INTRODUCTION 1

1.1 EARLY BEGINNINGS

The problem of designing efficient and unambiguous communication protocols existed long before the first computers were built. There is a long history of attempts to construct systems for transferring information quickly over long distances. From a protocol designer's point of view, the mishaps that were caused by misinterpreted communications are fascinating. Of course, the problems of the early systems were not always documented as diligently as the features.

BEACONS AND ALARUMS

Anything that is detectable over a large distance is a potential means of communication. In the play *Agamemnon* from 458 B.C., for instance, Aeschylus describes in detail how fire signals were used, supposedly, to communicate the fall of Troy to Athens over a distance of more than 300 miles. But the number of different messages that can be transferred by a single big fire is limited. A detailed account of that problem was given by the Greek historian Polybius in the 2nd century B.C.[1] It is probably one of the first explicit descriptions of data transmission methods. Polybius starts by explaining why a signaling method is useful in the first place.

> *''It is evident to all that in every matter, and especially in warfare, the power of acting at the right time contributes very much to the success of enterprises, and fire signals are the most efficient of all the devices which aid us to do this. For they show what has recently occurred and what is still in the course of being done, and by means of them anyone who cares to do so even if he is at a distance of three, four or even more days' journey can be informed. So that it is always surprising how help can be brought by means of fire messages when the situation requires it.''*

The use of fire signals must have been commonplace in Polybius' days. But, there were a few problems to solve.

1. *The Histories*, Book X, Chapter 43. The translation is by W.R. Patton and was published by Harvard University Press in 1925.

''Now in former times, as fire signals were simple beacons, they were for the most part of little use to those who used them. For the service should have been performed by signals previously determined upon, and as facts are indefinite, most of them defied communication by fire signals. To take the case I just mentioned, it was possible for those who had agreed on this to convey information that a fleet had arrived at Oreus, Peparethus, or Chalcis, but when it came to some of the citizens having changed sides or having been guilty of treachery or a massacre having taken place in the town, or anything of the kind, things that often happen, but cannot all be foreseen — ***and it is chiefly unexpected occurrences which require instant consideration and help*** *— all such matters defied communication by fire signal. For it was quite impossible to have a preconcerted code for things which there was no means of foretelling.''*

The crucial observation is the part in bold. Throughout this book we will see that it is still a problem. It is the unexpected sequences of events that lead to protocol failures, and the hardest problem in protocol design is precisely that we must try to expect the unexpected.

Polybius continues his account with a description of a new signaling method that he believed solved the communication problem. It is remarkably sophisticated, though it only partly solves the problem. The new system used two sets of five torches. By lighting between one and five torches in each set, a total of 5^2 characters could be encoded, sufficient to transmit arbitrary messages as a sequence of encoded letters.

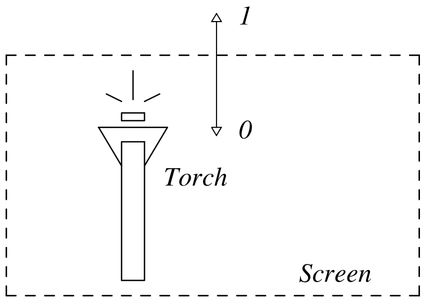

Figure 1.1 — Torch Telegraph

As shown in Figure 1.1, the torches could be used to send a binary torch code. A torch could be made visible to the remote receiver by raising it above a screen, and it could be hidden by lowering it. Polybius describes the torch code as follows.

''We take the alphabet and divide it into five parts, each consisting of five letters. There is one letter fewer in the last division, but it makes no practical difference. Each of the two parties who are about to signal to each other must now get ready five tablets and write one division of the alphabet on each tablet, and then come to an agreement that the man who is going to signal is in the first place to raise two torches and wait until the other replies by doing the same. This is for the purpose of conveying to each other that they are both at attention. These torches having been lowered, the dispatcher of the message will now raise the first set of torches on the left side indicating which tablet is to be consulted, i.e., one torch if it is the first, two if it is the second, and so on. Next he will raise the second set on the right on the same principle

to indicate what letter of the tablet the receiver should write down.''

No real improvements over Polybius' telegraph were made for almost twenty centuries, though there was no lack of inferior alternatives. In 1684, the English scientist Robert Hooke described a rather clumsy optical system that worked with large wooden characters. The characters could be displayed at a signaling station and observed from a distance with a telescope.[2] As far as we know, it was never put into practice.

In 1796, the German G. Huth invented an equally unsuccessful system that he named ``telephone.'' The idea was to place men with ``speaking tubes'' on roof tops and have them shout messages to each other. In fact, Huth's idea had been tried before. Alexander the Great (356-323 B.C.) is said to have used a twelve foot megaphone to shout commands to his armies from nearby hills. Not surprisingly, Polybius did not spend much time discussing this system.

Another remarkable device was used during the American Revolutionary War (1775-1783). It consisted of a pole from which any combination of three different objects could be displayed. With a barrel, a flag, and a basket, $2^3 - 1$ different messages could be transmitted, though obviously not in very rapid succession.

