs, and in July 1975, the network numbered 57 IMPs. By 1981, the number was 213 host computers, with another host connecting approximately every twenty days.

In 1968, two satellite links, traversing the Pacific and Atlantic oceans, to Hawaii and Norway, one, the Norwegian Seismic Array (NORSAR), were connected to the ARPANET. Moreover, from Norway, a terrestrial circuit added a London IMP to the network in 1973.

Given that its primary function was funding research and development, the ARPA, in 1975, transferred ARPANET control to the Defense Communications Agency, a component of the U.S. Department of Defense. In 1983, the U.S. military sub-networks of the ARPANET became the discrete Military Network (MILNET) for unclassified defense department communications; separating the civil and military networks reduced the 113-node ARPANET by 68 nodes.

Development: hardware

Support for inter-IMP circuits of up to 230.4 kbit/s was added in 1970, although considerations of cost and IMP processing power meant this capability was not actively used.

1971 saw the start of the use of the non-ruggedized (and therefore significantly lighter) Honeywell 316 as an IMP. It could also be configured as a Terminal IMP (TIP), which added support for up to 63 ASCII serial terminals through a multi-line controller in place of one of the hosts. The 316 featured a greater degree of integration than the 516, which made it less expensive and easier to maintain. The 316 was configured with 40 kB of core memory for a TIP. The size of core memory was later increased, to 32 kB for the IMPs, and 56 kB for TIPs, in 1973.

In 1975, BBN introduced IMP software running on the Pluribus multi-processor. These appeared in a small number of sites. In 1981, BBN introduced IMP software running on its own C/30 processor product.

The original IMPs and TIPs were phased out as the ARPANET was shut down after the introduction of the NSFNet, but some IMPs remained in service as late as 1989.

Senator Albert Gore, Jr. began to craft the High Performance Computing and Communication Act of 1991 (commonly referred to as "The Gore Bill") after hearing the 1988 report toward a National Research Network submitted to Congress by a group chaired by Leonard Kleinrock, professor of computer science at UCLA. The bill was passed on December 9, 1991 and led to the National Information Infrastructure (NII) which Al Gore called the "information superhighway".

Misconceptions of design goals

Common ARPANET lore posits that the computer network was designed to survive a nuclear attack. In *A Brief History of the Internet*, the Internet Society describe the coalescing of the technical ideas that produced the ARPANET:

It was from the RAND study that the false rumor started, claiming that the ARPANET was somehow related to building a network resistant to nuclear war. This was never true of the ARPANET, only the unrelated RAND study on secure voice considered nuclear war. However, the later work on Internetting did emphasize robustness and survivability, including the capability to withstand losses of large portions of the underlying networks.

Although the ARPANET was designed to survive subordinate-network losses, the principal reason was that the switching nodes and network links were unreliable, even without any nuclear attacks. About the resources scarcity that spurred the creation of the ARPANET, Charles Herzfeld, ARPA Director (1965–1967), said:

The ARPANET was not started to create a Command and Control System that would survive a nuclear attack, as many now claim. To build such a system was, clearly, a major military need, but it was not ARPA's mission to do this; in fact, we would have been severely criticized had we tried. Rather, the ARPANET came out of our frustration that there were only a limited number of large, powerful research computers in the country, and that many research investigators, who should have access to them, were geographically separated from them.

Retrospective

The support and management of ARPA contributed to the successful creation of the ARPANET. To wit, the *ARPANET Completion Report*, jointly published by the BBN company and ARPA, concludes that:

... it is somewhat fitting to end on the note that the ARPANET program has had a strong and direct feedback into the support and strength of computer science, from which the network, itself, sprang. [4]

In the wake of ARPANET being formally decommissioned on the 28th of February, 1990, Vinton Cerf wrote the following lamentation, entitled, "Requiem of the ARPANET":

It was the first, and being first, was best,
but now we lay it down to ever rest.
Now pause with me a moment, shed some tears.
For auld lang syne, for love, for years and years
of faithful service, duty done, I weep.
Lay down thy packet, now, O friend, and sleep.

-Vinton Cerf

The ARPANET in film and other media

- A 1969 Walt Disney movie, *The Computer Wore Tennis Shoes*
- A 1983 movie, *WarGames*, is a story of possible nuclear warfare averted by an intelligent but anxious teen who cracks into a government command and control system, the WOPR (a reference to the actual military WMCCS).

- A 1985 episode of the U.S. television sitcom *Benson* includes a scene in which ARPANET is accessed. This is believed to be the first incidence of a popular TV show referencing the Internet or its progenitors.
- In *Let the Great World Spin: A Novel*, published in 2009 but set in 1974 and written by Colum McCann, a character named The Kid and others use ARPANET from a Palo Alto computer to dial phone booths in New York City in order to hear descriptions of Philippe Petit's tight rope walk between the World Trade Center Towers.
- In *Metal Gear Solid 3: Snake Eater*, a character named Sigint takes part in the development of ARPANET after the events depicted in the game.
- The *Doctor Who* Past Doctor Adventures novel *Blue Box*, written in 2003 but set in 1981, includes a character predicting that by the year 2000 there will be four hundred machines connected to ARPANET.
- There is an electronic music artist known as *Arpanet*, Gerald Donald, one of the members of Drexciya. The name is formatted as a word instead of an acronym, but is still a clear nod to ARPANET. The artist's 2002 album *Wireless Internet* features commentary on the expansion of the internet via wireless communication, with songs such as *NTT DoCoMo*, dedicated to the mobile communications giant based in Japan.
- In numerous *The X-Files* episodes ARPANET is referenced and usually hacked into by The Lone Gunmen. This is most noticeable in the episode "Unusual Suspects".
- Thomas Pynchon's 2009 novel Inherent Vice, set in southern California circa 1970, contains a character who accesses the ARPANET throughout the course of the book. ARPANET is spelled therein as 'ARPAnet.'

Chapter 8

Packet Switching

Packet switching is a digital networking communications method that groups all transmitted data – regardless of content, type, or structure – into suitably-sized blocks, called *packets*. Packet switching features delivery of variable-bit-rate data streams (sequences of packets) over a shared network. When traversing network adapters, switches, routers and other network nodes, packets are buffered and queued, resulting in variable delay and throughput depending on the traffic load in the network.

Packet switching contrasts with another principal networking paradigm, circuit switching, a method which sets up a limited number of dedicated connections of constant bit rate and constant delay between nodes for exclusive use during the communication session. In case of traffic fees, for example in cellular communication, circuit switching is characterized by a fee per time unit of connection time, even when no data is transferred, while packet switching is characterized by a fee per unit of information.

Two major packet switching modes exist; connectionless packet switching, also known as datagram switching, and connection-oriented packet switching, also known as virtual circuit switching. In the first case each packet includes complete addressing or routing information. The packets are routed individually, sometimes resulting in different paths and out-of-order delivery. In the second case a connection is defined and preallocated in each involved node before any packet is transferred. The packets include a connection identifier rather than address information, and are delivered in order.

Packet mode communication may be utilized with or without intermediate forwarding nodes (packet switches). In all packet mode communication, network resources are managed by statistical multiplexing or dynamic bandwidth allocation in which a communication channel is effectively divided into an arbitrary number of logical variable-bit-rate channels or data streams. Each logical stream consists of a sequence of packets, which normally are forwarded by the multiplexers and intermediate network nodes asynchronously using first-in, first-out buffering. Alternatively, the packets may be forwarded according to some scheduling discipline for fair queuing or for differentiated or guaranteed quality of service, such as pipeline forwarding or time-driven priority (TDP). Any buffering introduces varying latency and throughput in transmission. In case of a shared physical medium, the packets may be delivered according to some packet-mode multiple access scheme.

History

The concept of switching small blocks of data was first explored by Paul Baran in the early 1960s. Independently, Donald Davies at the National Physical Laboratory in the UK had developed the same ideas (Abbate, 2000).

Leonard Kleinrock conducted early research in queueing theory which would be important in packet switching, and published a book in the related field of digital message switching (without the packets) in 1961; he also later played a leading role in building and management of the world's first packet switched network, the ARPANET.

Baran developed the concept of message block switching during his research at the RAND Corporation for the US Air Force into survivable communications networks, first presented to the Air Force in the summer of 1961 as briefing B-265 then published as RAND Paper P-2626 in 1962 and then including and expanding somewhat within a series of eleven papers titled On Distributed Communications in 1964. Baran's P-2626 paper described a general architecture for a large-scale, distributed, survivable communications network. The paper focuses on three key ideas: first, use of a decentralized network with multiple paths between any two points; and second, dividing complete user messages into what he called *message blocks* (later called packets); then third, delivery of these messages by store and forward switching.

Baran's study made its way to Robert Taylor and J.C.R. Licklider at the Information Processing Technology Office, both wide-area network evangelists, and it helped influence Lawrence Roberts to adopt the technology when Taylor put him in charge of development of the ARPANET.

Baran's work was similar to the research performed independently by Donald Davies at the National Physical Laboratory, UK. In 1965, Davies developed the concept of packet-switched networks and proposed development of a UK wide network. He gave a talk on the proposal in 1966, after which a person from the Ministry of Defense told him about Baran's work. A member of Davies' team met Lawrence Roberts at the 1967 ACM Symposium on Operating System Principles, bringing the two groups together.

Interestingly, Davies had chosen some of the same parameters for his original network design as Baran, such as a packet size of 1024 bits. In 1966 Davies proposed that a network should be built at the laboratory to serve the needs of NPL and prove the feasibility of packet switching. The NPL Data Communications Network entered service in 1970. Roberts and the ARPANET team took the name "packet switching" itself from Davies's work.

The first computer network and packet switching network deployed for computer resource sharing was the Octopus Network at the Lawrence Livermore National Laboratory that began connecting four Control Data 6600 computers to several shared storage devices (including an IBM 2321 Data Cell in 1968 and an IBM Photostore in

1970) and to several hundred ASR-33 Teletype terminals for time sharing use starting in 1968.

Connectionless and connection-oriented packet switching

The service actually provided to the user by networks using packet switching nodes can be either connectionless (based on datagram messages), or virtual circuit switching (also known as connection oriented). Some connectionless protocols are Ethernet, IP, and UDP; connection oriented packet-switching protocols include X.25, Frame relay, Asynchronous Transfer Mode (ATM), Multiprotocol Label Switching (MPLS), and TCP.

In connection oriented networks, each packet is labeled with a connection ID rather than an address. Address information is only transferred to each node during a connection set-up phase, when the route to the destination is discovered and an entry is added to the switching table in each network node through which the connection passes. The signalling protocols used allow the application to specify its requirements and the network to specify what capacity etc. is available, and acceptable values for service parameters to be negotiated. Routing a packet is very simple, as it just requires the node to look up the ID in the table. The packet header can be small, as it only needs to contain the ID and any information (such as length, timestamp, or sequence number) which is different for different packets.

In connectionless networks, each packet is labeled with a destination address, source address, and port numbers; it may also be labeled with the sequence number of the packet. This precludes the need for a dedicated path to help the packet find its way to its destination, but means that much more information is needed in the packet header, which is therefore larger, and this information needs to be looked up in power-hungry content-addressable memory. Each packet is dispatched and may go via different routes; potentially, the system has to do as much work for every packet as the connection-oriented system has to do in connection set-up, but with less information as to the application's requirements. At the destination, the original message/data is reassembled in the correct order, based on the packet sequence number. Thus a virtual connection, also known as a virtual circuit or byte stream is provided to the end-user by a transport layer protocol, although intermediate network nodes only provides a connectionless network layer service.

Packet-switched network

A **packet-switched network** is a digital communications network that groups all transmitted data, irrespective of content, type, or structure into suitably-sized blocks, called *packets*. The network over which packets are transmitted is a shared network which routes each packet independently from all others and allocates transmission resources as needed.

The principal goals of packet switching are to optimize utilization of available link capacity, minimize response times and increase the robustness of communication. When traversing network adapters, switches and other network nodes, packets are buffered and queued, resulting in variable delay and throughput, depending on the traffic load in the network.

The history of such networks can be divided into three eras: early networks before the introduction of X.25 and the OSI model, the X.25 era when many postal, telephone and telegraph (PTT) companies introduced networks with X.25 interfaces, and the Internet era when restrictions on connection to the Internet were removed.

Early networks

ARPANET and SITA HLN became operational in 1969. Before the introduction of X.25 in 1973, about twenty different network technologies were developed. There was a debate about the merits of two drastically different views as to proper division of labor between the hosts and the network. In the datagram system the host must detect loss or duplication of packets. Transmission Control Protocol /Internet Protocol (TCP/IP) is the best known example of a host to datagram protocol. In the virtual call system, the network guarantees sequenced delivery of data to the host. This results in a simpler host interface with less functionality than in the datagram model. X.25 is the best known virtual call protocol.

Inexpensive minicomputers were an important component in the early networks. In some cases custom I/O devices were added to allow inexpensive or exotic attachments to communication lines.

ARPANET

This is the principal survivor from the early era. TCP/IP, which was an important component of ARPANET2, was chosen for use in NSFNET which eventually became the Internet.

