Chapter 7
Bell Labs

Bell Labs (also known as Bell Laboratories, AT&T Bell Labs or Lucent Bell Labs) was founded by Western Electric Research Laboratories and the American Telephone and Telegraph company (AT&T) in 1925. Its goals were to explore fundamental areas of science likely to shape the telecoms industry.

It has a long and distinguished history of applied research, and its many inventions[1] include statistical process control which was developed by Walter Shewhart in the 1920s. Information theory and the field of cryptography were developed by Claude Shannon in the late 1940s. The invention of the transistor was a major milestone in the computing field, and it was developed by William Shockley and others in the late 1940s/early 1950s.

Error detecting and error correcting codes were developed by Richard Hamming in the late 1940s, and the UNIX operating system and the C programming language were developed by Dennis Ritchie and Kenneth Thompson in the early 1970s. The C++ programming language was developed by Bjarne Stroustroup in the early 1980s.

Eight Nobel Prizes have been awarded to Bell Labs researchers for their inventions, and the prestigious Turing Award has been won on two occasions by its researchers. Richard Hamming was awarded the Turing Prize in 1968 for his work on error detecting and error correcting codes, while Kenneth Thompson and Dennis Ritchie were awarded the Turing Prize for their work in the development of the UNIX operating system.

Bell Labs statisticians played an important role in the field of statistical process control, with Walter Shewhart developing a control chart to determine if process performance is under control and within the defined upper and lower control limits. Deming and Juran worked with Shewhart at Bell Labs in the 1920s, and these quality gurus developed important quality improvement programmes that later played an important role in transforming American and Japanese industry.

[1] It is not possible due to space limitations to discuss all of Bell Labs inventors and inventions, and the selection chosen inevitably reflects the bias of the author.

G. O'Regan, *Pillars of Computing*, DOI 10.1007/978-3-319-21464-1_7

Shannon's [Sha:48] developed a unified theory for communication as well as the mathematical foundations for the field. The key problem in communication theory is the reliable transmission of a message from a source point over a communications channel to a destination point. There may be noise in the channel that distorts the message, and the engineer wishes to ensure that the message received is that which has been sent. Shannon proposed two important theorems that establish the fundamental limits on communication. The first theorem deals with communication over a noiseless channel, and the second theorem deals with communication in a noisy environment.

Shannon is considered the father of modern cryptography with his influential 1949 paper [Sha:49] on secrecy systems. He established a theoretical basis for cryptography, and he defined the basic mathematical structures that underlie secrecy systems.

The transistor was invented by Bardeen, Brattan and Shockley in 1947, and they shared the 1956 Nobel Prize in Physics for this revolutionary invention. The transistor is a fundamental building block in electronics, and it acts as an electronic switch. It is used to implement Boolean functions in logic, and it consumes very little power and is much more reliable that the bulky vacuum tubes that preceded it.

Richard Hamming developed Hamming codes which are used for error detection and correction. Coding theory is a practical branch of mathematics that is concerned with the reliable transmission of information over communication channels. It allows errors to be detected and corrected, which is essential when messages are transmitted through a noisy communication channel. The channel could be a telephone line, radio link or satellite link. It is also applicable to storing information on storage systems such as the compact disc.

Researchers at Bell Labs have been active in the development of programming languages and operating systems. Kenneth Thompson and Denis Ritchie developed the UNIX operating system and C programming language, and Bjarne Stroustroup invented the C++ programming language. Bell Labs was also involved in the development of the first mobile phone system. There is more detailed information on Bell Labs in [Ger:13]. Next we discuss a selection of the Bell Labs inventors and their inventions.

7.1 Statistical Process Control

Walter Shewhart (Fig. 7.1) was a statistician at Bell Labs, and he is regarded as the founder of statistical process control (SPC). He developed the control chart (Fig. 7.2), which is a tool to monitor and control the process, with upper and lower limits for process performance specified. The process is under control if it is performing within these defined limits.

