CONTENTS

1

INTRODUCTION TO LABORATORY COMPUTER NETWORKS

Dan Terpstra, George C. Levy

INTRODUCTION

Chemistry, like many other experimental sciences, is becoming inundated by computers and computerized instrumentation. Although chemists are generally quick to take advantage of computers in their research, the concepts of distributed processing and computer networking are still foreign to most chemistry laboratories. However, laboratory mini- and micro-computers have become so prolific in modern research environments that a number of chemists and other scientists have begun to realize the necessity of enabling their computers to communicate with one another -- sharing data, programs, and physical resources.

Much of this book is devoted to documenting (through first-hand accounts) the pioneering efforts of some chemists who have anticipated the importance of computer networking in the chemical laboratory. The applications presented here vary widely in scope and level of sophistication. At one end of the spectrum we see essentially autonomous minicomputers loosely connected via slow-speed serial data links. The other end is occupied by highly interdependent mini- and micro- computers tightly coupled through high-speed parallel or serial data links. The networking schemes range from commercial implementations covering a large portion of the state of Illinois to entirely "in-house" hardware and software configurations spanning a single building or laboratory. These diverse approaches are woven together with a common thread: the recognition of the emergence of digital techniqes for laboratory measurement and control, and the necessity of providing efficient methods for handling the vast quantities of information produced by such methods.

Before dealing with specific network implementations, it is useful to construct a framework within which computer networks can be discussed. The goal of the remainder of this chapter is to provide just such a framework. A good starting point is a brief review of the historical development of laboratory computers describing mainframe systems, the introduction of laboratory minicomputers, and the current massive implementation of microprocessors and microcomputers[1]. With this information as background, the stage is set for a discussion of computer networks including their history, common network structures, and the important distinction between so called "long-haul" and "local-area" networks. Such an overview is not designed to be comprehensive, but to acquaint the reader with some of the vocabulary and concepts employed in the remaining chapters of this book.

MAINFRAME COMPUTERS

The atomic bomb and its mixed bag of technological implications was not the only offspring of weapons research during the second World War to have a profound effect on the last half of the 20th century. Weapons research also served as the progenitor of electronic digital computers, the consequences of which are only now beginning to be felt by many sectors of society.

A major computational problem during World War II was the rapid calculation of projectile trajectories for newly developed weapons. One mathematician estimated[2] that it would require 10 to 20 years for a person operating a mechanical calculator of the day to calculate a complete firing table for one such new weapon. Various mechanical and electro-mechanical devices were employed to actually perform these calculations. A relay computer called the Model III by Bell Telephone Laboratories (that could calculate such a table in roughly two months) was completed in 1944. It used 9000 relays, weighed approximately 10 tons, and could multiply two 7 digit decimal numbers in about a second.

The idea of a completely electronic calculational machine employing vacuum tube technology was introduced by Prof. J. W. Mauchly of the University of Pennsylvania's Moore School of Electrical Engineering in 1942. Funded by the government in 1943, the University developed the Electronic Numerical Integrator and Computer. ENIAC, as it was called, was too late for the war effort with its introduction in February of 1946, but it was used by the Government at both Los Alamos and the Aberdeen Proving Ground until 1955. This first fully electronic computer was immense by today's standards. It required 18,000 vacuum tubes and 150 kilowatts of electrical power to run. It occupied 1500 square feet and weighed 30 tons. The equivalent in computing power today would fit on a silicon chip the size of a fingernail and consume less than a watt of power. ENIAC could, however, perform a 10 digit decimal multiplication in less than 3 milliseconds -- about 3 orders of magnitude faster than the Bell

Model III introduced only two years earlier. ENIAC's major flaw lay in the fact that programming was done by physically plugging wires into a patch panel, a task that could take even an experienced operator many days to complete. This major drawback spawned computers in which the program, like the data, is maintained in memory -- as is done on modern computers. Two of these computers were the EDSAC (Electronic Delay Storage Automatic Computer) built at Cambridge University in 1949, and EDVAC (Electronic Discrete Variable Computer) introduced by the Moore School in 1950.