BNRNET

BNRNET was a network which Bell Northern Research developed for internal use. It initially had only one host but was designed to support many hosts. BNR later made major contributions to the CCITT X.25 project.

CTNE

Compañía Telefónica Nacional de España was the national telephone company in Spain. They developed a network called RETD.

CYCLADES

CYCLADES was an experimental French network. Louis Pouzin was the principal designer. Some ideas from this network were later incorporated into ARPANET.

DDX-1

This was an experimental network from Nippon PTT. It mixed circuit switching and packet switching. It was succeeded by DDX-2.

EIN nee COST II

European Informatics Network was a project to link several national networks. It became operational in 1976.

EPSS

EPSS (Experimental Packet Switching System) was an experiment of the UK Post Office. Ferranti supplied the hardware and software. The handling of link control messages (acknowledgements and flow control) was differed from that of most another networks and is not fully explained in the published literature.

GEIS

As General Electric Information Services (GEIS), General Electric was a major international provider of information services. The company originally designed a telephone network to serve as its internal (albeit continent-wide) voice telephone network.

In 1965, at the instigation of Warner Sinback, a data network based on this voice-phone network was designed to connect GE's four computer sales and service centers (Schenectady, Phoenix, Chicago, and Phoenix) to facilitate a computer time-sharing service, apparently the world's first commercial online service. (In addition to selling GE computers, the centers were computer service bureaus, offering batch processing services. They lost money from the beginning, and Sinback, a high-level marketing manager, was given the job of turning the business around. He decided that a time-sharing system, based on Kemney's work at Dartmouth—which used a computer on loan from GE—could be profitable. Warner was right.)

After going international some years later, GEIS created a network data center near Cleveland, Ohio. Very little has been published about the internal details of their network. (Though it has been stated by some that Tymshare copied the GEIS system to create their network, Tymnet.) The design was hierarchal with redundant communication links.

IPSANET

IPSANET was a semi-private network constructed by I. P. Sharp Associates to serve their time-sharing customers. It became operational in May 1976.

NPL

Donald Davies of the National Physical Laboratory, UK made many important contributions to the theory of packet switching. NPL built a single node network to connect sundry hosts at NPL.

OCTOPUS

Octopus was a local network at Lawrence Livermore National Laboratory. It connected sundry hosts at the lab to interactive terminals and various computer peripherals including a bulk storage system.

Philips Research

Philips Research Laboratories in Redhill, Surrey developed a packet switching network for internal use. It was a datagram network with a single switching node.

PUP

The **PARC Universal Packet** (**PUP** or **Pup**) was one of the two earliest internetwork protocol suites; it was created by researchers at Xerox PARC in the mid-1970s. The entire suite provided routing and packet delivery, as well as higher level functions such as a reliable byte stream, along with numerous applications. Further developments led to Xerox Network Systems (XNS).

RCP

RCP was an experimental network created by the French PTT. It was used to gain experience with packet switching technology before the specification of TRANSPAC was frozen. RCP was a virtual-call network in contrast to CYCLADES which was based on datagrams. RCP emphasised terminal to host and terminal to terminal connection; CYCLADES was concerned with host-to-host communication. TRANSPAC was introduced as an X.25 network. RCP may have influenced the specification of X.25.

RETD

Red Especial de Transmisión de Datos was a network developed in Spain. It became operational in 1972 and thus was the first public network.

SCANNET

"The experimental packet-switched Nordic telecommunication network SCANNET was implemented in Nordic technical libraries in 70's, and it included first Nordic electronic journal Extemplo. Libraries were also among first ones in universities to accommodate microcomputers for public use in early 80's."

SITA HLN

SITA is a consortium of airlines. Their High Level Network became operational in 1969 at about the same time as ARPANET. It carried interactive traffic and message-switching traffic. As with many non-academic networks very little has been published about it.

SNA

Systems Network Architecture (SNA) is IBM's proprietary networking architecture created in 1974. An IBM customer could acquire hardware and software from IBM and lease private lines from a common carrier. This allowed construction of a private network.

Telenet

Telenet was the first FCC-licensed public data network in the United States. It was founded by former ARPA IPTO director Larry Roberts as a means of making ARPANET technology public. He had tried to interest AT&T in buying the technology, but the monopoly's reaction was that this was incompatible with their future. Bolt, Beranack and Newman (BBN) provided the financing.

It initially used ARPANET technology but changed the host interface to X.25 and the terminal interface to X.29. Telenet designed these protocols and helped standardize them in the CCITT. Telenet was incorporated in 1973 and started operations in 1975. It went public in 1979 and was then sold to GTE.

Tymnet

Tymnet was an international data communications network headquartered in San Jose, CA that utilized virtual call packet switched technology and used X.25, SNA/SDLC, BSC and ASCII interfaces to connect host computers (servers)at thousands of large companies, educational institutions, and government agencies. Users typically connected via dial-up connections or dedicated async connections. The business consisted of a large public network that supported dial-up users and a private network business that allowed government agencies and large companies (mostly banks and airlines) to build their own dedicated networks. The private networks were often connected via gateways to the public network to reach locations not on the private network. Tymnet was also connected to dozens of other public networks in the U.S. and internationally via X.25/X.75

gateways. (Interesting note: Tymnet was not named after Mr. Tyme. Another employee suggested the name.)

XNS

Xerox Network Systems (**XNS**) was a protocol suite promulgated by Xerox, which provided routing and packet delivery, as well as higher level functions such as a reliable stream, and remote procedure calls. It was developed from PARC Universal Packet (PUP).

X.25 era

There were two kinds of X.25 networks. Some such as DATAPAC and TRANSPAC were initially implemented with an X.25 external interface. Some older networks such as TELENET and TYMNET were modified to provide a X.25 host interface in addition to older host connection schemes. DATAPAC was developed by Bell Northern Research which was a joint venture of Bell Canada (a common carrier) and Northern Telecom (a telecommunications equipment supplier). Northern Telecom sold several DATAPAC clones to foreign PTTs including the Deutsche Bundespost. X.75 and X.121 allowed the interconnection of national X.25 networks. A user or host could call a host on a foreign network by including the DNIC of the remote network as part of the destination address.

Austpac

AUSTPAC is an Australian public X.25 network operated by Telstra. Started by the then-Telecom in the early 1980s, AUSTPAC was Australia's first public packet-switched data network, supporting applications such as on-line betting, financial applications — the Australian Tax Office has made use of AUSTPAC — and remote terminal access to academic institutions, who maintained their connections to AUSTPAC up until the mid-late 1990s in some cases. Access can be via a dial-up terminal to a PAD, or, by linking a permanent X.25 node to the network.

ConnNet

ConnNet was a packet switched data network operated by the Southern New England Telephone Company serving the state of Connecticut.

Datanet 1

Datanet 1 was the public switched data network operated by the Dutch **PTT Telecom** (now known as KPN). Strictly speaking Datanet 1 only referred to the network and the connected users via leased lines (using the X.121 DNIC 2041), the name also referred to the public PAD service *Telepad* (using the DNIC 2049). And because the main Videotex service used the network and modified PAD devices as infrastructure the name Datanet 1 was used for these services as well. Although this use of the name was incorrect all these services were managed by the same people within one department of KPN contributed to the confusion.

Datapac

DATAPAC was the first operational X.25 network (1976). It covered major Canadian cities and was eventually extended to smaller centres.

Datex-P

Deutsche Bundespost operated this national network in Germany. The technology was acquired from Northern Telecom.

Eirpac

Eirpac is the Irish public switched data network supporting X.25 and X.28. It was launched in 1984, replacing Euronet. Eirpac is run by Eircom.

HIPA-NET

Hitachi designed a private network system for sale as a turnkey package to multi-national organizations. In addition to providing X.25 packet switching, message switching software was also included. Messages were buffered at the nodes adjacent to the sending and receiving terminals. Switched virtual calls were not supported, but through the use of "logical ports" an originating terminal could have a menu of pre-defined destination terminals.

Iberpac

Iberpac is the Spanish public packet switched network, providing X.25 services. Iberpac is run by Telefonica.

JANET

JANET was the UK academic and research network, linking all universities, higher education establishments, publicly funded research laboratories. The X.25 network was based mainly on GEC 4000 series switches, and run X.25 links at up to 8 Mbit/s in its final phase before being converted to an IP based network. The JANET network grew out of the 1970s SRCnet (later called SERCnet) network.

PSS

PSS was the UK Post Office (later to become British Telecom) national X.25 network with a DNIC of 2342. British Telecom renamed PSS under its GNS (Global Network

Service) name, but the PSS name has remained better known. PSS also included public dial-up PAD access, and various InterStream gateways to other services such as Telex.

Transpac

Transpac was the national X.25 network in France. It was developed locally at about the same time as DataPac in Canada. The development was done by the French PTT and influenced by the experimental RCP network. It began operation in 1978.

Internet era

When Internet connectivity was made available to anyone who could pay for an ISP subscription, the distinctions between national networks blurred. The user no longer saw network identifiers such as the DNIC. Some older technologies such as circuit switching have resurfaced with new names such as fast packet switching. Researchers have created some experimental networks to complement the existing Internet.

Internet2

Internet2 is not an actual network. It is a research consortium which has created the Abilene Network.

National LambdaRail

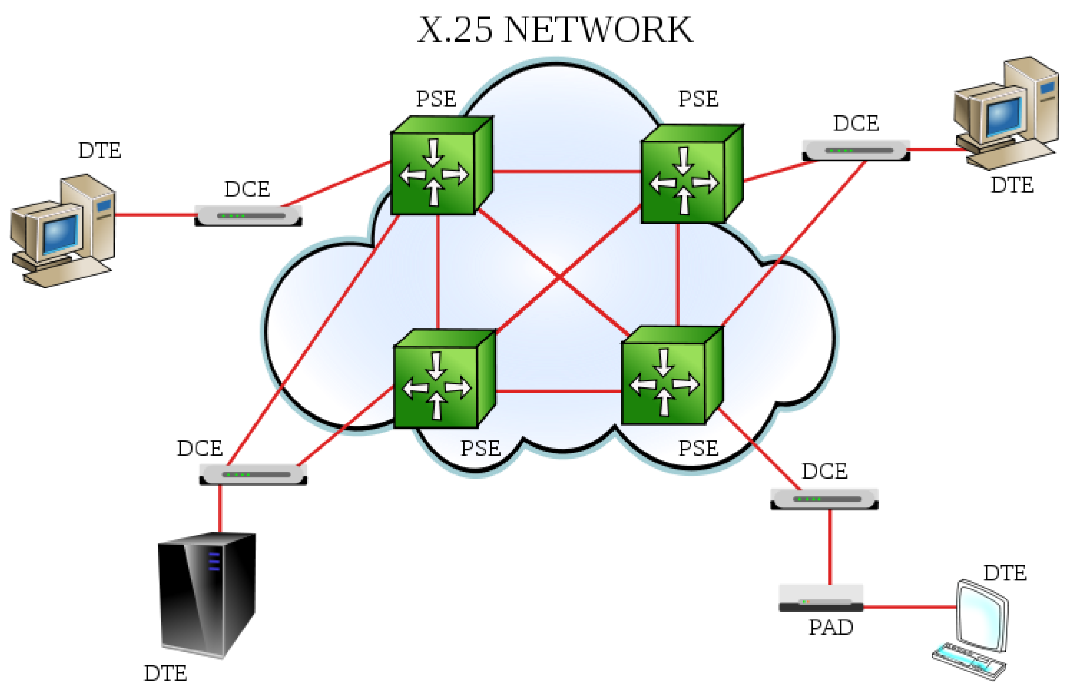
National LambdaRail is a high-speed national computer network in the United States that runs over fiber-optic lines, and is the first transcontinental Ethernet network working to establish a direct line of communications between international parties.

X.25 vs. Frame Relay packet switching

Both X.25 and Frame Relay provide connection-oriented packet switching, also known as virtual circuit switching. A major difference between X.25 and frame relay packet switching are that X.25 is a reliable protocol, based on node-to-node automatic repeat request, while Frame Relay is a non-reliable protocol, maximum packet length is 1000 bytes. Any retransmissions must be carried out by higher layer protocols. The X.25 protocol is a network layer protocol, and is part of the X.25 protocol suite, also known as the OSI protocol suite. It was widely used in relatively slow switching networks during the 1980s, for example as an alternative to circuit mode terminal switching, and for automated teller machines. Frame relay is a further development of X.25. The simplicity of Frame relay made it considerably faster and more cost effective than X.25 packet switching. Frame relay is a data link layer protocol, and does not provide logical addresses and routing. It is only used for semi-permanent connections, while X.25 connections also can be established for each communication session. Frame relay was used to interconnect LANs or LAN segments, mainly in the 1990s by large companies that had a requirement to handle heavy telecommunications traffic across wide area networks. (O'Brien & Marakas, 2009, p. 250) Despite the benefits of frame relay packet switching, many international companies are staying with the X.25 standard. In the United States, X.25 packet switching was used heavily in government and financial networks that use mainframe applications. Many companies did not intend to cross over to frame relay packet switching because it is more cost effective to use X.25 on slower networks. In certain parts of the world, particularly in Asia-Pacific and South America regions, X.25 was the only technology available. (Girard, 1997)

Chapter 9

X.25



X.25 network diagram

X.25 is an ITU-T standard protocol suite for packet switched wide area network (WAN) communication. An X.25 WAN consists of packet-switching exchange (PSE) nodes as the networking hardware, and leased lines, Plain old telephone service connections or ISDN connections as physical links. X.25 is a family of protocols that was used especially during the 1980s by telecommunications companies and in financial transaction systems such as automated teller machines. X.25 was originally defined by the International Telegraph and Telephone Consultative Committee (CCITT, now ITU-T) in a series of drafts and finalized in a publication known as *The Orange Book* in 1976.