The Shewhart model (also known as the *PDCA Cycle*) is a systematic approach to problem solving and process control (Table 7.1). It consists of four steps (Fig. 7.3) which then repeat, and these steps are *plan*, *do*, *check* and *act*. Shwehart's ideas

Fig. 7.1 Walter Shewhart

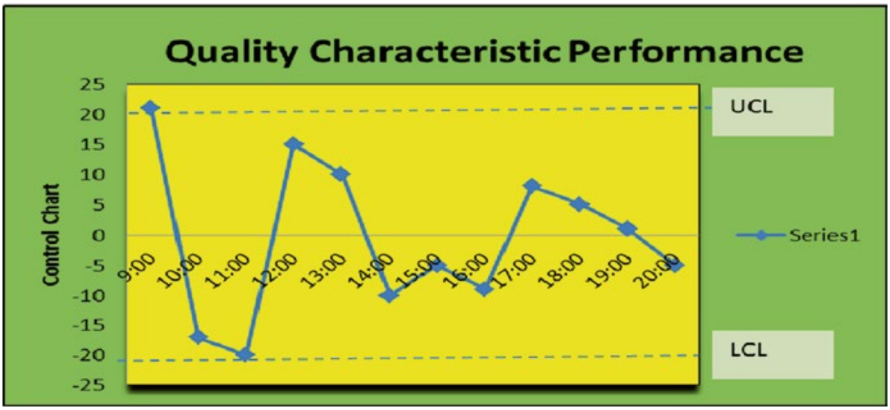


Fig. 7.2 Shewhart's control chart

Table 7.1 Shewhart cycle

Step	Description
Plan	This step identifies an improvement opportunity and outlines the problem or process improvement that will be addressed
	Select the problem to be addressed
	Describe current process
	Identify the possible causes of the problem
	Find the root cause of problems
	Develop an action plan to correct the root cause
Do	This step involves carrying out the improvements and it may involve a pilot of the proposed changes to the process
Check	This step involves checking the results obtained against the expected results to determine their effectiveness
Act	This step includes the analysis of the results to adjust process performance to achieve the desired results

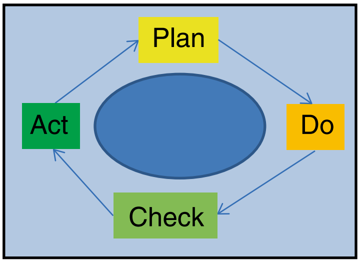


Fig. 7.3 Shewhart's PDCA cycle

were later applied to the Capability Maturity Model (CMM) as a way to control key software processes. Statistical process control plays an important role in process improvement.

Shewhart argued that quality and productivity improve as process variability is reduced. His influential book, *The Economic control of quality of manufactured product* [Shw:31] outlines the methods of statistical process control to reduce process variability. It prophesized that productivity would improve as process variability was reduced, and this was verified by Japanese engineers in the 1950s. Today, quality and quality improvement is fundamental to the success of a company.

7.2 Information Theory and Cryptography

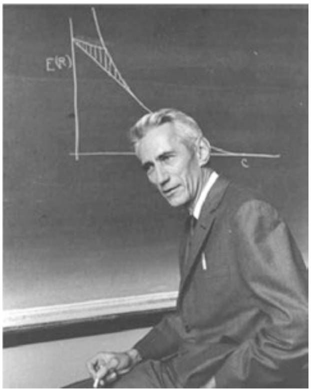
Claude Shannon (Fig. 7.4) was an American mathematician and engineer who made fundamental contributions to the computing field. He was the first person[2] to see the applicability of Boolean algebra to simplify the design of circuits and telephone routing switches. He showed that Boole's symbolic logic developed in the nineteenth century provided the perfect mathematical model for switching theory and for the subsequent design of digital circuits and computers.

His influential *Master's Thesis* [Sha:37] *is a key milestone in computing*, and it shows how to lay out circuits according to Boolean principles. It provides the theoretical foundation of switching circuits and *his insight of using the properties of electrical switches to do Boolean logic is the basic concept that underlies all electronic digital computers*.

Shannon realized that you could combine switches in circuits in such a manner as to carry out symbolic logic operations. This allowed binary arithmetic and more complex mathematical operations to be performed by relay circuits. He designed a circuit which could add binary numbers, and he later designed circuits which could make comparisons and thus is capable of performing a conditional statement. *This was the birth of digital logic and the digital computing age*.

[2] Victor Shestakov at Moscow State University also proposed a theory of electric switches based on Boolean algebra around the same time as Shannon. However, his results were published in Russian in 1941 whereas Shannon's were published in 1937.

Fig. 7.4 Claude Shannon



He moved to the Mathematics Department at Bell Labs in the 1940s and commenced work that would lead to the foundation of modern *Information Theory* and to the field of cryptography.

Shannon's work on Information theory was an immediate success with communications engineers. He established the theoretical basis for cryptography and defined the basic mathematical structures that underlie secrecy systems. He also made contributions to genetics and invented a chess-playing computer program in 1948.

7.2.1 Information Theory

The fundamental problem in information theory is to reproduce at a destination point, either exactly or approximately, the message that has been sent from a source point. The problem is that information may be distorted by noise, leading to differences between the received message, and the message that was originally sent. Shannon provided a mathematical definition and framework for information theory in *A Mathematical Theory of Communication* [Sha:48].