The solid state era was heralded in late 1947 with the invention of the point-contact transistor by William Shockley, John Bardeen, and Walter H. Brattain of Bell Telephone Laboratories. Although they received the Nobel Prize in 1956 for their efforts, the initial reception of their device was quite cool. In fact, transistors weren't incorporated into computers until the late 1950s when they were employed both by IBM and Univac. By this time, magnetic core memories had already replaced other forms of computer memories, improving speed by another factor of 100 over earlier models. Both magnetic tape and the newly introduced magnetic disks were now available as mass storage devices. User interaction on these early computers was primarily through the use of punched cards and line printers, a legacy that has remained with us for far too long.

Software development was originally much more slowly paced than hardware development, since the concept of a "program" itself was still crystallizing. Univac was an early leader in making computers easier to program. In the early 1950s, virtually all programming was done in machine language -- binary strings of 1s and 0s, perfect for the computer but almost indecipherable for the human. In an effort to alleviate this problem, Univac introduced the first interpreter in 1952. It read assembly code mnemonics line by line and interpreted them into machine language commands that were then immediately executed. Univac also introduced the first compiler language at about this time in which a program written in algebraic notation was compiled into a machine language program at one time by the computer. This machine language version could then be directly executed by the computer at any later time. IBM made a contribution to compiled computer languages in 1957 with the introduction of its Formula Translation programming language, otherwise known as FORTRAN. In 1959, two more popular computer languages were introduced: ALGOL (Algorithmic Language) -- developed jointly by the Association for Computing Machinery and the German Association of Applied Mathematics, and COBOL (Common Business Oriented Language) -- an offering from the U.S. Pentagon.

As mainframe computers entered the 1960s, programming became more and more sophisticated. The operating system, or "program to run programs" was introduced early in the decade. This eliminated the need for the user to be concerned about most of the "housekeeping" chores associated with running a program. Also in the 60s, timesharing developed as a way to maximize hardware resources. Before this time, most computers executed programs sequentially in what is known as "batch" execution. Dartmouth College and General Electric pioneered the implementation of a system in which computer resources were rapidly shared between many users and many programs, giving each one a small "share" of the computer's time (hence the name) before moving on to the next user and program. One of the results of timesharing at Dartmouth was to make the computer more accessible to large numbers of students. This led to the need for an easy to learn interactive computer language for instructional purposes. Such a language was BASIC (Beginner's All-Purpose Symbolic Instruction Code), developed at Dartmouth in 1964.

Hardware improvements were also far-reaching in the 60s. The integrated circuit was first demonstrated by Texas Instruments in 1958, and was becoming incorporated into a third generation of computers by 1964, supplanting the first-generation vacuum tube and second-generation discrete transistor based machines. Main memory was still prohibitively expensive, so "virtual memory" techniques were developed to move into mass memory the programs or data that were not currently in use and recall them as they were needed. Multiprogramming techniques and parallel processing, in which many streams of data were simultaneously operated upon, were both pioneered during this period. One parallel processor, the Illiac IV, was begun in 1967 by Burroughs Corporation. It contained 64 parallel processing elements, a megabyte of main memory, and could execute instructions at the rate of 200 million per second. Life was made easier for the user of large systems with the gradual replacement of teletypes and card readers by cathode ray terminals (CRTs). And the interchange of alphanumeric information was simplified by the introduction in 1963 of the American Standard Code for Information Interchange (ASCII) which is in essentially universal use in computer peripherals today.

By the 1970s the mainframe computer arena had stabilized in most instances. One writer states, "Instead of the revolutionary changes in fundamental organization or architecture that occurred in the 1950s and 1960s, developments in the 1970s were more evolutionary."3 This was primarily due to the fact that too much money and time had been invested in hardware specific programs to make affordable any dramatic changes in the architecture on which those programs had to run. Integrated circuits were gradually increasing in complexity, however, with the result that earlier architectures could be packed into much smaller boxes with much smaller price tags and power requirements, and concomitantly with higher speeds. In addition, semiconductor memory technologies had matured to the point where main memory limitations were no longer as constricting as they once had been. Large Scale Integrated (LSI) circuits on single silicon chips made it possible to disperse some of the intelligence of the computer into its peripheral devices, thus easing the overhead of monitoring many functions at once. This carried the concept of timesharing one step further into the realm of distributed processing and data communications networking, an area that will see extensive development in the current decade.