X.25 is today to a large extent replaced by less complex protocols, especially the Internet protocol (IP) although some telephone operators offer X.25-based communication via the signaling (*D*) channel of ISDN lines.

History

X.25 is one of the oldest packet-switched services available. It was developed before the OSI Reference Model. The protocol suite is designed as three conceptual layers, which correspond closely to the lower three layers of the seven-layer OSI model. It also supports functionality not found in the OSI Network Layer.

X.25 was developed in the ITU-T (formerly CCITT) Study Group VII based upon a number of emerging data network projects. Various updates and additions were worked into the standard, eventually recorded in the ITU series of technical books describing the telecommunication systems. These books were published every fourth year with different-colored covers. The X.25 specification is only part of the larger set of X-Series specifications on public data networks.

The *Public data network* was the common name given to the international collection of X.25 providers. Their combined network had large global coverage during the 1980s and into the 1990s.

Publicly-accessible X.25 networks (Compuserve, Tymnet, Euronet, PSS, Datapac, and Telenet) were set up in most countries during the 1970s and 80s, to lower the cost of accessing various online services.

Beginning in the early 1990s in North America, use of X.25 networks (predominated by Telenet and Tymnet) began being replaced with Frame Relay service offered by national telephone companies.

X.25 networks are in use throughout the world. A variant called AX.25 is also used widely by amateur packet radio. Racal Paknet, now known as Widanet, is still in operation in many regions of the world, running on an X.25 protocol base. In some countries, like the Netherlands or Germany, it is possible to use a stripped version of X.25 via the D-channel of an ISDN-2 (or ISDN BRI) connection for low volume applications such as point-of-sale terminals; but, the future of this service in the Netherlands is uncertain. Additionally X.25 is still under heavy use in the aeronautical business (especially in the Asian region) even though a transition to modern protocols like X.400 is without option as X.25 hardware becomes increasingly rare and costly. As recently as March 2006, the National Airspace Data Interchange Network has used X.25 to interconnect remote airfields with Air Route Traffic Control Centers.

Architecture

The general concept of X.25 was to create a universal and global packet-switched network. Much of the X.25 system is a description of the rigorous error correction needed to achieve this, as well as more efficient sharing of capital-intensive physical resources.

The X.25 specification defines only the interface between a subscriber (DTE) and an X.25 network (DCE). X.75, a very similar protocol to X.25, defines the interface between two X.25 networks to allow connections to traverse two or more networks. X.25 does not specify how the network operates internally—many X.25 network implementations used something very similar to X.25 or X.75 internally, but others used quite different protocols internally. The ISO equivalent protocol to X.25, ISO 8208, is compatible with X.25, but additionally includes provision for two X.25 DTEs to be directly connected to each other with no network in between. By separating the Packet-Layer Protocol, ISO 8208 permits operation over additional networks such as ISO 8802 LLC2 (ISO LAN) and the OSI data link layer.

X.25 originally defined three basic protocol levels or architectural layers. In the original specifications these were referred to as *levels* and also had a level number, whereas all ITU-T X.25 recommendations and ISO 8208 standards released after 1984 refer to them as *layers*. The layer numbers were dropped to avoid confusion with the OSI Model layers.

- Physical layer: This layer specifies the physical, electrical, functional and procedural characteristics to control the physical link between a DTE and a DCE. Common implementations use X.21, EIA-232, EIA-449 or other serial protocols.
- Data link layer: The data link layer consists of the link access procedure for data interchange on the link between a DTE and a DCE. In its implementation, the Link Access Procedure, Balanced (LAPB) is a data link protocol that manages a communication session and controls the packet framing. It is a bit-oriented protocol that provides error correction and orderly delivery.
- Packet layer: This layer defined a packet-layer protocol for exchanging control and user data packets to form a packet-switching network based on virtual calls, according to the Packet Layer Protocol.

The X.25 model was based on the traditional telephony concept of establishing reliable circuits through a shared network, but using software to create "virtual calls" through the network. These calls interconnect "data terminal equipment" (DTE) providing endpoints to users, which looked like point-to-point connections. Each endpoint can establish many separate virtual calls to different endpoints.

For a brief period, the specification also included a connectionless datagram service, but this was dropped in the next revision. The "fast select with restricted response facility" is intermediate between full call establishment and connectionless communication. It is widely used in query-response transaction applications involving a single request and response limited to 128 bytes of data carried each way. The data is carried in an extended call request packet and the response is carried in an extended field of the call reject packet, with a connection never being fully established.

Closely related to the X.25 protocol are the protocols to connect asynchronous devices (such as dumb terminals and printers) to an X.25 network: X.3, X.28 and X.29. This functionality was performed using a Packet Assembler/Disassembler or PAD (also known as a *Triple-X device*, referring to the three protocols used).

Relation to the OSI Reference Model

Although X.25 predates the OSI Reference Model (OSIRM), the Physical Layer of the OSI model corresponds to the X.25 *physical layer*, the Data Link Layer to the X.25 *data link layer*, and the Network Layer to the X.25 *packet layer*. The X.25 *data link layer*, LAPB, provides a reliable data path across a data link (or multiple parallel data links, multilink) which may not be reliable itself. The X.25 *packet layer*, provides the virtual call mechanisms, running over X.25 LAPB. The *packet layer* includes mechanisms to maintain virtual calls and to signal data errors in the event that the *data link layer* cannot recover from data transmission errors. All but the earliest versions of X.25 include facilities which provide for OSI network layer Addressing.

User device support

A Televideo terminal model 925 made around 1982

X.25 was developed in the era of dumb terminals connecting to host computers, although it also can be used for communications between computers. Instead of dialing directly “into” the host computer — which would require the host to have its own pool of modems and phone lines, and require non-local callers to make long-distance calls — the host could have an X.25 connection to a network service provider. Now dumb-terminal users could dial into the network's local “PAD” (Packet Assembly/Disassembly facility), a gateway device connecting modems and serial lines to the X.25 link as defined by the X.29 and X.3 standards.

Having connected to the PAD, the dumb-terminal user tells the PAD which host to connect to, by giving a phone-number-like address in the X.121 address format (or by giving a host name, if the service provider allows for names that map to X.121 addresses). The PAD then places an X.25 call to the host, establishing a virtual call. Note that X.25 provides for virtual calls, so *appears* to be a circuit switched network, even though in fact the data itself is packet switched internally, similar to the way TCP provides connections even though the underlying data is packet switched. Two X.25 hosts could, of course, call one another directly; no PAD is involved in this case. In theory, it doesn't matter whether the X.25 caller and X.25 destination are both connected to the same carrier, but in practice it was not always possible to make calls from one carrier to another.

For the purpose of flow-control, a sliding window protocol is used with the default window size of 2. The acknowledgements may have either local or end to end significance. A D bit (Data Delivery bit) in each data packet indicates if the sender requires end to end acknowledgement. When D=1, it means that the acknowledgement has end to end significance and must take place only after the remote DTE has acknowledged receipt of the data. When D=0, the network is permitted (but not required) to acknowledge before the remote DTE has acknowledged or even received the data.

While the PAD function defined by X.28 and X.29 specifically supported asynchronous character terminals, PAD equivalents were developed to support a wide range of proprietary intelligent communications devices, such as those for IBM System Network Architecture (SNA).

Error control

Error recovery procedures at the packet layer assume that the data link layer is responsible for retransmitting data received in error. Packet layer error handling focuses on resynchronizing the information flow in calls, as well as clearing calls that have gone into unrecoverable states:

- Level 3 Reset packets, which re-initializes the flow on a virtual call (but does not break the virtual call)
- Restart packet, which clears down all virtual calls on the data link and resets all permanent virtual circuits on the data link

Addressing and virtual circuits

An X.25 Modem once used to connect to the German Datex-P network

X.25 supports two types of virtual circuits, Virtual Calls (VC) and Permanent Virtual Circuits (PVC). Virtual Calls are established on an as-needed basis. For example, a VC is established when a call is placed and torn down after the call is complete. VCs are established through a call establishment and clearing procedure. On the other hand, Permanent Virtual Circuits are preconfigured into the network. PVCs are seldom torn down and thus provide a dedicated connection between end points. Virtual Calls were also commonly referred to as Switched Virtual Circuits (SVC).

VC may be established using X.121 addresses. The X.121 address consists of a three-digit *Data Country Code* (DCC) plus a network digit, together forming the four-digit *Data Network Identification Code* (DNIC), followed by the *National Terminal Number* (NTN) of at most ten digits. Note the use of a single network digit, seemingly allowing for only 10 network carriers per country, but some countries are assigned more than one DCC to avoid this limitation. Networks often used fewer than the full NTN digits for routing, and made the spare digits available to the subscriber (sometimes called the sub-address) where they could be used to identify applications or for further routing on the subscribers networks.

NSAP addressing facility was added in the X.25(1984) revision of the specification, and this enabled X.25 to better meet the requirements of OSI Connection Oriented Network Service (CONS). Public X.25 networks were not required to make use of NSAP addressing, but, to support OSI CONS, were required to carry the NSAP addresses and other ITU-T specified DTE facilities transparently from DTE to DTE. Later revisions allowed multiple addresses in addition to X.121 addresses to be carried on the same DTE-DCE interface: Telex addressing (F.69), PSTN addressing (E.163), ISDN addressing (E.164), Internet Protocol addresses (IANA ICP), and local IEEE 802.2 MAC addresses.

PVCs are permanently established in the network and therefore do not require the use of addresses for call setup. PVCs are identified at the subscriber interface by their logical channel identifier (see below). However, in practice not many of the national X.25 networks supported PVCs.

One DTE-DCE interface to an X.25 network has a maximum of 4095 logical channels on which it is allowed to establish virtual calls and permanent virtual circuits, although networks are not expected to support a full 4095 virtual circuits. For identifying the channel to which a packet is associated, each packet contains a 12 bit logical channel identifier made up of an 8-bit *Logical Channel Number* and a 4-bit *Logical Channel Group Number.* Logical channel identifiers remain assigned to a virtual circuit for the duration of the connection. Logical channel identifiers identify a specific logical channel between the DTE (subscriber appliance) and the DCE (network), and only has local significance on the link between the subscriber and the network. The other end of the connection at the remote DTE is likely to have assigned a different logical channel identifier. The range of possible logical channels is split into 4 groups: channels assigned to permanent virtual circuits, assigned to incoming virtual calls, two-way (incoming or outgoing) virtual calls, and outgoing virtual calls. (Directions refer to the direction of virtual call initiation as viewed by the DTE—they all carry data in both directions.) The ranges allowed a subscriber to be configured to handle significantly differing numbers of calls in each direction while reserving some channels for calls in one direction. All International networks are required to implement support for permanent virtual circuits, two-way logical channels and one-way logical channels outgoing; one-way logical channels incoming is an additional optional facility. DTE-DCE interfaces are not required to support more than one logical channel. Logical channel identifier zero will not be assigned to a permanent virtual circuit or virtual call. The logical channel identifier of zero is used for packets which don't relate to a specific virtual circuit (e.g. packet layer restart, registration, and diagnostic packets).

Billing

In public networks, X.25 was typically billed as a flat monthly service fee depending on link speed, and then a price-per-segment on top of this. Link speeds varied, typically from 2400bit/s up to 2 Mbit/s, although speeds above 64 kbit/s were uncommon in the public networks. A segment was 64 bytes of data (rounded up, with no carry-over between packets), charged to the caller (or callee in the case of reverse charged calls, where supported). Calls invoking the *Fast Select* facility (allowing 128 bytes of data in call request, call confirmation and call clearing phases) would generally attract an extra charge, as might use of some of the other X.25 facilities. PVCs would have a monthly rental charge and a lower price-per-segment than VCs, making them cheaper only where large volumes of data are passed.