He proposed the idea of converting data (e.g., pictures, sounds or text) to binary digits: i.e., binary bits of information. The information is then transmitted over the communication medium. Errors or noise may be introduced during the transmission, and the objective is to reduce and correct them. The received binary information is then converted back to the appropriate medium.

There were several communication systems in use prior to Shannon's 1948 paper. These included the telegraph machine, the telephone, the AM radio and early television from the 1930s. These were all designed for different purposes and used various media. Each of these was a separate field with its own unique problems, tools and methodologies.

Shannon's classic 1948 paper [Sha:48] provided a unified theory for communication and a mathematical foundation for the field. The message may be in any

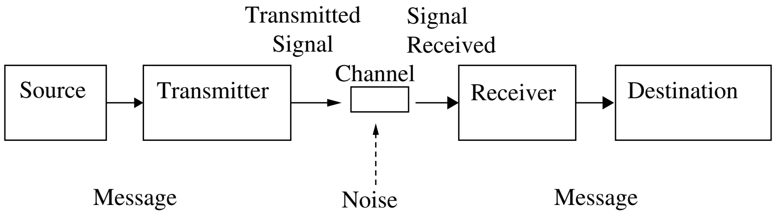


Fig. 7.5 Information theory

communications medium; e.g., television, radio and telephone. Information theory provides answers as to how rapidly or reliably a message may be sent from the source point to the destination point. Shannon identified five key parts of an information system (Fig. 7.5):

- Information Source
- Transmitter
- Channel
- Receiver
- Destination

He derived formulae for the information rate of a source and for the *c*apacity of a channel including noiseless and noisy cases. These were measured in bits per second, and he showed that for any information rate *R* less than the channel capacity *C*,[3] it is possible (by a suitable encoding) to send information at rate *R*, with an error rate less than any pre-assigned positive ε, over that channel.

Shannon's theory of information is based on probability theory and statistics. One important concept is that of *entropy*[4] which measures the level of uncertainty in predicting the value of a random variable. For example, the toss of a fair coin has maximum entropy, as there is no way to predict what will come next. Another words, a single toss of a fair coin has an entropy of one bit.

Shannon proposed two important theorems that establish the fundamental limits on communication. The first theorem (*Shannon's source coding theorem*) essentially states that *the transmission speed of information is based on its entropy or randomness*. It is possible to code the information (based on the statistical characteristics of the information source) and to transmit it at the maximum rate that the channel allows. Shannon's proof showed that an encoding scheme exists, but did not show how to construct one. This result was revolutionary as communication engineers at the time thought that the maximum transmission speed across a channel was related to other factors and not on the concept of information.

Shannon's *noisy-channel coding theorem* states that reliable communication is possible over noisy channels provided that the rate of communication is below a

[3] The channel capacity *C* is the limiting information rate (i.e., the least upper bound) that can be achieved with an arbitrarily small error probability. It is measured in bits per second.

[4] The concept of entropy is borrowed from the field of Thermodynamics.

certain threshold called the *channel capacity*. This result was revolutionary as it showed that a *transmission speed arbitrarily close to the channel capacity could be achieved with an arbitrarily low error*. The assumption at the time was that the error rate could only be reduced by reducing the noise level in the channel. Shannon showed that the desired transmission speed could be achieved by using appropriate encoding and decoding systems.

Shannon's theory also showed how to design more efficient communication and storage systems.

7.2.2 Cryptography

Shannon is considered the father of modern cryptography with his influential 1949 paper *Communication Theory of Secrecy Systems* [Sha:49]. He established a theoretical basis for cryptography and defined the basic mathematical structures that underlie secrecy systems (Fig. 7.6).

A secrecy system is defined to be a transformation from the space of all messages to the space of all cryptograms. Each possible transformation corresponds to encryption with a particular key, and the transformations are reversible. The inverse transformation allows the original message to be obtained provided that the key is known. A basic secrecy system is described in Fig. 7.6.

The first step is to select the key and to send it securely to the intended recipient. The choice of key determines the particular transformation to be used, and the message is then converted into a cryptogram (i.e., the encrypted text). The cryptogram is then transmitted over a channel (that is not necessarily secured from an enemy cryptanalyst) to the receiver, and the recipient uses the key to apply the inverse transformation. This allows the original message to be deciphered from the cryptogram.