There were some dramatic mainframe architectural advances in the 70s. One of the most spectacular was (and still is) the Cray-1 supercomputer, a vector processor introduced in 1976 by Seymour Cray, a founder of and former designer for Control Data Corporation (CDC). The computer itself stands roughly seven feet high in a cylindrical configuration to minimize interconnection distances, and is liquid-cooled to maintain proper operating temperatures. It can execute between 80 and 130 million floating point instructions per second, giving it current claim to the position of world's fastest computer. This claim may be challenged in the near future when Lawrence Livermore Laboratory introduces its new S-1 Mark IIA supercomputer, said to perform up to 400 million floating point operations (400 megaflops) per second. In addition, CDC has already announced its Cyber 205, claiming a maximum speed of up to 800 megaflops per second. Not one to rest on its laurels, Cray Research is reported to be working on the Cray-2, sure to keep it in competition for the world speed title[4].

The future for mainframe systems is difficult to predict, particularly since increasing levels of integration are making it virtually impossible to delineate the divisions between mainframe, mini- and micro-computers. Continuing trends in computer networking and distributed intelligence will inevitably obscure the boundaries of the "mainframe" computer as a monolithic device. It will become increasingly difficult to distinguish at what point one computer "ends" and the next "begins". This trend toward decentralization of computational hardware may find itself reversed to some extent by the impending implementation of Josephson junction devices. These devices offer nanosecond switching times and extremely low power consumption[5]. An entire computer using such devices could be built in eight cubic inches and operate at a factor of 5 faster than even the new supercomputers. One major drawback of Josephson junction technology is that this eight cubic inch computer is superconducting, and thus must operate at liquid helium temperatures, requiring complex support facilities and reinforcing the centralization of computational power. As unpredictable as the future of mainframe computers may be, they will certainly continue to play an active role in the computerization of society.

MINICOMPUTERS

Even as mainframe computers were becoming larger, faster and more sophisticated in their move from the transistors of the second generation to the integrated circuits of the third generation, another computer revolution was beginning that would soon profoundly affect scientific research laboratories. The first rumblings of this revolution were heard as early as 1959 when the newly formed Digital Equipment Corporation (DEC) began marketing its Programmed Data Processor, the PDP-1. This 18-bit computer was inexpensive for computers of that day, selling for an average price of $120,000. It took six years and seven models before DEC introduced what was to become the most successful minicomputer in history: the PDP-8, introduced in 1965. This computer had a capacity of up to 4096 12-bit words of core memory and sold for less than $20,000. It was the first mass produced computer, and the first to be given the name "minicomputer". The PDP-8 was not alone for long. By 1966, Hewlett-Packard had marketed its first "instrumentation computer" and in 1967 introduced a 16-bit general purpose mini-computer as well. Data General was founded in 1968 by three former DEC employees and introduced its 16-bit NOVA line that same year at a base price of only $8000. By this time the minicomputer was firmly established in industrial and laboratory environments nationwide.

The PDP-8 and its younger siblings and rivals ushered in an entirely different approach to the marketing and application of computers. Since the devices were mass-produced, mass-marketed, and highly competitive, the profit margin was much less than on mainframe systems. Minicomputer companies could not afford and did not provide the extensive hand-holding and customer support that had become standard with mainframes. Small profit margins also inhibited the development and sale of higher level language facilities since the user could rarely justify the high relative cost of such additional software capabilities.

The cost of software was not the only reason for a primary reliance on assembly programming. Minicomputers often were simply not big enough to support high level compilers or interpreters. Expensive main memory and severe memory address limitations demanded space efficiency that could often only be achieved through assembly level programming.

The new minicomputers were designed for the real world. They generally had short word lengths and high speeds, ideal for many real-time laboratory control and data acquisition requirements. Much attention was devoted to their input and output structures, making them easier to interface to laboratory equipment. They were generally designed to be single-user, single-task machines, often equipped with analog-to-digital and digital-to-analog converters. The applications to which these computers were put often demanded high program execution speeds, another strong argument at the time in favor of direct assembly language programming. Since the computers themselves were so inexpensive, they were often used in situations where budgets would not allow for sophisticated user input and output. The most commonly used I/O devices were the paper tape reader/punch and the teletype.