OPTICAL SYSTEMS

The first successful pre-electric telegraph system was developed by the French engineer Claude Chappe in 1793. His system consisted of large wooden constructions built on hill tops or church towers and was operated by civil servants equipped with telescopes. The semaphore had three movable parts, *regulator* and two *indicators*, as illustrated in Figure 1.2. The regulator was roughly 15 ft long, the indicators measured approximately 7 by 1 ft each.

It is not clear from the reports what the precise signaling ``alphabet'' was or how it was encoded in the positions of regulator and indicators. The semaphore arms could be moved only in 45° increments. Theoretically, with three movable parts, each semaphore could be set in 256 (8×8×4) different positions. Particularly confusing combinations were not used, for instance positions where the indicators duplicate the angle of the regulator. Reportedly, about half of the valid semaphore positions were used to encode digits, punctuation marks, upper- and lower-case letters, and the other half were used for special control codes. The civil servants were hired to read the semaphore position from the neighboring stations and to copy it onto their own semaphore to relay messages.

At the peak of its success, shortly before electric telegraphs took over, Chappe's system had grown into a complete network of no less than 556 semaphore stations covering more than 3000 miles and reaching nearly every part of France. Little is known

2. The telescope was also a recent invention at the time. It was described by Galileo in *Siderius Nuncius* (The Starry Messenger) in 1610.

about the specific operating procedures employed or the coordination problems that must have plagued the semaphore operators. What, for instance, was a semaphore operator supposed to do when two messages came in simultaneously from opposite directions?

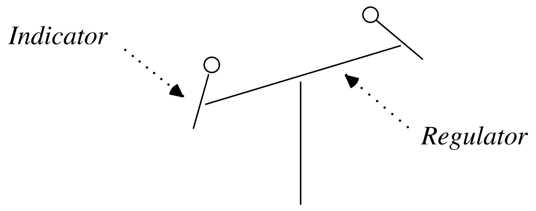

Figure 1.2 — Chappe's Semaphore

Almost every country had one or more variations of Chappe's optical telegraph in this period. The British admiralty, for instance, used a six-shutter semaphore designed by a Lord George Murray. Each shutter could be either opened or closed to transmit a message: a 6-bit binary code. The use of control messages is also documented for this system. All six shutters closed was used to signal *not ready*, all six shutters open meant *ready to send*.

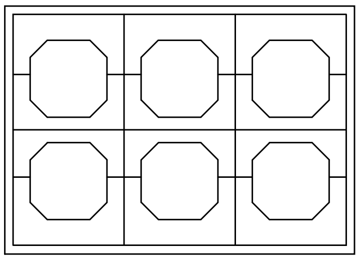

Figure 1.3 — George Murray's Six-Shutter Telegraph

A similar system, using ten shutters, was developed in Sweden. The Swedish system is documented in detail in a publication of its inventor, the Swedish Chancery Secretary A.N. Edelcrantz, called *Avhandling om Telegrapher* published in 1796. A coding table, based on a simple system for assigning numbers to shutter positions, was included.

> *''All telegraphic correspondence is started with a signal indicating that you want to speak, or a speak sign, which is left up until the receiver has given the corresponding alert sign. (...) When this is done, the speak sign is taken down and the first signal in the message is given. The receiver then takes down the alert sign and repeats the signal from the sender to show that it has read it correctly. The same procedure is repeated for all signals in the message.*

In 1796 this telegraph connected Stockholm and Åland. One shutter telegraph station, built in Furusund in 1836, has survived and can still be visited today (see also

Bibliographic Notes). The signaling codes used on the Swedish system include codes for session control (start, stop), error control, flow control (repeat), rate control (slower, faster), and even a negative acknowledgment, which was named appropriately ''cannot see.''

The transmission speed of the optical telegraphs varied. On Chappe's telegraph the semaphore position was changed once every 15-20 seconds. With a subset of 128 possible symbols (or 7 bits of information) this gave a transmission speed of roughly 0.5 bits/sec. The 10-bit code of the Swedish shutter telegraph was changed every 8-10 seconds, and the 6-bit code of the British system every 5 seconds, both giving a signaling speed of approximately 1 bit/sec.

The visibility of the semaphores must have been another concern of the operators. On an average of twenty days per year, for example, weather conditions prevented the usage of a shutter semaphore that connected five cities in The Netherlands between 1831 and 1839.

After 1840, the electric telegraph finally proved to be faster, more reliable, and less conspicuous than optical telegraphs.

ELECTROMAGNETISM

The principle of an electric telegraph was described as early as 1753 by a mysterious ''C.M.'' in a letter to the *Scots' Magazine*.[3] The identity of the author has never been fully established. Some sources say that the initials are those of a Charles Marshall from Renfrew (the letter was mailed from Renfrew). Others claim that the author was someone called Charles Morrison of Greenock. The letter describes an electric telegraph with a number of parallel wires: one for each different code, or character, to be transmitted. Small pithballs were placed at the receiver near the terminals of each wire. The sender could place a static electric charge on one of the wires (by discharging a Leyden jar) and cause the corresponding pithball at the receiver to move.