The enciphering of a message is a functional operation. Suppose *M* is a message, *K* the key and *E* is the encrypted message then:

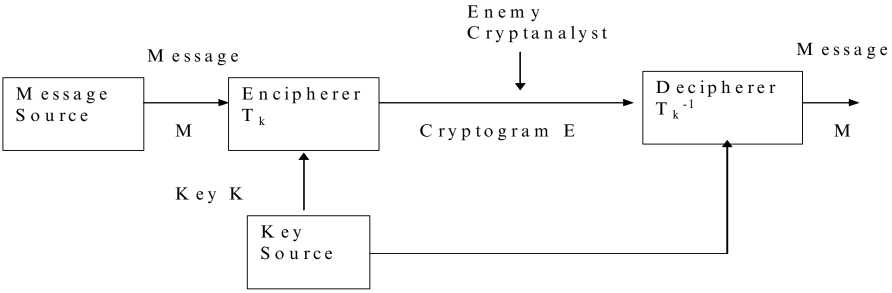


Fig. 7.6 Cryptography

$$E = f\left(M, K\right)$$

This is often written as a function of one variable $E = T_i\,M$ (where the index i corresponds to the particular key being used). It is assumed that there are a finite number of keys $K_1, \ldots, K_m$ and a corresponding set of transformations $T_1, T_2, \ldots, T_m$. Each key has a probability p_i of being chosen as the key. The encryption of a message M with key K_i is therefore given by:

$$E = T_i\,M$$

It is then possible to retrieve the original message from the received encrypted message by:

$$M = T_i^{-1}E$$

The channel may be intercepted by an enemy who will examine the cryptogram, and attempt to guess the key to decipher the message. For example, the cryptanalysts working at Bletchley Park in England during the Second World War regularly intercepted encrypted German naval messages being transmitted to their submarines in the Atlantic. They then used a machine that they had developed (called the "Bombe") to find the settings of the Enigma machine for that particular day. This allowed them to decipher the message and to protect Allied shipping in the Atlantic [ORg:11].

Shannon also showed that Vernam's cipher (also known as the *one time pad*) is a theoretically unbreakable cipher. Further, any unbreakable system must have essentially the same characteristics as the Vernam cipher. This cipler was invented by Gilbert Vernam at Bell Labs.

The Lorenz SZ 40/42 machine was used to encipher and decipher messages based on the Vernam cipher. These messages were sent by the German High Command in Berlin to Army Commands throughout occupied Europe. Tommy Flowers of the Post Office Research Station and the cryptanalysts at Bletchley Park developed the Colossus Mark I computer to crack the Lorenz codes, and this was work was invaluable around the time of the Normandy landings [ORg:12].

7.3 The Transistor

The early computers were large bulky machines taking up the size of a large room. They contained thousands of vacuum tubes (ENIAC contained over 18,000 vacuum tubes), and these tubes consumed large amounts of power and generated a vast quantity of heat. This led to problems with the reliability of the early computer, as several tubes burned out each day. This meant that machines were often non-functional for parts of the day, until the defective vacuum tube was identified and replaced.

There was therefore a need to find a better solution to vacuum tubes, and Shockley (Fig. 7.7) set up the solid physics research group at Bell Labs after the Second

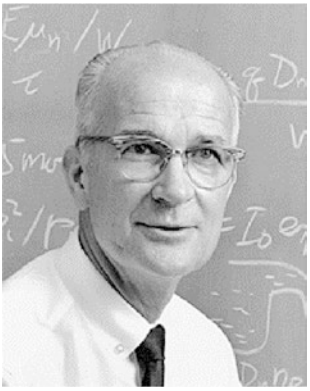


Fig. 7.7 William Shockley (Courtesy Chuck Painter, Stanford News Service)

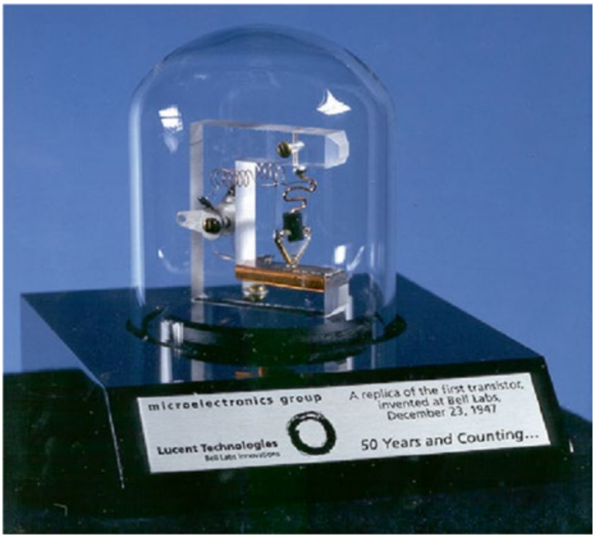


Fig. 7.8 Replica of transistor (Courtesy of Lucent Bell Labs)

World War. His goal was to find a solid-state alternative to the existing glass based vacuum tubes.