Minicomputers matured during the early 70s and as they did, they found new applications that were previously non-existent. The word lengths of these laboratory computers varied from the original 12-bit PDP-8 to the 20-bit Nicolet 1080, but most minicomputers had settled on 16-bit words as the optimum cost-effective architecture. In chemistry laboratories, minicomputers became attached to x-ray crystallographic equipment, to esr and mass spectrometers, and to gas and liquid chromatographic equipment. In some cases, the availability of the computer changed the entire texture of the experiment. For example, the implementation of the Fast Fourier Transform algorithm on minicomputers created fundamentally different methods of measurement for nmr and infrared spectroscopy.

The power of LSI circuitry resulted in greater sophistication for minicomputers in the late 70s. Large minicomputers began to look very similar to their mainframe counterparts. As minicomputers penetrated the small business and word processing environments, they developed memory remapping techniques to circumvent limited memory address space, they began to understand high level languages, and they supported increasingly sophisticated input and output devices. In addition, it was no longer necessary to restrict minicomputers to single users or single processes. Minicomputer operating systems became more intelligent and incorporated more of the features and facilities of mainframe operating systems. Minicomputer hardware incorporated priority interrupts, direct memory access, and other features that allowed the monitoring and control of many simultaneous real-time events.

By the end of the decade, some minicomputers had grown to 32-bit word lengths and the distinction between low-end mainframe and high-end mini became more a matter of semantics than of substance. At the same time, 16-bit microcomputers were encroaching on the lower end of the minicomputer spectrum, making the minicomputer's hold on its niche look precarious indeed. In fact, by 1980, a number of microcomputers outstripped the capabilities of their bigger brothers. The role of instrumental control that once belonged solely to the minicomputer is now often being given to a handful of microcomputers with the result that the minicomputer may soon become the "laboratory manager" for a number of intelligent instruments tied together by a local-area network. This course of events is already occurring, and is in fact one of the driving forces behind the accounts in this book.

MICROCOMPUTERS

The microcomputer, or microprocessor as the central processing chip itself is often called, had its beginnings in the electronic desktop calculators of the late 1960s. LSI circuits were becoming increasingly complex and calculator manufacturers were rapidly incorporating them into their products. Late in 1969, Intel Corporation was asked by a Japanese calculator firm to design an LSI chip for their calculators. Intel designed a chip that was flexible enough to be considered a general purpose computer as well as a calculator chip, and the microprocessor era was born. This chip, called the 4004, was announced in the middle of 1971. It and its three support chips comprised all of the elements of a simple computer. The 4004 cpu chip could handle roughly eight thousand bytes (8 bits) of memory and had a four bit word length. Also in 1969, both Intel and Texas Instruments were approached to design a slightly more complicated chip for an intelligent computer terminal. Intel succeeded, but its design was more powerful and much slower than what was required. Texas Instruments received the contract and Intel was left with the design for an eight bit version of its 4004. Early in 1972 Intel began marketing this device as the world's first eight bit microprocessor, the 8008. The 8008 initially sold for $200 but required ten times its worth in peripheral circuitry to turn it from microprocessor into microcomputer. Many other companies began marketing microprocessors and microcomputers, and by 1975 there were more than forty from which to choose.

Microprocessor applications soon stratified into two distinct areas. The first included control functions and rudimentary programming that were best served by the simplest of devices such as the 4004 or the popular TMS 1000 from Texas Instruments. These applications were as diverse as washing machines, microwave ovens, automobile carburation, and the like. Advances in this area tended to put increasing amounts of sophistication into a single silicon chip until by the late 70s chips were available that could, for example, convert an analog input into digital form, manipulate it, and convert it back into analog as an output. Single chips also became available that contained all the elements of a computer, including the processing unit, program memory, data memory, and input/output structures.

The second area of microcomputer applications emphasized the computer-like features of these amazing devices, and before long threw them into direct competition with minicomputers. As the speed and power of microprocessors improved, they began to be employed as actual computers.