X.25 packet types

Packet Type	DCE -> DTE	DTE -> DCE	Service	VC	PVC
Call setup and Clearing	Incoming Call	Call Request		X	
	Call Connected	Call Accepted		X	
	Clear Indication	Clear Request		X	
	Clear Confirmation	Clear Confirmation		X	
Data and Interrupt	Data	Data		X	X
	Interrupt	Interrupt		X	X
	Interrupt Confirmation	Interrupt Confirmation		X	X
Flow Control and Reset	RR	RR		X	X
	RNR	RNR		X	X
	REJ	REJ		X	X
	Reset Indication	Reset Request		X	X
	Reset Confirmation	Reset Confirmation		X	X
Restart	Restart Indication	Restart Request	X		
	Restart Confirmation	Restart Confirmation	X		
Diagnostic	Diagnostic		X		
Registration	Registration Confirmation	Registration Request	X		

X.25 details

The network may allow the selection of the maximal length in range 16 to 4096 octets (2^n values only) per virtual circuit by negotiation as part of the call setup procedure. The maximal length may be different at the two ends of the virtual circuit.

- Data terminal equipment constructs control packets which are encapsulated into data packets. The packets are sent to the data circuit-terminating equipment, using LAPB Protocol.
- Data circuit-terminating equipment strips the layer-2 headers in order to encapsulate packets to the internal network protocol.

X.25 facilities

X.25 provides a set of user facilities defined and described in ITU-T Recommendation X.2. The X.2 user facilities fall into five categories:

- essential facilities;
- additional facilities;
- conditional facilities;
- mandatory facilities; and,
- optional facilities.

X.25 also provides X.25 and ITU-T specified DTE optional user facilities defined and described in ITU-T Recommendation X.7. The X.7 optional user facilities fall into four categories of user facilities that require:

- subscription only;
- subscription followed by dynamic invocation;
- subscription or dynamic invocation; and,
- dynamic invocation only.

X.25 protocol versions

The CCITT/ITU-T versions of the protocol specifications are for Public Data Networks (PDN). The ISO/IEC versions address additional features for private networks (e.g. Local Area Networks (LAN) use) while maintaining compatibility with the CCITT/ITU-T specifications.

The user facilities and other features supported by each version of X.25 and ISO/IEC 8208 have varied from edition to edition. Several major protocol versions of X.25 exist:

- CCITT Recommendation X.25 (1976) Orange Book
- CCITT Recommendation X.25 (1980) Yellow Book
- CCITT Recommendation X.25 (1984) Red Book
- CCITT Recommendation X.25 (1988) Blue Book
- ITU-T Recommendation X.25 (1993) White Book
- ITU-T Recommendation X.25 (1996) Grey Book

The X.25 Recommendation allows many options for each network to choose when deciding which features to support and how certain operations are performed. This means each network needs to publish its own document giving the specification of its X.25 implementation, and most networks required DTE appliance manufacturers to undertake protocol conformance testing, which included testing for strict adherence and enforcement of their network specific options. (Network operators were particularly concerned about the possibility of a badly behaving or misconfigured DTE appliance taking out parts of the network and affecting other subscribers.) Therefore, subscriber's DTE appliances have to be configured to match the specification of the particular network to which they are connecting. Most of these were sufficiently different to prevent interworking if the subscriber didn't configure their appliance correctly or the appliance manufacturer didn't include specific support for that network. In spite of protocol conformance testing, this often lead to interworking problems when initially attaching an appliance to a network. This is in stark contrast to the Robustness Principle employed in the Internet Protocol family.

Public networks were adopters of the earlier protocol versions, but reluctant to upgrade fearing subscriber compatibility issues and struggling to justify the expense. Most public networks ended up running something roughly on a parity with X.25 (1980) with some parts of X.25 (1984). Private networks started using X.25 later and were more likely to take upgrades, and many of those operated something nearer to X.25 (1984) with a few X.25 (1988) features. By about 1990, X.25 development by all the major network switch vendors had ceased, and there were no significant implementations of the 1993 and 1996 protocol versions.

In addition to the CCITT/ITU-T versions of the protocol, four editions of ISO/IEC 8208 exist:

- ISO/IEC 8208: 1987, First Edition, compatible with X.25 (1980) and (1984)
- ISO/IEC 8208: 1990, Second Edition, compatible with 1st Ed. and X.25 (1988).
- ISO/IEC 8208: 1995, Third Edition, compatible with 2nd Ed. and X.25 (1993).
- ISO/IEC 8208: 2000, Fourth Edition, compatible with 3rd Ed. and X.25 (1996).

Chapter 10

FidoNet

```
                 __
                /  \
               /|oo \
              (_|  /_)
               _`@/_ \    _
              |     | \   \\
              | (*) |  \   ))
   ______     |__U__| /  \//
  / FIDO \     _//|| _\   /
 (________)   (_/(_|(____/
```

Old FidoNet logo

"New" FidoNet logo

FidoNet is a worldwide computer network that is used for communication between bulletin board systems. It was most popular in the early to mid 1990s, prior to the introduction of easy and affordable access to the Internet. The network continues to operate but has shrunk considerably, primarily due to the closing of many BBSes during the increase of popularity of the Internet.

Origin

FidoNet was originally founded as a non-commercial network in 1984 by Tom Jennings of San Francisco, California as a means to network BBSes that used his own "Fido" BBS software. Over time, other BBS software was independently adapted to support the relevant FidoNet protocols and the network became a popular means for computer users to communicate. Until 1994 with the arrival of commercial Internet, Fidonet was the only non Compuserve, Minerva or Prodigy methodology for most of the world's population to send email to and from the Milnet, Arpanet and Minitel networks.

FidoNet organizational structure

FidoNet is governed in a hierarchical structure according to FidoNet policy, with designated coordinators at each level to manage the administration of FidoNet nodes and resolve disputes between members. Network coordinators are responsible for managing the individual nodes within their area, usually a city or similar sized area. Regional coordinators are responsible for managing the administration of the network coordinators within their region, typically the size of a state, or small country. Zone coordinators are responsible for managing the administration of all of the regions within their zone. The world is divided into six zones, the coordinators of which elect one of themselves to be the "International Coordinator" of FidoNet.

Technical structure

FidoNet was historically designed to use modem-based dial-up (POTS) access between bulletin board systems, and much of its policy and structure reflected this.

The FidoNet system officially referred only to transfer of **Netmail**—the individual private messages between people using bulletin boards—including the protocols and standards with which to support it. A netmail message would contain the name of the person sending, the name of the intended recipient, and the respective FidoNet addresses of each. The FidoNet system was responsible for routing the message from one system to the other (details below), with the bulletin board software on each end being responsible for ensuring that only the intended recipient could read it. Due to the hobbyist nature of the network, any privacy between sender and recipient was only the result of politeness from the owners of the FidoNet systems involved in the mail's transfer. It was common, however, for system operators to reserve the right to review the content of mail that passed through their system.

Netmail allowed for the "attachment" of a single file to every message. This led to a series of "piggyback" protocols that built additional features onto FidoNet by passing information back and forth as file attachments. These included the automated distribution of files, and transmission of data for inter-BBS games.

By far the most commonly-used of these piggyback protocols was **Echomail**, public discussions similar to Usenet newsgroups in nature. Echomail was supported by a variety of software that collected up new messages from the local BBSes' public forums (the *scanner*), compressed it using ARC or ZIP, attached the resulting archive to a Netmail message, and sent that message to a selected system. On receiving such a message, identified because it was addressed to a particular "user", the reverse process was used to extract the messages, and a *tosser* put them back into the new system's forums.

Echomail was so popular that for many users, Echomail *was* the FidoNet. Private person-to-person Netmail was relatively rare.

Geographic structure

FidoNet is politically organized into a tree structure, with different parts of the tree electing their respective coordinators. The FidoNet hierarchy consists of Zones, Regions, Networks, Nodes and Points broken down more-or-less geographically.

The highest level is the Zone, which is largely continent-based:

- Zone 1 is North America
- Zone 2 is Europe, Former Soviet Union Countries, and Israel
- Zone 3 is Australasia
- Zone 4 is Latin America (except Puerto Rico)
- Zone 5 is Africa
- Zone 6 is Asia (excluding Israel and the Asian parts of Russia, which are listed in Zone 2) -- On 26 July 2007 Zone 6 was removed, and all remaining nodes were moved to Zone 3.

Each zone is broken down into regions, which are broken down into nets, which consist of individual nodes. Zones 7-4095 are used for "othernets"; groupings of nodes which use Fido-compatible software to carry their own independent message areas without being in any way controlled by FidoNet's political structure. Using un-used zone numbers would ensure that each network would have a unique set of addresses, avoiding potential routing conflicts and ambiguities for systems that belonged to more than one network.

FidoNet addresses

FidoNet addresses explicitly consist of a **Zone** number, a **Network** number (or region number), and a **Node** number. They are written in the form `Zone:Network/Node`. The FidoNet structure also allows for semantic designation of region, host, and hub status for particular nodes, but this status is not directly indicated by the main address.

For example, consider a node located in Tulsa, Oklahoma, USA with an assigned node number is 918, located in Zone 1 (North America), Region 19, and Network 170. The full FidoNet address for this system would be `1:170/918`. The *region* was used for administrative purposes, and was only part of the address if the node was listed directly underneath the Regional Coordinator, rather than one of the networks that were used to divide the region further.

FidoNet policy requires that each FidoNet system maintain a *nodelist* of every other member system. Information on each node includes the name of the system or BBS, the name of the node operator, the geographic location, the telephone number, and software capabilities. The nodelist is updated weekly, to avoid unwanted calls to nodes that had shut down, with their phone numbers possibly having been reassigned for voice use by the respective telephone company.

To accomplish regular updates, coordinators of each network maintain the list of systems in their local areas. The lists are forwarded back to the International Coordinator via automated systems on a regular basis. The International Coordinator would then compile a new nodelist, and generate the list of changes (nodediff) to be distributed for node operators to apply to their existing nodelist.

Routing of FidoNet mail

In a theoretical situation, a node would normally forward messages to a *hub*. The hub, acting as a distribution point for mail, might then send the message to the Net Coordinator. From there it may be sent through a Regional Coordinator, or to some other system specifically set up for the function. Mail to other zones might be sent through a Zone Gate.

For example, a FidoNet message might follow the path:

- 1:170/918 *(node)* to 1:170/900 *(hub)* to 1:170/0 *(net coordinator)* to 1:19/0 *(region coordinator)* to 1:1/0 *(zone coordinator)*. From there, it was distributed 'down stream' to the destination node(s).

Originally there was no specific relationship between network numbers and the regions they reside in. In some areas of FidoNet, most notably in Zone 2, the relationship between region number and network number are entwined. For example, 2:201/329 is in Net 201 which is in Region 20 while 2:2410/330 is in Net 2410 which is in Region 24. Zone 2 also relates the node number to the hub number if the network is large enough to contain any hubs. This effect may be seen in the nodelist by looking at the structure of Net 2410 where node 2:2410/330 is listed under Hub 300. This is not the case in other zones.

In Zone 1, things are much different. Zone 1 was the starting point and when Zones and Regions were formed, the existing nets were divided up regionally with no set formula. The only consideration taken was where they were located geographically in respect to the region's mapped outline. As net numbers got added, the following formula was used.

Region number X 20

Then when some regions started running out of network numbers, the following was also used.

Region number X 200

Region 19, for instance, contains nets 380-399 and 3800-3999 in addition to those that were in Region 19 when it was formed.

Part of the objective behind the formation of local nets was to implement cost reduction plans by which all messages would be sent to one or more hubs or hosts in compressed form (ARC was nominally standard, but PKZIP is universally supported); one toll call could then be made during off-peak hours to exchange entire message-filled archives with an out-of-town uplink for further redistribution.

In practice, the FidoNet structure allows for any node to connect directly to any other, and node operators would sometimes form their own toll-calling arrangements on an ad-hoc basis, allowing for a balance between collective cost saving and timely delivery. For instance, if one node operator in a network offered to make regular toll calls to a particular system elsewhere, other operators might arrange to forward all of their mail destined for the remote system, and those near it, to the local volunteer. Operators within individual networks would sometimes have cost-sharing arrangements, but it was also common for people to volunteer to pay for regular toll calls either out of generosity, or to build their status in the community.

This ad-hoc system was particularly popular with networks that were built on top of FidoNet. Echomail, for instance, often involved relatively large file transfers due to its popularity. If official FidoNet distributors refused to transfer Echomail due to additional toll charges, other node operators would sometimes volunteer. In such cases, Echomail messages would be routed to the volunteers' systems instead.

The FidoNet system was best adapted to an environment in which local telephone service was inexpensive and long-distance calls (or intercity data transfer via packet-switched networks) costly. Therefore, it fared somewhat poorly in Japan, where even local lines are expensive, or in France, where tolls on local calls and competition with Minitel or other data networks limited its growth.

Points

As the number of messages in Echomail grew over time, it became very difficult for users to keep up with the volume while logged into their local BBS. **Points** were introduced to address this, allowing technically savvy users to receive the already compressed and batched Echomail (and Netmail) and read it locally on their own machines.

To do this, the FidoNet addressing scheme was extended with the addition of a final address segment, the point number. For instance, a user on the example system above might be given point number 10, and thus could be sent mail at the address `1:170/918.10`.

In real-world use, points are fairly difficult to set up. The FidoNet software typically consisted of a number of small utility programs run by manually-adjusted scripts. Reading and editing the mail required either a "sysop editor" or a BBS be run locally.