Shockley was born in England in 1910 to American parents, and he grew up at Palo Alto in California. He earned his PhD from Massachusetts Institute of Technology in 1936, and he joined Bell Labs shortly afterwards. The solid physics research team included John Bardeen and Walter Brattain, and they would later share the 1956 Nobel Prize in Physics with him for their invention of the transistor.

Their early research was unsuccessful, but by late 1947 Bardeen and Brattan succeeded in creating a point contact transistor independently of Shockley, who was working on a junction-based transistor. Shockley believed that the points contact transistor would not be commercially viable, and his junction point transistor was announced in mid-1951 with a patent granted later that year (Fig. 7.8). The junction point transistor soon eclipsed the point contact transistor and became dominant in the market place.

Shockley published a book on semiconductors in 1950 [Sho:50], and he resigned from Bell Labs in 1955. He formed Shockley Laboratory for Semiconductors (part of Beckman Instruments) at Mountain View in California. This company played an important role in the development of transistors and semiconductors, and several of its staff later formed semiconductor companies in the Silicon Valley area.

Shockley was the director of the company but his management style alienated several of his employees. This led to the resignation of eight key researchers in 1957 following his decision not to continue research into silicon-based semiconductors. This gang of eight went on to form Fairchild Semiconductors and other companies in the Silicon Valley area in the following years.

They included Gordon Moore and Robert Noyce, who founded Intel in 1968. National Semiconductors and Advanced Micro Devices were formed by other employees from Fairchild. Shockley Semiconductors and these new companies formed the nucleus of what became Silicon Valley.

7.4 Hamming Codes

Richard Hamming (Fig. 7.9) was born in Chicago in 1915 and he obtained his bachelor's degree in mathematics from the University of Chicago in 1937. He earned his PhD degree in mathematics from the University of Illinois in 1942. He worked on the Manhattan project at the Los Alamos Laboratory from 1945 to 1946, and he took a position at Bell Labs in 1946.

He became interested in the problem of the reliable transmission of information over a communication channel, and in particular in detecting whether an error has actually occurred in transmission, and algorithms for correcting such errors. He created a family of error correcting codes which are called *Hamming Codes*, and he introduced fundamental concepts such as *Hamming Distance*, *minimum Hamming Distance* and *Hamming Matrix*.

Fig. 7.9 Richard Hamming

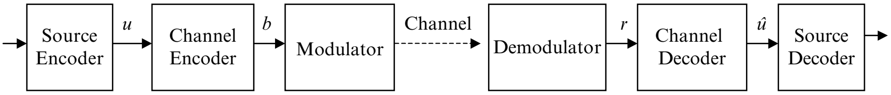


Fig. 7.10 Basic digital communication

Coding theory is a practical branch of mathematics that allows errors to be detected and corrected, and this is essential when messages are transmitted through a noisy communication channel. The channel could be a telephone line, radio link or satellite link, and coding theory is applicable to fixed line, mobile and satellite communications. It is also applicable to storing information on storage systems such as the compact disc.

Coding includes theory and practical algorithms for error detection and correction, and this is essential in modern communication systems that require reliable and efficient transmission of information.

An error correcting code encodes the data by *adding a certain amount of redundancy* to the message. This enables the original message to be recovered if a small number of errors have occurred. The extra symbols added are also subject to errors, as reliable transmission cannot be guaranteed in a noisy channel.

The basic structure of a digital communication system is shown in Fig. 7.10. It includes transmission tasks such as source encoding, channel encoding and modulation; and receiving tasks such as demodulation, channel decoding and source decoding.

The modulator generates the signal that is used to transmit the sequence of symbols b across the channel. The transmitted signal may be altered due to the fact that there is noise in the channel, and the signal received is demodulated to yield the sequence of received symbols r.

Therefore a channel code is employed to enable errors to be detected and corrected. The channel encoder introduces redundancy into the information sequence u, and the channel decoder uses the redundancy for error detection and correction. This enables the transmitted symbol sequence $\hat{u}$. to be estimated.

Coding theory is based on pure mathematics and it uses fundamental results from group theory, ring theory, vector spaces and finite field theory. There is a readable introduction to coding theory in Chap. 9 of [ORg:12].

7.4.1 Block Codes

A (n,k) block code is a code in which all codewords are of length n and all information words are of length k and $n>k$. The fundamental idea of the (n,k) block code is that the information word (i.e., a block of length k) is converted to a codeword (i.e., a block of length n).