Shortly after 1830 a new insight into electromagnetic induction was obtained through the work of Michael Faraday in England and Joseph Henry in the United States. In England, the principle was used in 1837 by William Cooke in the construction of the first electric telegraph. Cooke used an electric charge to deflect a compass needle in a small magnetic field at the receiving instrument. The idea for such a ''needle telegraph'' was perfected by Cooke in cooperation with Sir Charles Wheatstone. It was patented in 1837 as a *Method of Giving Signals and Sounding Alarums at Distant Places by Means of Electric Currents Transmitted through Metallic Circuits*. In the United States similar work was done by Samuel Morse and Theodore Vail.

3. The *Scots' Magazine*, February 17, 1753, Vol. XV, p. 73. The letter was titled *An expeditious method for conveying intelligence*.

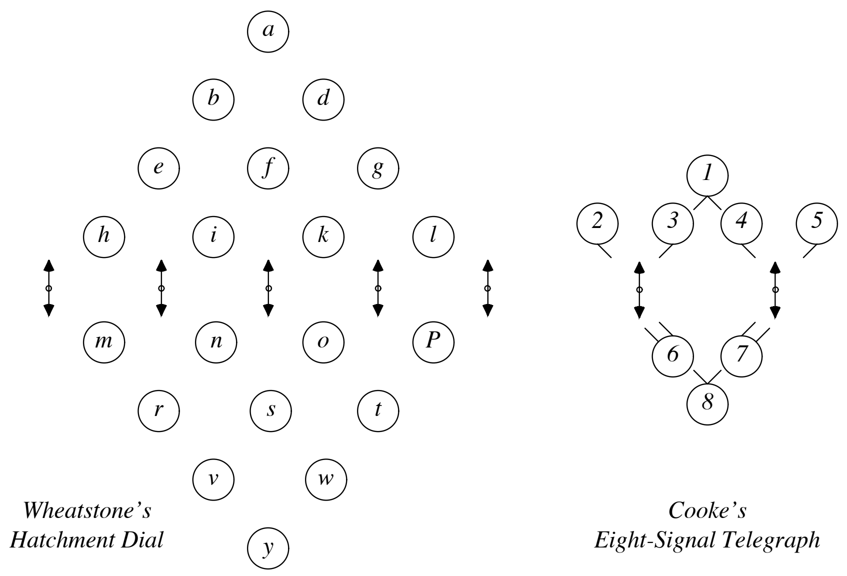

Figure 1.4 — The First Multiple-Needle Telegraphs

The first patent, dated 12 June 1837, was for a telegraph system with five magnetic needles. Any combination of two out of the five needles could be deflected either left or right, enough to signal twenty different letters. In Figure 1.4 a five-needle and a two-needle telegraph are shown. On both instruments only two needles would be deflected at a time. Together the two needles would point at the character or the code being transmitted. A little later, Cooke and Wheatstone also developed transmission codes for single-needle telegraphs that included a small number of control codes, such as *repeat* and *wait*. The *repeat* code, for instance, was sent on a single-needle telegraph as a sequence of ten clicks of the needle to the right.

William Cooke made great efforts to sell his system to the railway companies in England as a method for traffic control. In 1842, Cooke published an amusing booklet with a long title,[4] which documents his lobbying. He was perhaps a little too optimistic about the potential benefits:

> *''... trains might proceed fearlessly, whether in time or out of time, whether on the right or on the wrong line, as their speed could always be slackened soon enough to avoid a collision.''*

The system was readily adopted and used on several lines of the Great Western

4. *Telegraphic Railways or the single way recommended by safety, economy, and efficiency, under the safeguard and control of the electric telegraph — with particular reference to railway communication with Scotland, and to Irish Railways*. (Cooke [1842]).

Railways in England. The first experiments showed that the operating expenses were only one-tenth of those for optical telegraphs, and the transmission speeds were much higher.

Unfortunately, one of the first applications of the electric telegraph was to protect notoriously dangerous stretches of railroad, such as single-track lines and tunnels. Many railway accidents from this period were caused by subtle misunderstandings between the signalmen using the new equipment.

TRAIN CRASHES

The cause of a railway accident is usually investigated and documented in minute detail, so there is no shortage of material on the early protocol design problems. A single example may suffice to illustrate how major accidents could result merely from an unexpected combination of events. To be sure, the accident to be described could have been prevented if an adequate protocol had been used for the communication between the signalmen.