In North America (Zone 1) points were used only briefly, and even then only to a limited degree. Dedicated offline mail reader programs such as Blue Wave, Squiggy and Silver Xpress (OPX) were introduced in the mid 1990s, and quickly rendered the point system obsolete. Many of these packages supported the QWK offline mail standard.

In other parts of the world, especially Europe, this was different. Contrary to North America where local calls usually are free, in Europe local calls are mostly metered and so there was an incentive to keep the duration of the calls as short as possible. Point software employs standard compression (ZIP, ARJ etc.) and so keeps the calls down to a few minutes a day at most.

In Europe (Zone 2) pointing became very popular. Many regions distribute a pointlist in parallel with the nodelist. The pointlist segments are maintained by Net- and Region Pointlist Keepers and the Zone Point List Keeper assembles them into the Zone pointlist. At the peak of FidoNet there were over 120,000 points listed in the Z2 pointlist. Listing points is on a voluntary basis and not every point is listed, so how many points there really were is anybody's guess. As of June 2006, there are still some 50,000 listed points. Most of them are in Russia and Ukraine.

Technical specifications

FidoNet contained several technical specifications for compatibility between systems. The most basic of all was *FTS-0001*, with which all FidoNet systems were required to comply as a minimal requirement. FTS-0001 defined:

- Handshaking - the protocols used by mailer software to identify each other and exchange meta information about the session.
- Transfer protocol *(XMODEM)* - the protocols to be used for transferring files containing FidoNet mail between systems.
- Message format - the standard format for FidoNet messages during the time which they were exchanged between systems.

Other specifications that were commonly used provided for *echomail*, different transfer protocols and handshake methods (*e.g.: Yoohoo/Yoohoo2u2, EMSI*), file compression, nodelist format, transfer over reliable connections such as the Internet (Binkp), and other aspects.

Zone mail hour

Since computer bulletin boards historically used the same telephone lines for transferring mail as were used for dial-in human users of the BBS, FidoNet policy dictates that at least one designated line of each FidoNet node must be available for accepting mail from other FidoNet nodes during a particular hour of each day.

"Zone Mail Hour," as it was named, varies depending on the geographic location of the node, and was designated to occur during the early morning. The exact hour varies depending on the time zone, and any node with only one telephone line is required to reject human callers. In practice, particularly in later times, most FidoNet systems tend to accept mail at any time of day when the phone line is not busy, usually during night.

FidoNet deployments

Although monolithic software that encompassed all required functions in one package is available (e.g. D'Bridge), most FidoNet deployments were designed in a modular fashion. A typical deployment would involve several applications that would communicate through shared files and directories, and switch between each other through carefully designed scripts or batch files.

Arguably the most important piece of software on a DOS-based Fido system was the **FOSSIL driver**, which was a small device driver which provided a standard way for the Fido software to talk to the modem. This driver needed to be loaded before any Fido software would work. An efficient FOSSIL driver meant faster, more reliable connections.

Mailer software was responsible for transferring files and messages between systems, as well as passing control to other applications, such as the BBS software, at appropriate times. The mailer would initially answer the phone and, if necessary, deal with incoming mail via FidoNet transfer protocols. If the mailer answered the phone and a human caller was detected rather than other mailer software, the mailer would exit, and pass control to the BBS software, which would then initialise for interaction with the user. When outgoing mail was waiting on the local system, the mailer software would attempt to send it from time to time by dialing and connecting to other systems who would accept and route the mail further. Due to the costs of toll calls which often varied between peak and off-peak times, mailer software would usually allow its operator to configure the optimal times in which to attempt to send mail to other systems.

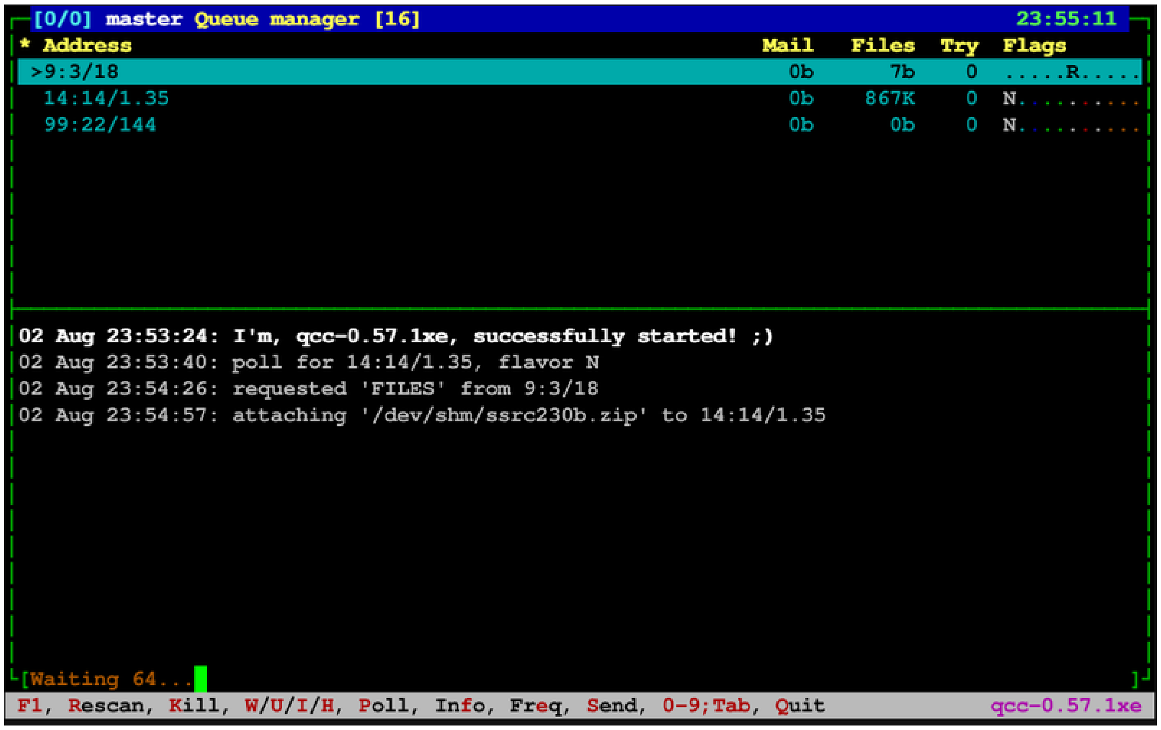
BBS software was used to interact with human callers to the system. BBS software would allow dial-in users to use the system's message bases and write mail to others, locally or on other BBSes. Mail directed to other BBSes would later be routed and sent by the mailer, usually after the user had finished using the system. Many BBSes also allowed users to exchange files, play games, and interact with other users in a variety of ways (i.e.: node to node chat).

A **scanner/tosser** application, such as FastEcho, FMail, TosScan and Squish, would normally be invoked when a BBS user had entered a new FidoNet message that needed to be sent, or when a mailer had received new mail to be imported into the local messages bases. This application would be responsible for handling the packaging of incoming and outgoing mail, moving it between the local system's message bases and the mailer's inbound and outbound directories. The scanner/tosser application would generally be responsible for basic routing information, determining which systems to forward mail to.

In later times, **message readers** or **editors** that were independent of BBS software were also developed. Often the System Operator of a particular BBS would use a devoted message reader, rather than the BBS software itself, to read and write FidoNet and related messages. One of the most popular editors in 2008 was GoldED+. In some cases FidoNet nodes, or more often FidoNet points, had no public bulletin board attached, and existed only for the transfer of mail for the benefit of the node's operator. Most nodes in 2009 had no BBS access, but only points, if anything.

The original *Fido BBS* software, and some other FidoNet-supporting software from the 1980s, is no longer functional on modern systems. This is for several reasons, including problems related to the Y2K bug. In some cases, the original authors have left the BBS or shareware community, and the software, much of which was closed source, has been rendered abandonware.

Several DOS based legacy FidoNet Mailers such as FrontDoor, Intermail, MainDoor and D'Bridge from the early 1990s can still be run today under Windows without a modem, by using the freeware NetFoss Telnet FOSSIL driver, and by using a Virtual Modem such as NetSerial. This allows the mailer to "Dial" an IP address or hostname via Telnet, rather than dialing a real POTS phone number. There are similar solutions for Linux such as MODEMU (modem emulator) which has limited success when combined with DOSEMU (DOS emulator). Mail Tossers such as FastEcho and FMail are still used today under both Windows and Linux/DOSEMU.



```
[0/0] master Queue manager [16]                                                   23:55:11
* Address                                                          Mail    Files  Try  Flags
 >9:3/18                                                            0b       7b    0   .....R....
  14:14/1.35                                                        0b     867K    0   N.........
  99:22/144                                                         0b       0b    0   N.........

02 Aug 23:53:24: I'm, qcc-0.57.1xe, successfully started! ;)
02 Aug 23:53:40: poll for 14:14/1.35, flavor N
02 Aug 23:54:26: requested 'FILES' from 9:3/18
02 Aug 23:54:57: attaching '/dev/shm/ssrc230b.zip' to 14:14/1.35

[Waiting 64...
F1, Rescan, Kill, W/U/I/H, Poll, Info, Freq, Send, 0-9;Tab, Quit                  qcc-0.57.1xe
```

File queue in qcc, the ncurses UI for qico. The addresses are made-up.

There are several modern Windows based FidoNet Mailers available today with source code, including Argus, Radius, and Taurus. MainDoor is another Windows based Fidonet mailer, which also can be run using either a modem or directly over TCP/IP. Two popular free and open source software FidoNet mailers for Unix-like systems are the binkd (cross-platform, IP-only, uses the binkp protocol) and qico (supports modem communication as well as the IP protocol of ifcico and binkp).

On the **hardware** side, Fido systems were usually well-equipped machines, for their day, with quick CPUs, high-speed modems and 16550 UARTs, which were at the time an upgrade. As a Fidonet system was usually a BBS, it needed to quickly process any new mail events before returning to its 'waiting for call' state. In addition, the BBS itself usually necessitated lots of storage space. Finally, a FidoNet system usually had at least one dedicated phoneline. Consequently, operating a Fidonet system often required significant financial investment, a cost usually met by the owner of the system.

FidoNet availability

While the use of FidoNet has dropped dramatically compared with its use up to the mid-1990s, it is still popular in Russia and former republics of the USSR. Some BBSes, including those that are now available for users with Internet connections via telnet, also retain their FidoNet netmail and echomail feeds.

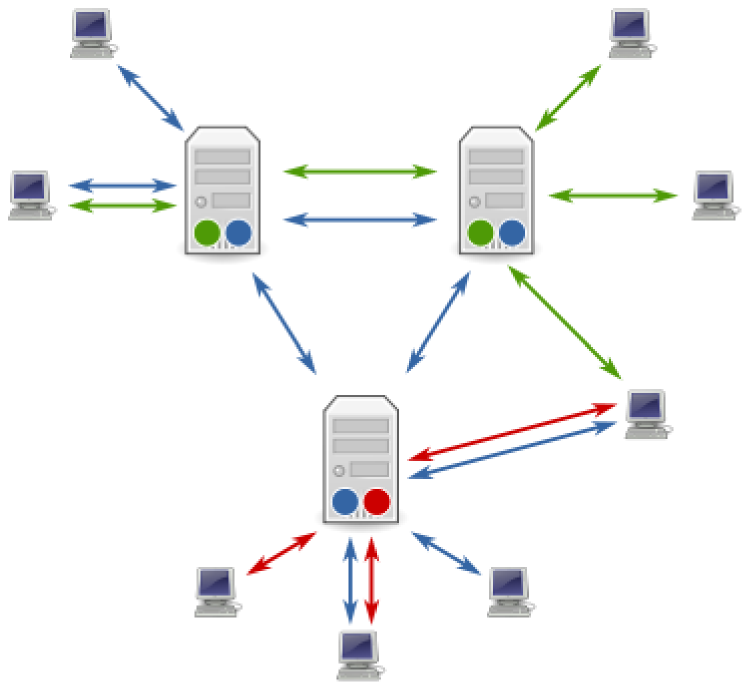
Some of FidoNet's echomail conferences are available via gateways with the Usenet news hierarchy using software like UFGate. There are also mail gates for exchanging messages between Internet and FidoNet. Widespread net abuse and e-mail spam on the Internet side has caused some gateways (such as the former 1:1/31 IEEE fidonet.org gateway) to become unusable or cease operation entirely.

FidoNews

FidoNews, first published in 1984, is the weekly newsletter of the FidoNet community. Throughout its history, it has been published by various people and entities, including the short-lived International FidoNet Association.

Chapter 11

Usenet



A diagram of Usenet servers and clients. The blue, green, and red dots on the servers represent the groups they carry. Arrows between servers indicate newsgroup group exchanges (feeds). Arrows between computers and servers indicate that a user is subscribed to a certain group and reads or submits articles.

Usenet is a worldwide distributed Internet discussion system. It developed from the general purpose UUCP architecture of the same name.