Consider an information sequence u_0, u_1, u_2, … of discrete information symbols (usually binary 0 or 1). The information sequence is then grouped into blocks of length k as follows:

$$\underbrace{u_0u_1u_2\ldots u_{k-1}}\quad\underbrace{u_ku_{k+1}u_{k+2}\ldots u_{2k-1}}\quad\underbrace{u_{2k}u_{2k+1}u_{2k+2}\ldots u_{3k-1}}\cdots$$

Each block is of length k (i.e., the information words are of length k), and each information word is then encoded separately into codewords of length n. For example, the information word $u_0u_1u_2\ldots\ u_{k-1}$ is uniquely mapped to a code word $b_0b_1b_2\ldots.\ b_{n-1}$

$$\left(u_0u_1u_2\ldots u_{k-1}\right) \rightarrow \boxed{\text{Encoder}} \rightarrow \left(b_0b_1b_2\ldots b_{n-1}\right)$$

These code words are then transmitted across the communication channel and the received words are then decoded. The received word $r=(r_0r_1r_2\ \ldots\ r_{n-1})$ is then decoded into the information word $\hat{u}=(\hat{u}_0\hat{u}_1\hat{u}_2\ \ldots\hat{u}_{k-1})$.

$$\left(r_0r_1r_2\ldots r_{n-1}\right) \rightarrow \boxed{\text{Decoder}} \rightarrow \left(\hat{u}_0\hat{u}_1\hat{u}_2\ldots\hat{u}_{k-1}\right)$$

The decoding is done in two steps with the received n-block word r first decoded to an n-block codeword, which is then decoded into the k-block information word $\hat{u}$. The encoding, transmission and decoding of an (n,k) block is summarized in Fig. 7.11 below.

A generator matrix is typically employed to provide an efficient encoding and decoding mechanism [ORg:12].

7.4.2 Hamming Distance

The distance between two codewords $b=(b_0b_1b_2\ \ldots b_{n-1})$ and $b'=(b_0'b_1'b_2'\ \ldots\ b_{n-1}')$ measures how close the codewords b and b' are to each other. It is given by the *Hamming distance*:

$$\text{dist}\left(b,b'\right)=\left|\left\{i : b_i \neq b_i^{'}, 0 \leq i < n\right\}\right|$$

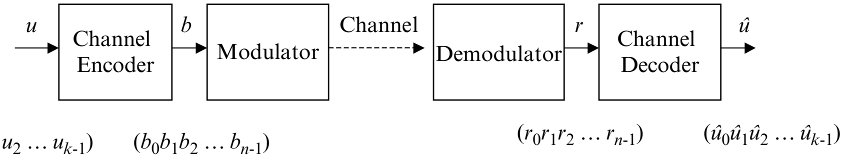

u
Channel Encoder
b
Modulator
Channel
Demodulator
r
Channel Decoder
û
$(u_0u_1u_2\ \ldots\ u_{k-1})$
$(b_0b_1b_2\ \ldots\ b_{n-1})$
$(r_0r_1r_2\ \ldots\ r_{n-1})$
$(\hat{u}_0\hat{u}_1\hat{u}_2\ \ldots\ \hat{u}_{k-1})$

Fig. 7.11 Encoding and decoding of an (n,k) block

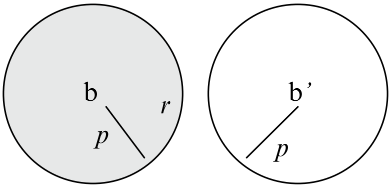


Fig. 7.12 Error correcting capability sphere

The *minimum Hamming distance* for a code **B** consisting of M codewords $b_1,\ldots, b_M$ is given by:

$$d = \min\left\{\text{dist}\left(b,b'\right) : \text{where } b \neq b' \text{ and } b,b' \in \mathbf{B}\right\}$$

The minimum Hamming distance offers a way to assess the error detection and correction capability of a channel code. Consider two codewords b and b' of an (n,k) block code $\mathbf{B}(n,k,d)$.

Then, the distance between these two codewords is greater than or equal to the minimum Hamming distance d, and so errors can be detected as long as the erroneously received word is not equal to a codeword different from the transmitted code word.

That is, the *error detection capability* is guaranteed as long as the number of errors is less than the minimum Hamming distance d, and so the number of detectable errors is $d-1$.

The distance between any two codewords is at least d, and so if the number of errors is less than $^d/_2$ then the received word can be correctly decoded to the codeword b. That is, the *error correction capability* is given by:

$$E_{\text{cor}} = \frac{d-1}{2}$$

An error-correcting sphere (Fig. 7.12) is employed to illustrate the error correction of a received word to the correct codeword b. This may be done when all received words are within the error-correcting sphere with radius p ($<^d/_2$).