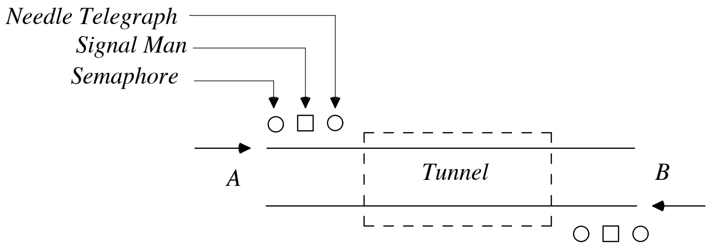

Figure 1.5 — Clayton Tunnel

The accident occurred in the Clayton tunnel, which must have been one of the best protected railway sections in England. On each end of the 1.5 mile long tunnel, 24 hours per day, signalmen were on duty. Furthermore, in 1841, the tunnel was equipped with a new space-interval block-signaling system. There were semaphore signals on each end of the tunnel, and the block-interval system guaranteed that any train passing a green signal automatically set that signal to red. It was up to the signalmen to reset the signals to green, but before doing so they were required to make certain that trains that had entered the tunnel on one side had indeed emerged again at the other end.

There were two tracks through the tunnel: one for each direction. At all times, only one train was allowed per track in the tunnel. As a further safety measure the tunnel had been equipped with a single-needle telegraph. This system was set up for the exchange of a small number of predefined messages between the signalmen on both ends of the tunnel.

Typically, after allowing a train to enter one side of the tunnel, the signalman at that side transmitted the code *train in tunnel* to his colleague. When (and if) the train emerged from the tunnel at the other end, his colleague responded with the code *tunnel is free*. Upon the receipt of that message, the first signalman could reset the entrance signal to allow the next train to enter.

To make the system foolproof, yet a third message code had been added with which a signalman could ask his colleague: *has the train left the tunnel?* The presence of the two signalmen guaranteed that the tunnel could be used safely even if, for any reason, the semaphore signal on either side of the tunnel malfunctioned. If a semaphore failed to show red after a train had passed, the signalman was warned by a bell. He could then use red and white flags to signal trains and keep the traffic going.

Still, the protocol turned out to be incompletely specified. Here is what happened in August 1861.

- A first train passes semaphore A and fails to set the signal to red. As expected, the bell warns the signalman at A (call him signalman A). He dutifully first transmits the code *train in tunnel* to his colleague, and then fetches the red flag to warn the next train.
- A second train, however, is too fast, and has already passed the green signal. Fortunately, its driver catches a glimpse of the red flag just in time as he enters the tunnel. A third train is warned in time and comes to a full stop before the tunnel entrance.
- Signalman A returns to his box and again signals *train in tunnel* to indicate that there are now two trains in the tunnel. The protocol did not account for this event so the meaning of two subsequent *train in tunnel* messages had not been specified. However, since it was unlikely that the second train could overtake the first one, no real problem existed. The only problem for signalman A was to find out from his colleague when both trains had left the tunnel, so that the third one could enter.
- To alert his colleague to the problem, signalman A transmits the only other appropriate message he has: *has the train left the tunnel?* At this point there is no hope of recovery. Even if the signalman at B could understand precisely what the problem was, he had no way of communicating this. After seeing the first train emerge from the tunnel he responds, in full agreement with his instructions, *tunnel is clear*.
- Signalman A cannot know if he should wait for two subsequent *tunnel is clear* messages or whether the message can be taken literally. He decides that both trains must have left the tunnel and allows the third train to enter by waving a white flag. The driver of the second train, though, had seen the red flag while entering the tunnel and has come to a full stop in the middle of the tunnel. After some deliberation the driver decides to play it safe and back out of the tunnel.
- In the collision that followed 21 people died and 176 were injured.

It is hard to assess who would be to blame for this accident. Once, by a freak combination of events, it had become possible for the second train to enter the tunnel before the first one had left it, there was no way to recover. The common sense of both the signalmen and the driver of the second train could not prevent the accident. The set of instructions given to the signalmen was incomplete. At the time, though, some were more eager to blame the relatively new block signaling method or the telegraph instruments than the men who had drafted the operating procedures for the signalmen's interactions.

In the early days of the railways, many accidents and near accidents were the result of an outright lack of means for communication. Later, when the right tools were available, it was discovered how surprisingly difficult it can be to establish unambiguous rules for communication. A historian of railway disasters (Nock [1967]) described the problem as follows, much in line with Polybius' earlier observations:

> *''One can almost hear the same comment being made time after time. 'I could not imagine that could ever happen.' Yet bitter experience showed that it could, and gradually the regulations and railway engineering practice were elaborated.''*

The problem was to design a practical, common sense set of rules that was efficient to use under normal circumstances and that allowed for a safe recovery from unexpected events.

1.2 THE FIRST NETWORKS

Though originally the electric telegraph was mostly used for railway signaling, it did not take long before it became more generally available. In 1851 the stock exchanges in London and Paris had been connected by telegraph, and the first public telegraph companies were founded. By 1875 almost 200,000 miles of telegraph line were in operation. At first, the telegraphs were operated with either needle instruments or Morse signaling keys. The most frequently used signaling code was a modified Morse code. The original Morse code used three signaling elements of varying duration: dots, dashes, and long dashes. The modern version was introduced in 1851, using a variable length binary code of the two familiar signaling elements: dots and dashes.