Duke University graduate students Tom Truscott and Jim Ellis conceived the idea in 1979 and it was established in 1980. Users read and post messages (called *articles* or *posts*, and collectively termed *news*) to one or more categories, known as *newsgroups*. Usenet resembles a bulletin board system (BBS) in many respects, and is the precursor to the various Internet forums that are widely used today. Usenet can be superficially regarded as a hybrid between email and web forums. Discussions are threaded, with modern news reader software, as with web forums and BBSes, though posts are stored on the server sequentially.

One notable difference between a BBS or web forum and Usenet is the absence of a central server and dedicated administrator. Usenet is distributed among a large, constantly changing conglomeration of servers that store and forward messages to one another in so-called *news feeds*. Individual users may read messages from and post messages to a local server operated by their Internet service provider, university, or employer.

Introduction

Usenet is one of the oldest computer network communications systems still in widespread use. It was conceived in 1979 and publicly established in 1980 at the University of North Carolina at Chapel Hill and Duke University, over a decade before the World Wide Web was developed and the general public got access to the Internet. It was originally built on the "poor man's ARPANET," employing UUCP as its transport protocol to offer mail and file transfers, as well as announcements through the newly developed news software such as A News. The name USENET emphasized its creators' hope that the USENIX organization would take an active role in its operation.

The articles that users post to Usenet are organized into topical categories called newsgroups, which are themselves logically organized into hierarchies of subjects. For instance, `sci.math` and `sci.physics` are within the `sci` hierarchy, for science. When a user subscribes to a newsgroup, the news client software keeps track of which articles that user has read.

In most newsgroups, the majority of the articles are responses to some other article. The set of articles which can be traced to one single non-reply article is called a thread. Most modern newsreaders display the articles arranged into threads and subthreads.

When a user posts an article, it is initially only available on that user's news server. Each news server, however, talks to one or more other servers (its "newsfeeds") and exchanges articles with them. In this fashion, the article is copied from server to server and (if all goes well) eventually reaches every server in the network. The later peer-to-peer networks operate on a similar principle; but for Usenet it is normally the sender, rather than the receiver, who initiates transfers. Some have noted that this seems an inefficient protocol in the era of abundant high-speed network access. Usenet was designed for the times when networks were much slower, and not always available. Many sites on the original Usenet network would connect only once or twice a day to batch-transfer messages in and out.

Usenet has significant cultural importance in the networked world, having given rise to, or popularized, many widely recognized concepts and terms such as "FAQ" and "spam".

The format and transmission of Usenet articles is similar to that of Internet e-mail messages. The difference between the two is that Usenet articles can be read by any user whose news server carries the group to which the message was posted, as opposed to email messages which have one or more specific recipients.

Today, Usenet has diminished in importance with respect to Internet forums, blogs and mailing lists. The difference, though, is that Usenet requires no personal registration with the group concerned, that information need not be stored on a remote server, that archives are always available, and that reading the messages requires not a mail or web client, but a news client. Many still use `alt.binaries` for data transfer.

ISPs, news servers, and newsfeeds

Many Internet service providers, and many other Internet sites, operate news servers for their users to access. ISPs that do not operate their own servers directly will often offer their users an account from another provider that specifically operates newsfeeds. In early news implementations, the server and newsreader were a single program suite, running on the same system. Today, one uses separate newsreader client software, a program that resembles an email client but accesses Usenet servers instead.

Not all ISPs run news servers. A news server is one of the most difficult Internet services to administer well because of the large amount of data involved, small customer base (compared to mainstream Internet services such as email and web access), and a disproportionately high volume of customer support incidents (frequently complaining of missing news articles that are not the ISP's fault). Some ISPs outsource news operation to specialist sites, which will usually appear to a user as though the ISP ran the server itself. Many sites carry a restricted newsfeed, with a limited number of newsgroups. Commonly omitted from such a newsfeed are foreign-language newsgroups and the `alt.binaries` hierarchy which largely carries software, music, videos and images, and accounts for over 99 percent of article data.

There are also Usenet providers that specialize in offering service to users whose ISPs do not carry news, or that carry a restricted feed.

Newsreader clients

Newsgroups are typically accessed with special client software that connects to a news server. Newsreader clients are available for all major operating systems. Modern mail clients or "communication suites" commonly also have an integrated newsreader. Often, however, these integrated clients are of low quality, compared to standalone newsreaders, and incorrectly implement Usenet protocols, standards and conventions. Many of these integrated clients, for example the one in Microsoft's Outlook Express, are disliked by purists because of their misbehavior.

With the rise of the World Wide Web (WWW), web front-ends have become more common. Web front ends have lowered the technical entry barrier requirements to that of one application and no Usenet NNTP server account. There are numerous websites now offering web based gateways to Usenet groups, although some people have begun filtering messages made by some of the web interfaces for one reason or another. Google Groups is one such web based front end and web browsers can access Google Groups via news: protocol links directly.

Moderated and unmoderated newsgroups

A minority of newsgroups are **moderated,** meaning that messages submitted by readers are not distributed directly to Usenet, but instead are emailed to the **moderators** of the newsgroup for approval. The moderator is to receive submitted articles, review them, and inject approved articles so that they can be properly propagated worldwide. Articles approved by a moderator must bear the Approved: header line. Moderators ensure that the messages that readers see in the newsgroup conform to the charter of the newsgroup, though they are not required to follow any such rules or guidelines. Typically, moderators are appointed in the proposal for the newsgroup, and changes of moderators follow a succession plan.

Historically, a mod.* hierarchy existed before Usenet reorganization. Now, moderated newsgroups may appear in any hierarchy.

Usenet newsgroups in the Big-8 hierarchy are created by proposals called a Request for Discussion, or RFD. The RFD is required to have the following information: newsgroup name, checkgroups file entry, and moderated or unmoderated status. If the group is to be moderated, then at least one moderator with a valid email address must be provided. Other information which is beneficial but not required includes: a **charter**, a rationale, and a moderation policy if the group is to be moderated. Discussion of the new newsgroup proposal follows, and is finished with the members of the Big-8 Management Board making the decision, by vote, to either approve or disapprove the new newsgroup.

Unmoderated newsgroups form the majority of Usenet newsgroups, and messages submitted by readers for unmoderated newsgroups are immediately propagated for everyone to see. Minimal editorial content filtering vs propagation speed form one crux of the Usenet community. One little cited defense of propagation is canceling a propagated message, but few Usenet users use this command and in fact some news readers don't offer cancellation commands, in part because article storage expires in relatively short order anyway.

Creation of moderated newsgroups often becomes a hot subject of controversy, raising issues regarding censorship and the desire of a subset of users to form an intentional community.

Technical details

Usenet is a set of protocols for generating, storing and retrieving news "articles" (which resemble Internet mail messages) and for exchanging them among a readership which is potentially widely distributed. These protocols most commonly use a flooding algorithm which propagates copies throughout a network of participating servers. Whenever a message reaches a server, that server forwards the message to all its network neighbors that haven't yet seen the article. Only one copy of a message is stored per server, and each server makes it available on demand to the (typically local) readers able to access that server. The collection of Usenet servers has thus a certain peer-to-peer character in that they share resources by exchanging them, the granularity of exchange however is on a different scale than a modern peer-to-peer system and this characteristic excludes the actual users of the system who connect to the news servers with a typical client-server application, much like an email reader.

RFC 850 was the first formal specification of the messages exchanged by Usenet servers. It was superseded by RFC 1036.

In cases where unsuitable content has been posted, Usenet has support for automated removal of a posting from the whole network by creating a cancel message, although due to a lack of authentication and resultant abuse, this capability is frequently disabled. Copyright holders may still request the manual deletion of infringing material using the provisions of World Intellectual Property Organization treaty implementations, such as the U.S. Online Copyright Infringement Liability Limitation Act.

On the Internet, Usenet is transported via the Network News Transfer Protocol (NNTP) on TCP Port 119 for standard, unprotected connections and on TCP port 563 for SSL encrypted connections which is offered only by a few sites.

Organization

The major set of worldwide newsgroups is contained within nine hierarchies, eight of which are operated under consensual guidelines that govern their administration and naming. The current *Big Eight* are:

- *comp.** – computer-related discussions (comp.software, comp.sys.amiga)
- *humanities.** – fine arts, literature, and philosophy (humanities.classics, humanities.design.misc)
- *misc.** – miscellaneous topics (misc.education, misc.forsale, misc.kids)
- *news.** – discussions and announcements about news (meaning Usenet, not current events) (news.groups, news.admin)
- *rec.** – recreation and entertainment (rec.music, rec.arts.movies)
- *sci.** – science related discussions (sci.psychology, sci.research)

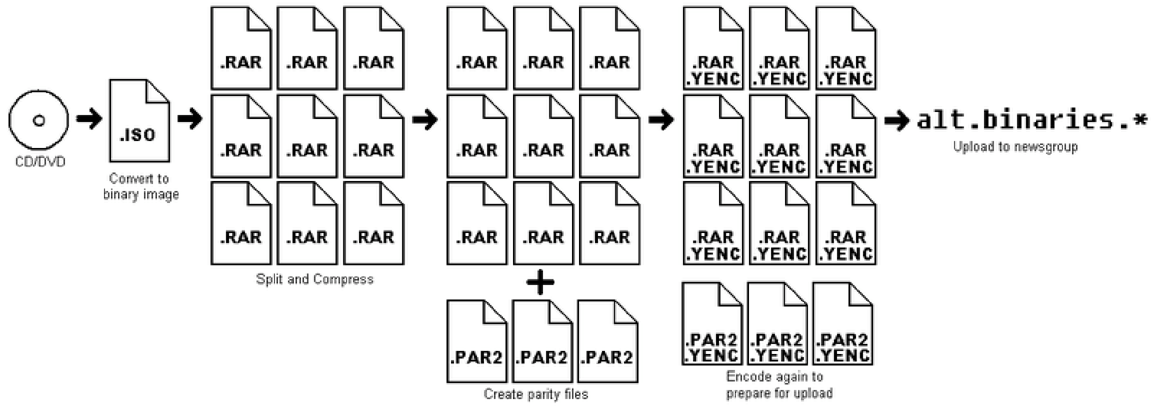
- *soc.** – social discussions (soc.college.org, soc.culture.african)
- *talk.** – talk about various controversial topics (talk.religion, talk.politics, talk.origins)

The alt.* hierarchy is not subject to the procedures controlling groups in the Big Eight, and it is as a result less organized. However, groups in the alt.* hierarchy tend to be more specialized or specific—for example, there might be a newsgroup under the Big Eight which contains discussions about children's books, but a group in the alt hierarchy may be dedicated to one specific author of children's books. Binaries are posted in alt.binaries.*, making it the largest of all the hierarchies.

Many other hierarchies of newsgroups are distributed alongside these. Regional and language-specific hierarchies such as *japan.**, *malta.** and *ne.** serve specific countries and regions such as Japan, Malta and New England. Companies administer their own hierarchies to discuss their products and offer community technical support. Microsoft has closed its newsserver as from June 2010, providing support for its products over forums now. Some users prefer to use the term "Usenet" to refer only to the Big Eight hierarchies; others include alt as well. The more general term "netnews" incorporates the entire medium, including private organizational news systems.

Informal sub-hierarchy conventions also exist. *.answers are typically moderated cross-post groups for FAQs. An FAQ would be posted within one group and a cross post to the *.answers group at the head of the hierarchy see by some as a refining of information in that news group. Some subgroups are recursive—to the point of some silliness in alt.*.

Binary content



A visual example of the many complex steps required to prepare data to be uploaded to usenet newsgroups. These steps must be done again in reverse to download data from usenet.

Usenet was originally created to distribute text content encoded in the 7-bit ASCII character set. With the help of programs that encode 8-bit values into ASCII, it became practical to distribute binary files as content. Binary posts, due to their size and oftendubious copyright status, were in time restricted to specific newsgroups, making it easier for administrators to allow or disallow the traffic.

The usenet is not only used to distribute (binary) content, but also to store backup data. This method is called usenet backup or uBackup.

The oldest widely used encoding method is uuencode, from the Unix UUCP package. In the late 1980s, Usenet articles were often limited to 60,000 characters, and larger hard limits exist today. Files are therefore commonly split into sections that require reassembly by the reader.

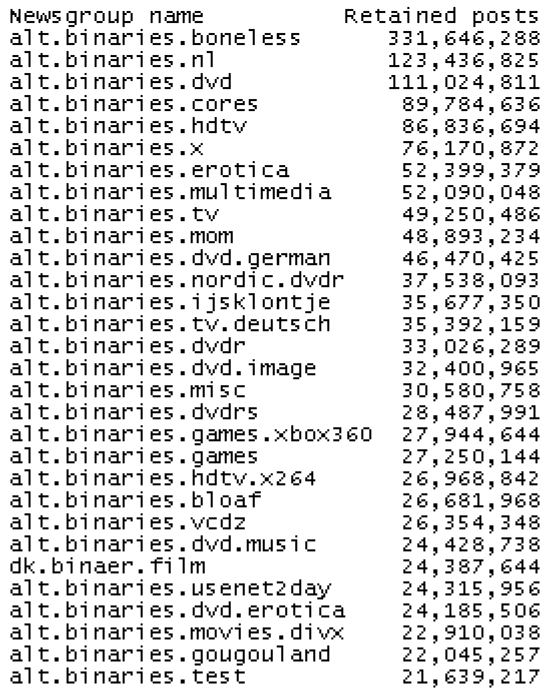
With the header extensions and the Base64 and Quoted-Printable MIME encodings, there was a new generation of binary transport. In practice, MIME has seen increased adoption in text messages, but it is avoided for most binary attachments. Some operating systems with metadata attached to files use specialized encoding formats. For Mac OS, both Binhex and special MIME types are used.