If the received word r is different from b in less than $^d/_2$ positions, then it is decoded to b (as it is more than $^d/_2$ positions from the next closest codeword b'). That is, b is the closest codeword to the received word r (provided that the error- correcting radius is less than $^d/_2$).

7.5 UNIX and C

The UNIX operating system was developed by Ken Thompson, Dennis Ritchie and others at Bell Labs in the early 1970s (Fig. 7.13). It is a multitasking and multiuser operating system that is written almost entirely in C.

Fig. 7.13 Ken Thompson and Dennis Ritchie with President Clinton in 1999

Denis Ritchie was an American computer scientist who developed the C programming language at Bell Labs. He also co-developed the UNIX operating system with Ken Thompson. He was born in New York in 1941, and he earned a PhD in Physics and Applied Mathematics from Harvard University in 1967.

He joined Bell Labs in 1967 and designed and implemented the C programming language at Bell Labs in the early 1970s. The origin of this language is closely linked to the development of the UNIX operating system, and C was originally used for systems programming. It later became very popular for both systems and application programming, and it influenced later language development such as C++ and Java.

7.5.1 C Programming Language

Richie developed the C programming language at Bell Lab in 1972, and it became a popular programming language that is used widely in industry. It is a systems and applications programming language.

It was originally designed as the language to write the kernel for the UNIX operating system. It had been traditional up to then to write the operating system kernel in an assembly language, and the use of a high-level language such as C was a paradigm shift. This led to C's use as a systems programming language on several other operating systems (e.g., Windows and Linux), and C also influenced later language development. The C programming language is described in detail in [KeR:78].

The language provides high level and low-level capabilities, and a C program that is written in ANSI C is quite portable. It may be compiled for a wide variety of computer platforms and operating systems (with minimal changes to the source code). C is a procedural programming language, and it includes conditional statements such as the *if statement*; the *switch statement*; iterative statements such as the *while* statement and *do* statement; and the assignment statement which is specified by "=".

The language includes several pre-defined data types including integers and floating point numbers.

int	(integer)
long	(long integer)
float	(floating point real)
double	(double precisionreal)

It allows more complex data types to be created using the concept of a structure (s*truct*). It allows the use of pointers to access memory locations, and this allows the memory locations to be directly referenced and modified.

C is a block structured language, and a program is structured into functions (or blocks). Each function block contains its own variables and functions. A functions may call itself (i.e., *recursion is allowed*).

One key criticism of C is that it is easy to make errors in C programs and to thereby produce undesirable results. For example, one of the easiest mistakes to make is to accidently write the assignment operator "=" for the equality operator "==". This totally changes the meaning of the original statement.

The philosophy of C to allow statements to be written as concisely as possible, and this is potentially dangerous.[5] The use of pointers potentially leads to problems as uninitialised pointers may point anywhere in memory, and the program may potentially overwrite anywhere in memory.

Therefore, the effective use of C requires experienced and disciplined programmers; well documented source code; and formal peer reviews of the source code by other team members to ensure that the code is readable and easy to maintain, as well as providing confidence in its correctness.

7.5.2 UNIX

The UNIX operating system was developed by Ken Thompson, Dennis Ritchie and others at Bell Labs in the early 1970s. It is a multitasking and multiuser operating system that is written almost entirely in C. UNIX arose out of work by Massachusetts Institute of Technology, General Electric and Bell Labs on the development of a general timesharing operating system called *Multics*.

[5] It is easy to write a one line C program that is incomprehensible. The maintenance of poorly written code is a challenge unless programmers follow good programming practice. This discipline needs to be enforced by formal reviews of the source code.

Bell Labs decided in 1969 to withdraw from the Multics project and to use General Electric's GECOS operating system. However, several of the Bell Lab researchers decided to continue the work on a smaller scale operating system using a Digital PDP-7 minicomputer. They later used a PDP-11 minicomputer, and the result of their work was UNIX. It became a popular and widely used operating system that was used initially by universities and the US government, but it later became popular in industry.

It is a powerful and flexible operating system and is used on a variety of machines from micros to supercomputers. It is designed to allow several users access the computer at the same time and to share its resources, and it offers powerful real time sharing of resources.

It includes features such as *multitasking* which allows the computer to do several things at once; *multiuser* capability which allows several users to use the computer at the same time; *portability* of the operating system which allows it to be used on several computer platforms with minimal changes to the code; and a collection of tools and applications.