A first improvement made to this still manually operated system was the paper tape punch reader. In 1858 Wheatstone built the *Wheatstone Automatic*, with which transmission speeds of 300 words per minute could be achieved (about 30 bits/sec). It was used until very recently. After 1920 special ''tele-typewriter'' keyboards and printers were connected directly to the telegraph wires. The 5-bit code that was used on these machines was developed by the Frenchman Emil Baudot in 1874. By 1925 complete ''telex'' (telegraph-exchange) networks were in operation.

In the same period, between 1850 and 1950, two other now familiar methods of communication were developed: telephone and radio. Elisha Gray and Alexander Graham Bell, for instance, filed their applications for a patent on the invention of the telephone[5] in 1876, and in 1897 Guiglielmo Marconi built and used the first radio telegraph.

5. Bell's patent, in fact, did not mention the word ''telephone'' at all; it was titled *Improvements in Telegraphy*.

MASTER-SLAVE PROTOCOLS

The demands for the thoroughness of new communication protocols increased dramatically after 1950, when protocol execution was first automated on large mainframe computers.

One of the earliest programmable computers, the ENIAC, was built at the University of Pennsylvania in 1946. It weighed in at 30 tons. As we know, after the invention of the transistor in 1947 by J. Bardeen, W.H. Brattain, and W. Shockley of AT&T Bell Laboratories, subsequent systems quickly became both smaller and faster. Though size is not really an issue in protocol design, speed is. Even today, it continues to change the nature of the protocol design problem.

The first computers had to be connected to peripheral devices, such as paper tape readers and teletype keyboards. Since computers were initially large, expensive, and scarce, one single ''intelligent'' mainframe was often connected to large arrays of ''dumb'' peripherals.

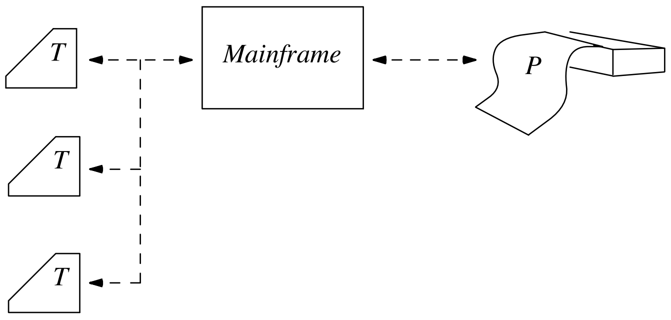

Figure 1.6 — Master-Slave Protocols

At first, the peripherals were at fairly close range, say within the same room as the mainframe computer, connected by multidrop lines. If there were no data to transfer to the peripherals, the mainframe would ''poll'' the peripherals to see if any of them had data to return or a status report to file.

Already in 1956 the first experiments took place with long-distance data transmission from computer to computer across telephone wires, causing fundamentally different types of control problems. Six years later the first data transmission via a satellite (Telstar) took place.

The first data communications protocols run on computers were rather simple encodings of the heuristics of manual operations. The procedures were used to solve a traditional master-slave coordination problem. At all times one of the two parties involved in the communication was in control and responsible for all data transfer, recovery, synchronization, and connection management tasks. Many of the older protocols were designed with this concept in mind. IBM's *Bisync* protocol, for example, dates from this period. In the 1960s, with direct connections of mainframe computers via data networks, the protocol design problem became more important. The data speeds were higher, the traffic load larger, and much of the convenience of master-slave relations was lost. Mainframes were now talking directly to each other, connected in networks of peers.

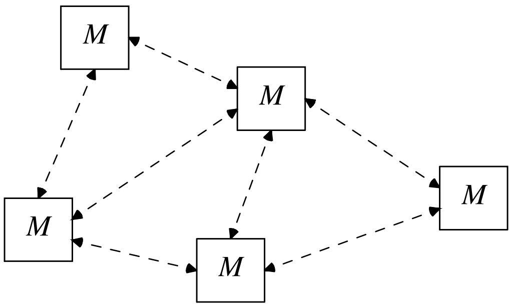

Figure 1.7 — Network of Peers

PEER PROTOCOLS

The first large-scale computer networks were the airline reservation systems from the early 1960s. The SABRE system from American Airlines, for instance, was built in 1961. In 1969 a large general-purpose packet-switching network was developed, sponsored by the U.S. Department of Defense. This ARPA (Advanced Research Projects Agency) network connected almost 1200 nodes by 1985. The Internet, a successor to the ARPA network, grew from about 25,000 nodes in 1987 to an estimated 250,000 nodes towards the end of 1989.

The number of private and public data networks is expected to continue to grow rapidly. The technology available for the construction of these systems is often sophisticated, at least as far as the hardware and the basic operating procedures are concerned. Yet, though the systems may now operate with optical fibers and satellite links, the problems that have to be solved to utilize a communication system effectively are essentially the same as in the days of Polybius.

The protocol design problem is to establish agreement about the usage of shared resources in a network of peers. It is not immediately clear which process is responsible for which task; those responsibilities may have to be negotiated. If more than one process erroneously assumes responsibility for a task, havoc can result. The network designers of the 1960s learned the hard way that very unlikely sequences of events really do happen and can ruin the best design.