In 1974, Intel introduced what was to become the most sought after 8-bit microprocessor ever: the 8080. It rapidly found its way into a wide variety of applications, and in the process spawned a competing company (Zilog) with a superior product: the Z-80, a faster 8-bit microprocessor designed to run programs written for the 8080, as well as programs using the expanded Z-80 instruction set. Intel combatted the Z-80 with an upgraded version of the 8080 that was called the 8085 and eventually ran even faster than the 4 MHz Z-80A. The Intel family of LSI chips was marketed by Intel both at the chip level, and at the board or system level. Intel's Intellec systems utilized the Multibus interconnection system, which has recently become standardized by the Institute of Electrical and Electronics Engineers as IEEE-796.

Another important development that followed on the heels of the 8080 was the S-100 bus. Originally marketed by Altair in 1976, the S-100 was intended as a vehicle for introducing the 8080 to computer hobbyists. A number of small companies began selling low priced S-100 bus compatible hardware, and the S-100 rapidly became a de facto standard in small business and research applications as well as in computer hobbyist circles. The IEEE recognized the popularity of the S-100 bus, and in 1979 it proposed an industry-wide standard (IEEE-696) for S-100 compatible products. In just a few short years, microprocessor chips had become not just smart controllers to be hidden inside a product, but computers in their own right, able to compete head-to-head with minicomputers.

High performance 16-bit microcomputers arrived on the scene late in 1978 with the 8086 from Intel, followed by Zilog's Z8000, Motorola's 68000, and National's 16000. Support circuitry on single silicon chips began to proliferate for microcomputers. It is now possible, for example, to obtain single package devices that can control floppy disk drives, handle sophisticated data encryption schemes, do floating point arithmetic, or even interface to local-area computer networks. In fact, since each of these devices contains a microprocessor of its own, it is not at all uncommon to find a microcomputer system that contains half a dozen or more microprocessors operating in parallel under the control of a master processor to coordinate its computational tasks.

From the outset, the software tools available for microcomputers were much more extensive than those for minicomputers at an earlier, parallel stage of their development. There are a number of reasons for this, not the least of which is the dramatic and continuing reduction in cost of semiconductor memory. Most 8-bit microcomputers are designed to support 64 kilobytes (one byte equals eight bits) of memory, and many microsystems are already finding this limit restrictive. This would have been unthinkable a decade earlier when minicomputers were being designed to support only four to eight kilowords of memory. The newer 16-bit microcomputers can handle megawords of memory, still too expensive for most implementations, but indicative of further projected reductions in memory costs.

Another reason for the plethora of excellent microcomputer software tools is the development of cheap and reliable mass storage in the form of floppy disks. A floppy disk is an eight inch or five and one quarter inch flexible plastic disk coated with iron oxide and kept in a padded folder for protection. It looks much like a 45 rpm record, but it can hold up to half a megabyte or more of programs and data for less than five dollars. The first comprehensive floppy disk operating system designed specifically for microcomputers was, like so many other microcomputer firsts, developed in conjunction with the Intel 8080. This operating system, CP/M, was introduced in 1975 by a company that has since become Digital Research[6]. CP/M provides facilities formerly available only on mini- and mainframe computers. Much in the same way that the S-100 bus became a hardware standard for the 8080 family, CP/M is becoming a software standard. Digital Research has since marketed the first multi-user microcomputer operating system (MP/M) and the first network microcomputer operating system (CP/Net); both are essentially supersets of CP/M and both run on the 8080 family of microcomputers.

Large main memory sizes, readily available mass memory, and the huge numbers of microcomputers sold, all coupled with the experience gained in the development of minicomputers, made it profitable to develop high level software for microcomputers. The first resident high level language for micros was PL/M, a modification of IBM's PL/1 language introduced by Intel for the 8080 in 1974. Today a subset of PL/1 itself is available under the CP/M operating system, along with a wide variety of BASIC interpreters, Fortran, Cobol, Pascal, or any one of a number of other languages. In addition to this broad spectrum of languages, excellent hardware and software debugging tools are available in the form of emulators and supervisory programs to speed software testing.

As microprocessors get more powerful and more specialized, it seems inevitable that the tasks to be performed will be fragmented into a number of common "building block" sub-tasks, each handled by its own dedicated microprocessor. Building a computer will then be little more than deciding how many and what types of these "building blocks" are needed to fulfill a given set of requirements, and connecting the blocks in the manner prescribed by the makers of the circuits. This leads directly to the idea of distributed or multiprocessing where a number of processors work together, each fulfilling a small part of the total function of the computer. Such schemes are becoming increasingly visible as we enter the decade of the 80s and, as should be obvious from the nature of this book, are already underway in some contexts. They will be discussed in futher detail in the remainder of this chapter.