There are three levels of the UNIX system: *kernel*, *shell*, and *tools and applications*. For more detailed information on UNIX see [Rob:05].

7.6 C++ Programming Language

The C++ programming language was developed by Bjarne Stroustrup (Fig. 7.14) at Bell Labs in the early 1980s. It was designed as an object-oriented language, and it provides a significant extension to the capabilities of the C programming language.

Fig. 7.14 Bjarne Stroustrup

Stroustroup was born in Aarhus, Denmark in 1950, and he earned his PhD degree in Computer Science from the University of Cambridge in 1979. His PhD was concerned with the design of distributed systems.

He moved to New Jersey in 1979 and joined the Computer Science research center at Bell Labs. He developed the C++ programming language at Bell Labs, and C++ is a widely used object-oriented language.

Stroustroup was the head of the Large Scale Programming Research Department from its creation until 2002, when he moved to the University of Texas. He developed the C++ programming language in 1983 as an object-oriented extension of the C programming language.

C++ was designed to use the power of object-oriented programming and to maintain the speed and portability of C. It provides a significant extension of C's capabilities, but it does not force the programmer to use the object-oriented features of the language.

A key difference between C++ and C is in the concept of a class. A *class* is an extension to the concept of a structure which is used in C. The main difference is that while a C data *structure* can hold only *data, a* C++ *class* may hold both *data* and *function*s.

An *object* is an instantiation of a class: i.e., the class is essentially the type, whereas an object is essentially a variable of that type. Classes are defined in C++ by using the keyword *class*.

The members of a class may be either data or function declarations, and an access specifier is used to specify the access rights for each member (e.g., *private*, *public* or *protected*).

Private members of a class are accessible only by other members of the same class; public members are accessible from anywhere where the object is visible; and protected members are accessible by other members of same class and also from members of their derived classes.

7.7 Advanced Mobile Phone System

Bell Labs played an important role (with Motorola) in the development of the analog mobile phone system in the United States. It developed a system in the mid-1940s that allowed mobile users to place and receive calls from automobiles, and Motorola developed mobile phones for automobiles. However, these phones were large and bulky and they consumed a lot of power. A user needed to keep the automobile's engine running in order to make or receive a call.

Bell Labs first proposed the idea of a cellular system back in the late 1940s, when they proposed hexagonal rings for mobile communication. Large geographical areas were divided into cells, where each cell had its own base station and channels. The available frequencies could be used in parallel in different cells without disturbing each other. Mobile telephone could now, in theory, handle a large number of subscribers. However, it was not until the late 1960s that Bell prepared a detailed plan for the cellular system.

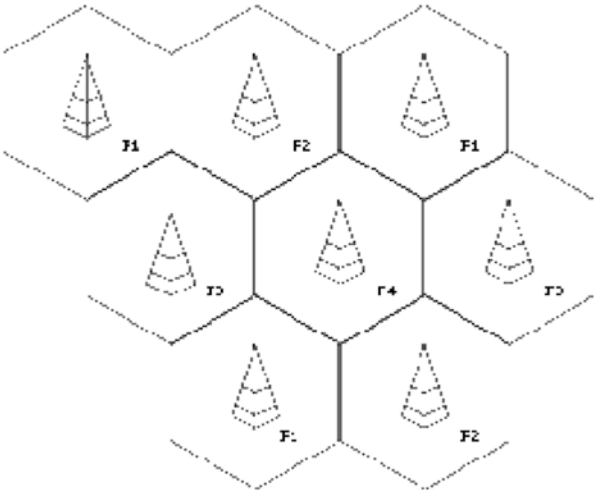


Fig. 7.15 Frequency reuse in cellular networks

The Advanced Mobile Phone System (AMPS) standard was developed by Bell Labs from 1968 to 1983 (Fig. 7.15), and it was introduced into the United States in 1983. Motorola and other telecommunication companies designed and built phones for this cellular system. AMPS uses separate frequencies (or channels) for each conversation and requires considerable bandwidth for a large number of users.

Motorola was the first company to develop a hand-held mobile phone. This was the DynaTAC (or *Brick*), and the Motorola team was led by Martin Cooper (Fig. 23.1). Cooper made the first mobile call to Joel Engels of Bell Labs. The phone weighed over a kilogram and it had a talk time of about 30 min. Further, it took over 10 h to recharge.

AMPS is the first generation of cellular technology, and so it has several weaknesses when compared to today's cellular systems. It was susceptible to static or noise, and there was no protection from eavesdropping with a scanner.

AMPS was later replaced by Global System for Mobile Communication (GSM) and Code Division Multiple Access (CDMA) technologies.