Entire networks can be paralyzed by faulty or incomplete protocols. Although a collision of two data streams on a satellite channel seems harmless compared to a head-on collision of two trains, in both cases the damage can be substantial.

1.3 PROTOCOLS AS LANGUAGES

The term *protocol* for a data communications procedure was first used by R.A. Scantlebury and K.A. Bartlett at the National Physical Laboratory in England, in a memorandum that was written in April 1967. The memorandum was titled *A protocol for use in the NPL data communications network*.

We already know that a protocol is a kind of agreement about the exchange of information in a distributed system. A full protocol definition, in fact, looks much like a language definition.

- ☐ It defines a precise *format* for valid messages, such as the dots and dashes that make up the Morse code (a syntax).
- ☐ It defines the *procedure rules* for the data exchange (a grammar).
- ☐ And it defines a *vocabulary* of valid messages that can be exchanged, with their meaning (semantics).

We will come up with a slightly extended definition of a protocol in the next chapter. But note that the grammar of the protocol must be logically consistent and complete: under all possible circumstances the rules should prescribe in unambiguous terms what is allowed and what is forbidden. In practice, this is a difficult requirement to meet.

Although protocols, in one form or another, have been used on long-distance communication systems throughout history, until recently there was always a human operator who could be relied upon to make common sense decisions to resolve unexpected problems. In the 5-bit telex-code there are even two special symbols to invoke human action: the code `10010` means *who is there?*, and the code `11010` rings a bell.

In using machines rather than human operators, we have the same communication and coordination problems, but this time the errors can happen faster, and we can no longer rely on human intervention to recover from the unexpected cases.

One important hidden requirement of protocol design is now obvious: not only should there be rules for the exchange of information, there should also be an *agreement* between the sender and the receiver about those rules. IBM's *Bisync* protocol, for instance, had been implemented on many different systems, and on each new system it was embellished with the inevitable common sense of the implementer for shortcuts and improvements. These slightly differing interpretations of the rules of the *Bisync* protocol ruled out any hope that two arbitrarily chosen implementations of the same protocol could really communicate. Instead of leading to stricter guidelines for the design, specification, and implementation of protocols, this led to the institution of international standardization bodies.

1.4 PROTOCOL STANDARDIZATION

Many standardization bodies are active in the area of data communications. Examples are the National Institute of Science and Technology (NIST, formerly the National Bureau of Standards or NBS), the Federal Telecommunications Standards Committee (FTSC), and the Institute of Electrical and Electronics Engineers (IEEE). The two most important standardization bodies in this area, however, are the ISO and the CCITT.

- The International Standards Organization (ISO) includes many national standards bodies, such as the American National Standards Institute (ANSI). ANSI is responsible for important standards such as the ASCII character code and the RS232 interface definition. The ISO is organized in technical committees (TC), each organized in subcommittees (SC), and working groups (WG). TC97, for instance, is concerned with standards for computers, TC97/SC6 deals with telecommunications, and TC97/SC6/WG1 works on standards for data link protocols. The ASCII code is formally known as ISO standard 646. Unlike the CCITT, the ISO is not a treaty organization and membership is voluntary.
- The Comité Consultatif International Télégraphique et Téléphonique (CCITT) is part of the International Telecommunications Union (ITU). The CCITT is a U.N. treaty organization that was formed in 1956 by the union of two separate entities: the CCIT (telegraph systems) and the CCIF (telephone systems). Today it includes many of the public telephone companies, such as the European PTTs and America's AT&T. The U.S. Department of State is also an official member of the organization. The CCITT is organized in study groups (SG) and working parties (WP). SGVII, for instance, is concerned with data communication networks, and SGVII/WP2 works on network interfaces. The 5-bit telex code is officially known as CCITT-Alphabet No. 2. The best known protocol recommendations published by the CCITT are X.21 and X.25 (see also Chapter 2). X.21 has the dubious honor (see Bibliographic Notes to Chapter 11) of being the first reference protocol to be validated by exhaustive reachability analysis.

Another organization that does important work in this area, though it is not directly involved with protocol standardization, is the International Federation for Information Processing (IFIP). One of IFIP's aims is to serve as a bridge organization that connects the work performed in bodies such as the CCITT and the ISO. Like the ISO the IFIP is organized in Technical Committees (TC), where each Technical Committee is further subdivided into Working Groups (WG). TC6, for instance, is devoted to data communications, and WG 6.1 studies *Architecture and Protocols for Computer Networks*. IFIP was established in 1960.

Of course, protocol standardization still does not solve the protocol design problem itself. After all, what good is an international standard that is incomplete or even faulty? The standardization bodies face the same problem as all other protocol designers, and one can well say that ''design by committee'' does not always guarantee the best results.

Before this problem can be solved, we will need convincing methods to *design* and *describe* protocols, and effective methods to *check* that any protocol submitted to a standardization body is correct. Clearly, to design and describe a protocol we need to be able to express its design criteria, and to verify a protocol effectively we need to be able to check that its design criteria are met.