Other lesser known encoding systems that may have been used at one time were BTOA, XX encoding, BOO, and USR encoding.

In an attempt to reduce file transfer times, an informal file encoding known as yEnc was introduced in 2001. It achieves about a 30% reduction in data transferred by assuming that most 8-bit characters can safely be transferred across the network without first encoding into the 7-bit ASCII space.

The standard method of uploading binary content to Usenet is to first archive the files into RAR archives (for large files usually in 15 MB, 50 MB or 100 MB parts) then create Parchive files. Parity files are used to recreate missing data. This is needed often, as not every part of the files reaches a server. These are all then encoded into yEnc and uploaded to the selected binary groups.

Binary retention time



Newsgroup name	Retained posts
alt.binaries.boneless	331,646,288
alt.binaries.nl	123,436,825
alt.binaries.dvd	111,024,811
alt.binaries.cores	89,784,636
alt.binaries.hdtv	86,836,694
alt.binaries.x	76,170,872
alt.binaries.erotica	52,399,379
alt.binaries.multimedia	52,090,048
alt.binaries.tv	49,250,486
alt.binaries.mom	48,893,234
alt.binaries.dvd.german	46,470,425
alt.binaries.nordic.dvdr	37,538,093
alt.binaries.ijsklontje	35,677,350
alt.binaries.tv.deutsch	35,392,159
alt.binaries.dvdr	33,026,289
alt.binaries.dvd.image	32,400,965
alt.binaries.misc	30,580,758
alt.binaries.dvdrs	28,487,991
alt.binaries.games.xbox360	27,944,644
alt.binaries.games	27,250,144
alt.binaries.hdtv.x264	26,968,842
alt.binaries.bloaf	26,681,968
alt.binaries.vcdz	26,354,348
alt.binaries.dvd.music	24,428,738
dk.binaer.film	24,387,644
alt.binaries.usenet2day	24,315,956
alt.binaries.dvd.erotica	24,185,506
alt.binaries.movies.divx	22,910,038
alt.binaries.gougouland	22,045,257
alt.binaries.test	21,639,217

This is a list of the 30 biggest groups on Giganews on March 3, 2008, and is an example of the massive retention capabilities of a commercial usenet server.

Each newsgroup is generally allocated a certain amount of storage space for post content. When this storage has been filled, each time a new post arrives, old posts are deleted to make room for the new content. If the network bandwidth available to a server is high but the storage allocation is small, it is possible for a huge flood of incoming content to overflow the allocation and push out everything that was in the group before it. If the flood is large enough, the beginning of the flood will begin to be deleted even before the last part of the flood has been posted.

Binary newsgroups are only able to function reliably if there is sufficient storage allocated to a group to allow readers enough time to download all parts of a binary posting before it is flushed out of the group's storage allocation. This was at one time how posting of undesired content was countered; the newsgroup would be flooded with random garbage data posts, of sufficient quantity to push out all the content to be suppressed. This has been compensated by service providers allocating enough storage to retain everything posted each day, including such spam floods, without deleting anything.

The average length of time that posts are able to stay in the group before being deleted is commonly called the *retention time*. Generally the larger usenet servers have enough capacity to archive several weeks of binary content even when flooded with new data at the maximum daily speed available. A good binaries service provider must not only accommodate users of fast connections (3 megabit) but also users of slow connections

(256 kilobit or less) who need more time to download content over a period of several days or weeks.

Major NSPs don't delete any articles less than a year old, resulting in a retention time of more than 700 days.

Legal issues

While binary newsgroups can be used to distribute completely legal user-created works, open-source software, and public domain material, some binary groups are used to illegally distribute commercial software, copyrighted media, and pornography.

For example, some binary groups such as *alt.binaries.warez.** exist solely for the illegal distribution of commercial software.

ISP-operated Usenet servers frequently block access to all *alt.binaries.** groups to both reduce network traffic and to avoid related legal issues. Commercial Usenet service providers claim to operate as a telecommunications service, and assert that they are not responsible for the user-posted binary content transferred via their equipment. In the United States, Usenet providers can qualify for protection under the DMCA Safe Harbor regulations, provided that they establish a mechanism to comply with and respond to takedown notices from copyright holders.

Removal of copyrighted content from the entire Usenet network is a nearly impossible task, due to the rapid propagation between servers and the retention done by each server. Petitioning a Usenet provider for removal only removes it from that one server's retention cache, but not any others. It is possible for a special *post cancellation* message to be distributed to remove it from all servers, but many providers ignore cancel messages by standard policy, because they can be easily falsified and submitted by anyone. For a takedown petition to be most effective across the whole network, it would have to be issued to the origin server to which the content has been posted, but has not yet been propagated to other servers. Removal of the content at this early stage would prevent further propagation, but with modern high speed links, content can be propagated as fast as it arrives, allowing no time for content review and takedown issuance by copyright holders.

Establishing the identity of the person posting illegal content is equally difficult due to the trust-based design of the network. Like SMTP email, servers generally assume the header and origin information in a post is true and accurate. However, as in SMTP email, Usenet post headers are easily falsified so as to obscure the true identity and location of the message source. In this manner, Usenet is significantly different from modern P2P services; most P2P users distributing content are typically immediately identifiable to all other users by their network address, but the origin information for a Usenet posting can be completely obscured and unobtainable once it has propagated past the original server.

Also unlike modern P2P services, the identity of the downloaders is hidden from view. On P2P services a downloader is identifiable to all others by their network address. On Usenet, the downloader connects directly to a server, and only the server knows the address of who is connecting to it. Some Usenet providers do keep usage logs, but not all make this logged information casually available to outside parties such as the RIAA.

History

Newsgroup experiments first occurred in 1979. Tom Truscott and Jim Ellis of Duke University came up with the idea as a replacement for a local announcement program, and established a link with nearby University of North Carolina using Bourne shell scripts written by Steve Bellovin. The public release of news was in the form of conventional compiled software, written by Steve Daniel and Truscott.

Network

UUCP networks spread quickly due to the lower costs involved, and the ability to use existing leased lines, X.25 links or even ARPANET connections. By 1983, the number of UUCP hosts had grown to 550, nearly doubling to 940 in 1984.

As the mesh of UUCP hosts rapidly expanded, it became desirable to distinguish the Usenet subset from the overall network. A vote was taken at the 1982 USENIX conference to choose a new name. The name Usenet was retained, but it was established that it only applied to news. The name UUCPNET became the common name for the overall network.

In addition to UUCP, early Usenet traffic was also exchanged with Fidonet and other dial-up BBS networks. Widespread use of Usenet by the BBS community was facilitated by the introduction of UUCP feeds made possible by MS-DOS implementations of UUCP such as UFGATE (UUCP to FidoNet Gateway), FSUUCP and UUPC. The Network News Transfer Protocol, or NNTP, was introduced in 1985 to distribute Usenet articles over TCP/IP as a more flexible alternative to informal Internet transfers of UUCP traffic. Since the Internet boom of the 1990s, almost all Usenet distribution is over NNTP.

Software

Early versions of Usenet used Duke's A News software. At Berkeley an improved version called B News was produced by Matt Glickman and Mark Horton. With a message format that offered compatibility with Internet mail and improved performance, it became the dominant server software. C News, developed by Geoff Collyer and Henry Spencer at the University of Toronto, was comparable to B News in features but offered considerably faster processing. In the early 1990s, InterNetNews by Rich Salz was developed to take advantage of the continuous message flow made possible by NNTP versus the batched store-and-forward design of UUCP. Since that time INN development has continued, and other news server software has also been developed.

Public venue

Usenet was the initial Internet community and the place for many of the most important public developments in the commercial Internet. It was the place where Tim Berners-Lee announced the launch of the World Wide Web, where Linus Torvalds announced the Linux project, and where Marc Andreessen announced the creation of the Mosaic browser and the introduction of the image tag, which revolutionized the World Wide Web by turning it into a graphical medium.

Internet jargon and history

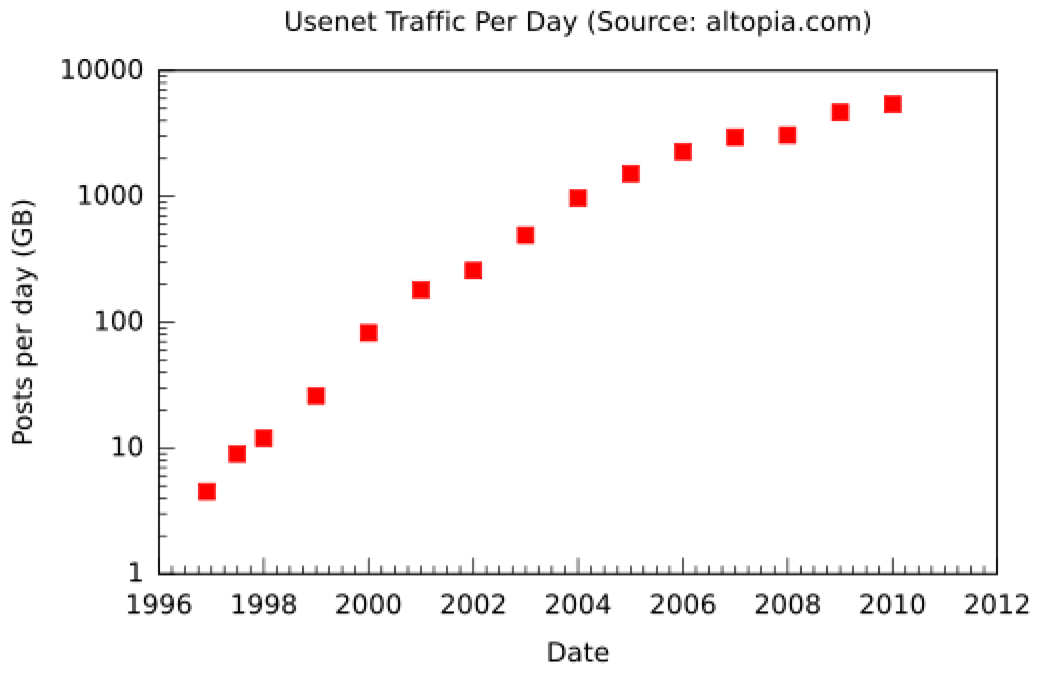
Many terms now in common use on the Internet—so-called "jargon"—originated or were popularized on Usenet. Likewise, many conflicts which later spread to the rest of the Internet, such as the ongoing difficulties over spamming, began on Usenet.
"Usenet is like a herd of performing elephants with diarrhea (sic). Massive, difficult to redirect, awe-inspiring, entertaining, and a source of mind-boggling amounts of excrement when you least expect it." -- Gene Spafford, 1992

Decline

By 2008, the Usenet's popularity was in steep decline, with some sources declaring it dead. In May 2010, Duke University, whose implementation had kicked off Usenet more than 30 years earlier, decommissioned its Usenet server, citing low usage and rising costs.

Usenet traffic today

Over time, the amount of Usenet traffic has steadily increased. Today, the average number of all text posts made in all Big-8 newsgroups is 1,800 new messages every hour, with an average of 25,000 messages per day. However, these averages are minuscule in comparison to the traffic in the binary groups. Much of this traffic increase reflects not an increase in discrete users or newsgroup discussions, but instead the combination of massive automated spamming and an increase in the use of *.binaries* newsgroups in which large files are often posted publicly. A small sampling of the change (measured in feed size per day) follows:



Daily Volume	Date	Source
4.5 GB	1996-12	Altopia.com
9 GB	1997-07	Altopia.com
12 GB	1998-01	Altopia.com
26 GB	1999-01	Altopia.com
82 GB	2000-01	Altopia.com
181 GB	2001-01	Altopia.com
257 GB	2002-01	Altopia.com
492 GB	2003-01	Altopia.com
969 GB	2004-01	Altopia.com
1.30 TB	2004-09-30	Octanews.net
1.38 TB	2004-12-31	Octanews.net
1.52 TB	2005-01	Altopia.com
1.34 TB	2005-01-01	Octanews.net
1.30 TB	2005-01-01	Newsreader.com
1.81 TB	2005-02-28	Octanews.net
1.87 TB	2005-03-08	Newsreader.com
2.00 TB	2005-03-11	Various sources
2.27 TB	2006-01	Altopia.com
2.95 TB	2007-01	Altopia.com
3.07 TB	2008-01	Altopia.com
3.80 TB	2008-04-16	Newsdemon.com
4.60 TB	2008-11-01	Giganews.com

4.65 TB	2009-01	Altopia.com
6.00 TB	2009-12	Newsdemon.com
5.42 TB	2010-01	Altopia.com
8.00 TB	2010-09	Newsdemon.com

In 2008, Verizon Communications, Time Warner Cable and Sprint Nextel signed an agreement with Attorney General of New York Andrew Cuomo to shut down access to sources of child pornography. Time Warner Cable stopped offering access to Usenet. Verizon reduced its access to the "Big 8" hierarchies. Sprint stopped access to the alt.* hierarchies. AT&T stopped access to the alt.binaries.* hierarchies. Cuomo never specifically named Usenet in his anti-child pornography campaign. David DeJean of *PC World* said that some worry that the ISPs used Cuomo's campaign as an excuse to end portions of Usenet access, as it is costly for the internet service providers. In 2008 AOL, which no longer offered Usenet access, and the four providers that responded to the Cuomo campaign were the five largest internet service providers in the United States; they had more than 50% of the U.S. ISP marketshare. On June 8, 2009, AT&T announced that it would no longer provide access to the Usenet service as of July 15, 2009.