COMPUTER NETWORKS

In its simplest and most general definition, a computer network is the means by which discrete computers exchange electronic information. However, as is often the case, this simple definition harbors a wealth of variation and complexity. For example: the number of computers involved in a given network may be as few as two or as many as several hundred or even several thousand; the computers in a network may all be identical, sharing machine level programs and memory resources, or they may be totally dissimilar, sharing only data and mutually defined messages; a computer network may cover as little as a few meters via simple twisted-pair or coaxial cable, or it may span the globe using communications satellites and microwave transmission techniques.

Despite the number of apparent differences, all computer networks share certain common features. These features have been identified and broken into seven somewhat arbitrary "layers" (illustrated in Figure 1.1) by national and international committees concerned with such matters[7,8]. All seven of these layers need not be present in every network implementation, but taken together they provide a comprehensive architecture for computer networking. The sole purpose of any given layer in this network heirarchy is to communicate with the same layer in another network computer. This is referred to as peer-to-peer communication. In order to fulfill this function, a layer must be able to pass information to and from the layers immediately above and below it (called layer-to-layer communication). Thus, for example, the network layer in one computer would communicate with the network layer in another computer by passing information down to the link layer and finally to the physical layer where peer-to-peer communication transfers the information to the other computer. It would then be passed back up through the physical and link layers of the second computer and received as peer-to-peer communication by the network layer of the second computer. To complete this transfer, the link layer, for example, is only required to pass information between itself and the network layer above it, or between itself and the physical layer below it. With this scheme, the structure of any individual layer can be modified independently of other layers, as long as the interface between layers is kept constant.

In order to clarify the roles played by each of these seven network layers, an analogy may prove helpful:

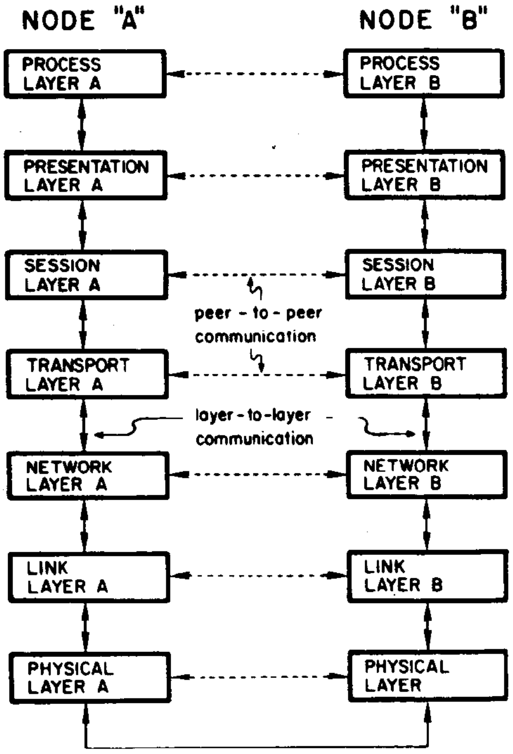

FIGURE 1.1. The seven layers of network protocol between network node A and network node B.[8]

<u>Process Layer</u>: Distinguished Professor A has conceptualized a brilliant new hypothesis that she wishes to share with her colleague, Doctor B (a peer-to-peer communication). She realizes that she cannot share her mental processes directly with Dr. B, but must convert them to a presentable verbal or mathematical form. In much the same way, specific user input or program results (ideas) form the process layer for a given computer, but are not directly transferrable to another computer.

<u>Presentation Layer</u>: Prof. A composes her hypothesis into a collection of sentences and formulae that she feels she can present to Dr. B (process layer-to-presentation layer interface). The computer must also convert specific data into a format understandable by another computer.

<u>Session Layer</u>: Prof. A decides that she will discuss her hypothesis in a verbal conversation with Dr. B. Implicit in this decision is a set of social amenities that will serve as cues for the beginning, middle, and end of the conversation session: "Hello Bill, this is..."; "I called about..."; "Give my best to