The problem to define a common format for the specification protocols in standardization documents has been studied for many years. Three protocol specification languages have now been developed: SDL, Lotos, and Estelle. They are commonly referred to as the three FDTs, or Formal Description Techniques.

- □ The Specification and Description Language (SDL) was developed by study groups SGXI and SGX of the CCITT. It is meant specifically for the specification and design of telecommunications systems, such as telephone switches. The study was started in 1968. A first version became CCITT Recommendation Z101-Z104 in 1976, and revised versions were published in 1982 and in 1985. A final, stable version was approved in 1987. There are two, largely equivalent, variants of SDL in use: a graphical form and a program form. The flow charting language used in the first part of this book is loosely based on the graphical form (see Appendix B).
- □ The Language of Temporal Ordering Specifications (Lotos) is being developed within the ISO, TC97/SC21/WG1. Lotos is also called a ‘‘process algebra.’’ It is based on the Calculus of Communicating Systems (CCS) developed by Robin Milner at the University of Edinburgh. The main goal of the process algebras is the formal specification of process behaviors on a high level of abstraction. The algebras define a rigorous set of transformation rules and equivalence relations that can allow a designer to reason formally about behaviors. Lotos was issued as ISO international standard IS8807 in February 1989.
- □ Estelle is a second formal description technique being developed within another subgroup of ISO TC97/SC21/WG1. A total of three subgroups of WG1 studying formal description techniques have been active since 1981. (The third subgroup studies architectural methods.) The language Estelle is based on an extended finite state machine concept (see Chapter 8). It was issued as ISO international standard IS9074 in July 1989.

Lotos is the only FDT from this range that specifically also addresses the design problem. We can learn quite a lot from the experience gained here. None of the FDTs, however, have addressed also the problem that complete designs must be verifiable at the protocol specification level. We must be able to check, preferably with automated tools, that a design meets its requirements. As it stands, verifiability cannot be guaranteed for any of the FDTs. Both Lotos and SDL specifications, for instance, can specify infinite systems, which renders many verification problems formally undecidable. There is an active area of research to develop tools for subsets of the languages, but also here the problems to be solved are formidable.

1.5 SUMMARY

Protocol design is not a new problem. It is as old as communication itself. Only when the interpretation of the protocol rules had to be automated on high-speed machines, was it discovered that protocol design in itself can be a challenging problem. The protocols being developed today are larger and more sophisticated than ever before. They try to offer more functionality and reliability, but as a result they have increased in size and in complexity. The problem that a designer now faces is fundamental: how to design large sets of rules for information exchange that are minimal, logically consistent, complete, and efficiently implemented. The problem can be approached from two sides.

- ☐ Given a problem, how can a designer solve it systematically so that design requirements are realized?
- ☐ Given a protocol, how can an analyzer demonstrate convincingly that it conforms to the correctness requirements?

In this book we study the fundamental problem of designing and analyzing protocols that formalize interactions in distributed systems. Typically, these will be interactions of computers, but they apply equally well to the interaction of people with torch telegraphs. The problem in all such systems is to come up with an unambiguous set of rules that allows one to initiate, maintain, and complete information exchanges reliably.

DESIGN DISCIPLINE

First we need to understand what the basic problems are, and we spend the first few chapters studying that. Next, we need to establish a design discipline, a set of self-imposed constraints that can help us avoid trouble. But that is not all. All freshly designed protocols, no matter how disciplined their designers have been, must be treated with suspicion.

Every protocol should be considered to be incorrect until the opposite is proven.

We will argue that to prove the correctness or incorrectness of protocols, a good set of efficient and automated design tools is indispensable.

DESIGN TOOLS

Not even the best set of rules can prevent all errors. That is a simple fact of life. We must require, however, that protocol rules always provide for a graceful recovery from the errors that do occur. It is not good enough if the protocol rules *allow* for an interpretation that prevents disaster in unexpected circumstances. We must require that the rules *preclude* interpretations that may lead to disaster.

The design methods we develop in this book are based on the concept of a *validation model*. A validation model expresses the essential characteristics of the protocol, without going into the details of its implementation. Automated tools can interpret these validation models and find the flaws in the design with relentless precision.

In the next chapters we begin exploring the general structure of communication protocols and some of the basic issues involved in protocol design.