AOL announced that it would discontinue its integrated Usenet service in early 2005, citing the growing popularity of weblogs, chat forums and on-line conferencing. The AOL community had a tremendous role in popularizing Usenet some 11 years earlier, with all of its positive and negative aspects. This change marked the end of the legendary Eternal September.

In August, 2009, Verizon announced that it would discontinue access to Usenet on September 30, 2009. In April 2010, Cox Communications announced (via email) that it would discontinue Usenet service, effective June 30, 2010. JANET(UK) announced it will discontinue Usenet service, effective July 31, 2010, citing Google Groups as an alternative. Microsoft announced that it would discontinue support for its public newsgroups (msnews.microsoft.com) from June 1, 2010, offering web forums as an alternative.

Primary reasons cited for the discontinuance of Usenet service by general ISPs include the decline in volume of actual readers due to competition from blogs, along with cost and liability concerns of increasing proportion of traffic devoted to file-sharing and spam on unused or discontinued groups.

At the same time, active discussion traffic has shifted away from ISPs, toward dedicated Usenet servers accessible via newsreader or the Web. Other sites host and archive Usenet newsgroups geared to specific topics via web interface, such as **compgroups.net** for the *comp.* groups hierarchy.

Archives

Public archives of Usenet articles have existed since the early days of Usenet, such as the system created by Kenneth Almquist in late 1982. Distributed archiving of Usenet posts was suggested in November 1982 by Scott Orshan, who proposed that "Every site should keep all the articles it posted, forever." Also in November of that year, Rick Adams responded to a post asking "Has anyone archived netnews, or does anyone plan to?" by stating that he was, "afraid to admit it, but I started archiving most "useful" newsgroups as of September 18." In June of 1982, Gregory G. Woodbury proposed an "Automatic access to archives" system that consisted of "automatic answering of fixed-format messages to a special mail recipient on specified machines."

In 1985, two news archiving systems and one RFC were posted to the net. The first system, called keepnews, by Mark M. Swenson of The University of Arizona, was described as "a program that attempts to provide a sane way of extracting and keeping information that comes over Usenet." The main advantage of this system was to allow users to mark articles as worthwhile to retain. The second system, YA News Archiver by Chuq Von Rospach, was similar to keepnews, but was "designed to work with much larger archives where the wonderful quadratic search time feature of the Unix ... becomes a real problem." The same Chuq Von Rospach in early 1985 posted a detailed RFC for "Archiving and accessing usenet articles with keyword lookup." This RFC described a program that could "generate and maintain an archive of usenet articles and allow looking up articles based on the article-id, subject lines, or keywords pulled out of the article itself." Also included was C code for the internal data structure of the system.

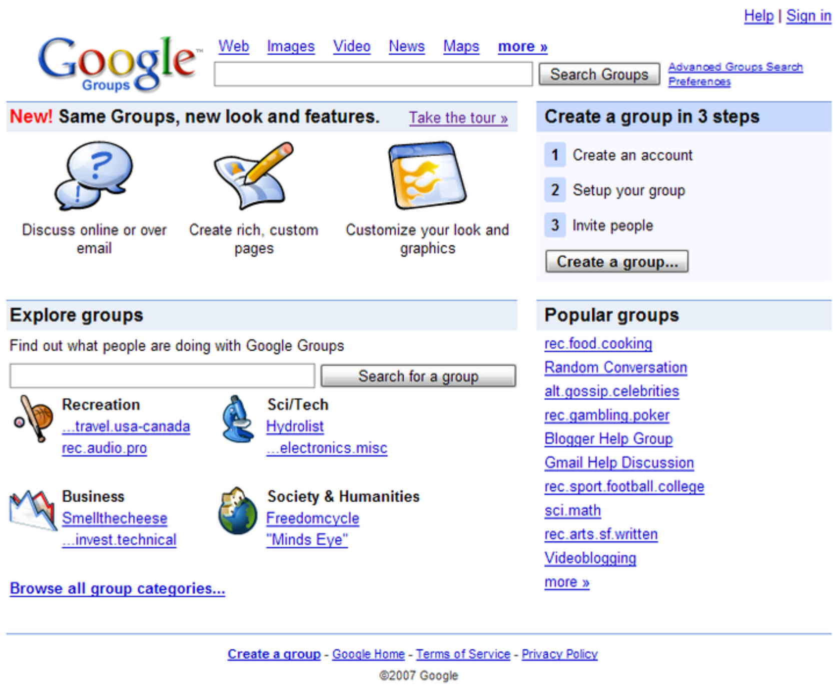
The desire to have a fulltext search index of archived news articles is not new either, one such request having been made in April 1991 by Alex Martelli who sought to "build some sort of keyword index for [the news archive]." In early May, Mr. Martelli posted a summary of his responses to the net, noting that the "most popular suggestion award must definitely go to 'lq-text' package, by Liam Quin, recently posted in alt.sources."

Today, the archiving of Usenet has led to a fear of loss of privacy. An archive simplifies ways to profile people. This has partly been countered with the introduction of the *X-No-Archive: Yes* header, which is itself seen as controversial.

Google Groups

Google Groups

Google groups



Help | Sign in

Web Images Video News Maps **more »**

Search Groups — Advanced Groups Search — Preferences

New! **Same Groups, new look and features.** Take the tour »

Discuss online or over email

Create rich, custom pages

Customize your look and graphics

Create a group in 3 steps

1 Create an account

2 Setup your group

3 Invite people

Create a group...

Explore groups

Find out what people are doing with Google Groups

Search for a group

Recreation
...travel.usa-canada
rec.audio.pro

Sci/Tech
Hydrolist
...electronics.misc

Business
Smellthecheese
...invest.technical

Society & Humanities
Freedomcycle
"Minds Eye"

Browse all group categories...

Popular groups

rec.food.cooking
Random Conversation
alt.gossip.celebrities
rec.gambling.poker
Blogger Help Group
Gmail Help Discussion
rec.sport.football.college
sci.math
rec.arts.sf.written
Videoblogging
more »

Create a group - Google Home - Terms of Service - Privacy Policy

©2007 Google

Google Groups screenshot

Developer(s)	Google
Written in	Python
Operating system	Cross-platform (web-based application)
Type	Newsgroups electronic mailing lists

Google Groups is a service from Google Inc. that supports discussion groups, including many Usenet newsgroups, based on common interests. Membership in Google Groups is free of charge and many groups are anonymous. Users can find discussion groups related to their interests and participate in threaded conversations, either through a web interface or by e-mail. They can also start new groups. Google Groups also includes an archive of Usenet newsgroup postings dating back to 1981 and supports reading and posting to Usenet groups. Users can also set up mailing list archives for e-mail lists that are hosted elsewhere.

History

In February 2001, Google acquired Deja News, which provided a search engine to access an archive of Usenet newsgroup articles. Users were then able to access these Usenet newsgroups through the new Google Groups interface. By the end of 2001 the archive had been supplemented with other archived messages dating back to 11 May 1981. These early posts from 1981-1991 were donated to Google by the University of Western Ontario, based on archives by Henry Spencer from the University of Toronto. Shortly after, Google released a new version, which allowed users to create their own (non-Usenet) groups.

In February 2006, Google modified the interface of Google Groups, adding profiles and post ratings.

In October 2010, Google announced it would be dropping support for welcome messages, pages, and files effective January 2011.

Kinds of groups hosted by Google

Google provides two distinct kinds of groups: traditional Usenet groups, and non-Usenet groups that are more similar to mailing lists. The latter type is accessible only by web or by e-mail, not by NNTP. The Google Groups user interface and help messages do not use a distinct name for mailing-list style groups, referring to both styles of group as "Google Groups."

Google recognizes the X-No-Archive header and displays messages containing it for only seven days, after which the article becomes no longer available to the public. Google also recognizes the "-- " Usenet signature delimiter, and removes the significant space at the end (thus, proper Usenet signatures can't be added to articles posted via Google Groups).

Notable interface features

Groups search

Google Search incorporates public groups into its results. Searches return the posts which most match the search query, and if any groups match, they will be displayed at the top of the results with a link to the Google Groups directory.

Profiles

Users may create public profiles which are linked from all of their posts.

Rating posts

A user can rate a post with 1 to 5 out of 5 stars. A post's rating is based on the average of all the user ratings it gets, and a thread's rating is based on the average rating of all the posts in the thread. Users may not rate their own posts.

Starring threads

Users may mark up to 200 threads as "starred" to track them centrally.

E-mail masking

To prevent scammers or spammers from harvesting e-mail addresses from a group, Google masks all e-mail addresses on its web interface by replacing up to the last 3 characters of the username with no less than three dots. To view the full e-mail address, a user must respond to a CAPTCHA challenge. E-mail addresses are only masked when viewing a Google Group or Usenet newsgroup through the web interface, never when subscribers receive messages by e-mail, nor when the Usenet articles are distributed to other servers. Google Groups does not allow users to obfuscate their own e-mail addresses.

Group web pages

The **group pages** were introduced in the beta version of October 5, 2006 (promoted from beta status on January 24, 2007). They can be edited by group members or group managers and can store files for download. On September 22, 2010 Google announced plans for turning off the group pages suggesting users to move their content to Google Docs or Google Sites. Starting in November 2010, the group pages will become read-only (allowing only viewing/downloading existing content) while in February 2011 they will be turned off completely.

Official Google Groups

Google has created several official help groups for some of its services, such as Gmail. In these groups, users can ask and answer questions about the relevant Google service. Each official group has a Google representative who occasionally responds to queries. Google representatives always have a blue G symbol in their nicknames.

Some official groups include:

- Google Groups Help Forum: was an official Google Groups help group until August 2, 2010, when it was archived (made read-only).
- Google Page Creator Discussion Group: an official Google Page Creator help group.

Google also uses Google Groups to host their Google Friends and Google Page Creator Updates mailing lists, which are announcement-only groups where only moderators can post.

There are also help forums, which appear to have different functionality from Google Groups:

- Gmail Help Forum: an official Gmail help forum.
- Google Talk Help Forum: an official Google Talk help forum.
- Google Base Help Forum: an official Google Base help forum.
- Google Web Search Help Forum: an official Google search help forum.
- Google Webmaster Help Forum: an official help forum for webmasters.
- AdWords Help Forum: an official Google AdWords help forum.
- Google Maps Forum: an official Google Maps help forum.

Criticism

The late Lee Rizor, also known as "Blinky the Shark," started the Usenet Improvement Project, a project which is highly critical of Google Groups and its users. The project aims to "make Usenet participation a better experience." They have accused Google Groups of turning a blind eye to an "increasing wave of spam" from its servers and of encouraging an Eternal September of "lusers" and "lamers" arriving in established groups *en masse*. The Usenet Improvement Project provides several killfile examples to block messages posted by Google Groups users in several newsreaders.

On 16 October 2003, John Wiley & Sons sent a letter to Google after discovering that copyrighted text from a book they published was made available for download on a Google group.

Slashdot and *Wired* contributors have criticized Google for its inattention to a search engine for Google Groups, leaving many older postings virtually inaccessible.

Outages

For about one week starting August 19, 2009, Google Groups did not send new articles to moderation for moderated Usenet groups such as `comp.lang.c++.moderated`, causing a severe reduction of traffic in those groups. A second such outage occurred from September 16–23, 2009. A Google representative acknowledged the problem on September 22, 2009 in a posting to the Google help forums.

Since November 24, 2009, outages still persist on Google Groups. Pages are being lost the moment they are published and e-mail notifications are yielding broken links. Even though the "Is Something Broken" support forum is full of complaints, Google has not acknowledged or acted to fix the problem.

Blocking

Google Groups has been blocked in Turkey since April 10, 2008 by the order of a court in Turkey. According to The Guardian, the court banned Google Groups following a libel complaint by Adnan Oktar against the service. Google Groups was the first of several websites to be blocked by the Turkish Government in rapid succession solely for including material which allegedly offended Islam.