EXERCISES

1-1. The transmission code developed by Polybius for his torch telegraph divided the 24-letter Greek alphabet into five groups. The first four groups had five letters each, and the fifth group had the remaining four.
The telegraph worked with two groups of torches: one was used to encode the group number, the other to transmit the character number within that group. Transmission took place character by character, by raising and lowering torches in the two groups. There were no codes for spaces to separate words, nor for any kind of punctuation. (Punctuation was not used yet in written Greek either.) There was, however, one additional control message to signal the start of a message: two torches raised simultaneously (see the quotation from Polybius on page 2).
What are the possible synchronization problems, in the absence of a proper agreement on the order in which the torches in the two groups are to be lowered and raised? □

1-2. Estimate the transmission speed of the torch telegraph and compare it with Chappe's system. How long does it take to transmit the message ''protocol failure?'' □

1-3. Polybius recommended the compaction of messages to reduce transmission time and thus the number of errors. Comment on this discipline. Hint: consider the opposite technique of increasing redundancy to protect against transmission and interpretation errors. □

1-4. If the signalmen at the Clayton tunnel had had the complete character set on their needle telegraphs, consider how they could have used it to resolve the problem. The length of the tunnel is 1.5 miles, the speed of the trains was approximately 45 miles per hour, and the transmission speed of a needle telegraph is about 25 symbols per minute.
The problem for the signalmen was to establish the whereabouts of the second train. At the crucial moment the second train was backing out of the tunnel to where the third train was waiting. The signalman at A assumed that the second train had already left the tunnel; the signalman at B did not know that a second train was involved. □

1-5. Try to revise the protocol for the Clayton Tunnel to avoid completely the possibility of the accident. Do not assume that the number of trains in the tunnel is always either zero or one, and do not assume that trains always travel in one direction. □

1-6. The complete code for the needle telegraph had a *repeat* message that could be used to request the retransmission of the last message sent by the other station. Consider what would happen if this discipline was strictly enforced and the *repeat* message itself was the last transmitted message of both stations. □

1-7. (Jon Bentley) If a telephone call is unexpectedly terminated, there is an informal ''telephone protocol'' which says that the caller should redial the call. If the called party is unaware of this protocol a curious problem results. A ''Lover's Paradox'' prevents contact from being made when both parties try to establish it simultaneously. What is the protocol flaw? Assume the callers are machines, how could the machines be programmed to prevent the problem from repeating itself ad infinitem? What happens to this protocol if both parties have a ''call interrupt'' feature (the ability to take an extra call when already offhook)? □

BIBLIOGRAPHIC NOTES

The French engineer Claude Chappe was born in Brûlon, France, in 1763. He originally joined a religious order as a monk, but in 1791 was forced to leave the order.

Together with his brother Ignace he set up a shop to work on the telegraph. His only publication was a short note on the optical telegraph from 1798 (Chappe [1798]). His life is described in a book by his brother, published in 1824 (Chappe [1824]). Claude Chappe committed suicide in 1805, supposedly when others claimed credit for his inventions.

The shutter telegraph used in England was designed by Lord George Murray in 1794. It is described in Reid [1886] and Michaelis [1965]. The system was in operation until 1816. The Edelcrantz system, and its signaling code, is described in Edelcrantz [1796]. Malmgren [1964] and Herbarth [1978] write that the optical system coexisted with the first electric telegraphs for a period of about five years. Herbarth [1978] includes a detailed history of the optical telegraph networks that were built in France, Sweden, England, and Germany. A photo of the telegraph station in Furusund provided the logo for the 11th conference on Protocol Specification, Testing and Verification, held in Stockholm in 1991.

The needle telegraphs of Cooke and Wheatstone were used for signaling on British railways until well into the twentieth century. A description of the early telegraphs, such as the one installed in the Clayton tunnel, can be found in Hubbard [1965], Marland [1964], Michaelis [1965], Prescott [1877], and Bennet and Davey [1965]. Only two of the five-needle telegraph instruments shown in Figure 1.4 were ever built. One of these is now in the London Science Museum; the other is in the Berlin Postal Museum.

It is still not uncommon, though less frequent, that railway signaling procedures are revised after a major accident has demonstrated that unlikely events do occur in practice. The cause of even minor railway accidents is usually studied in great detail and well documented; see for instance Nock [1967], Rolt [1976], Schneider and Mase [1968], and Shaw [1978].

Much is also known about the sometimes elaborate drum signaling methods used by African and Australian tribes and the smoke and fire signals of the American Indians. Descriptions can be found in Mallery [1881] and Hodge [1910]. Hooke's optical telegraph and the American ''basket telegraph'' are described in Still [1946].

The first use of the term ''protocol'' for data communications systems was attributed to Scantlebury and Bartlett in Campbell-Kelly [1988]. He writes:

> *''Bartlett's recollection is that the term 'procedure' had been used up to that point but was now objected to on the grounds that it had acquired a special meaning in the ALGOL report.''*

The term became a permanent part of computer jargon when it was adopted in the early 1970s by the developers of the ARPA network (Pouzin and Zimmerman [1978]).

The specification language SDL is documented in CCITT [1988]; see also Saracco, Smith, and Reed [1989], Rockstrom and Saracco [1982], and Saracco and Tilanus [1987]. The construction of an automated validator for a subset of SDL is discussed in Holzmann and Patti [1989]. Excellent introductions to Lotos can be found in Brinksma [1987, 1988], Bolognesi and Brinksma [1987], and Eijk, Vissers, and Diaz [1989]. An overview of the calculus for communication systems CCS can be found in Milner [1980].

For a different perspective of protocol standardization work and the development of the three FDTs see also Bochmann [1986] and Vissers [1990]. Estelle is described in Budkowski and Dembinski [1987].