The Minicomputer in On-Line Systems

Small Computers in Terminal-Based Systems and Distributed Processing Networks

Martin Healey
David Hebditch

Winthrop Publishers, Inc., Cambridge, Massachusetts

Library of Congress Cataloging in Publication Data

Healey, Martin.
The minicomputer in on-line systems.

Includes index.
1. Minicomputers. 2. On-line data processing.
3. Electronic data processing—Distributed processing.
I. Hebditch, David Leroy. II. Title.
QA76.5.H444 001.64 81-589
IBSN 0-87626-579-4 AACR2

10 9 8 7 6 5 4 3 2 1

Foreword

Computing, like any high technology business, is burdened with unanswered questions, some serious, others comparatively trivial. A splendid example of the latter category has been the endless debate over who invented the first stored program computer. A more critical inquiry is posed in discussions about just what a minicomputer is, or for that matter what a terminal, a small business system, or a terminal computer is.

Debate questions are always fun. Engaging in deep dialog is a lovely way of spending time. It is at least as good as attending conferences, writing papers, and otherwise avoiding the need for work. Discussing matters whose resolution is largely a semantic issue clearly permits the practitioner the luxury of avoiding technical decisions due last week or last month. Since it is not considered cricket to terminate the conversation of people who debate these lofty and near-impenetrable matters, the clever gamesman can survive by avoiding decisions, with their consequent risk of error.

It has long been known that certain MIS professionals have very bad attitudes on these matters. They attempt to solve problems by creating workable solutions. This evil practice should be stopped, because it might force the debating societies to stop debating and go to work—the last thing the talkers desire. If one is forced to bet one's reputation on a path or a fork in the road, the chances for error are high. Better to debate than to do!

Books about minicomputers invariably seem to fall into the trap of spending several hundred pages attempting to define minicomputers. The matter used to be very simple for pragmatic types who didn't worry about elegance. The parameters were fairly simple: (1) physical size, and (2) manufacturer's name. Consequently, a computer that could be picked up by an average-size individual, one manufactured by anybody other than a traditional mainframer, was by definition a minicomputer. Unfortunately, some recent developments have destroyed this easy definition. There aren't many people who can get their arms safely wrapped around an H-P 3000 or a DEC PDP-11/44. Anyone who *can* is a candidate for several Olympic awards. The other side of the equation has also come apart: IBM entered the minicomputer business directly, Univac joined by acquisition, and Honeywell reenergized its long dormant minicomputer operation. End of one pass at decision-making criteria!

So what is a minicomputer? Can a machine with a 32-bit-wide data bus, a 300 ns cache memory, and a million bytes of main store be called a mini-anything? Of course it can't. The truth is that yesterday's definitions are now nearly meaningless. There is a tidy, smooth curve of machine power beginning with microprocessors and running exponentially up to a Cray-1. Somewhere along this curve the traditional independent observer can decide that these are micros, these are minis, these are maxis, and everything else is a mainframe. You pays your money and takes your choice; who cares what it is called?

Having thus neatly solved the question of defining the minicomputer, it is necessary to find the separation between one of them, whatever they are, and a terminal. This should be a good deal easier because the parameters are better known.

A terminal is a box. That box is remote from a computer. So far, so good. Does a terminal have: (1) intelligence, to process information, (2) private store, (3) mass storage even in limited quantities, (4) human intervention devices such as keyboards, and/or (5) connections to the computer? These questions define the various classes of terminals. If there are enough yes answers, perhaps the box in question is not a terminal at all but a computer. Can the answer be determined numerically? Does a single yes mean terminal and three or more mean computer?

As with the case of the minicomputer definition, I am now involved in a semantic snarl-up at least partially of my own making. Indeed, does it really matter? Isn't it more than enough to say that a terminal is a remote electronic device that has the power to somehow communicate something to somewhere in one of a variety of forms? Enough!

None of this semantic footwork really matters very much to the dirty-handed workers in the trade. People take a machine, grab some devices that work remotely and go to work building systems that provide intelligence to the end user community residing at the periphery. That is what it is all about. How the goal is reached and what it is called should be relegated to the Oxford Union.

Having placed definitions in the category of a problem that has been solved (a longtime ploy of politicians who believe that if the problem is said to be solved it is), we must turn to the real snags in computing during the next few years.

The data communications explosion begun so promisingly a few years ago continues to run into roadblocks. Most of the obstacles are erected by longtime communications carriers whose general approach is to try to bar all the new freedoms the FCC carefully nursed into life. The common carriers are not stupid. They see major revenue sources flying away as their aging, analog-based plant rapidly falls into disuse before its natural accounting life is ended. There is little quarrel with the notion that a forty year write-off on technological equipment is an invitation to disaster. However, the fiscal policies and practices of the carriers were established and firmly set long before computers were invented. The repeated, public predictions by Howard Anderson of the Yankee Group that the common carriers will be an easy target when the computer people really start after them seem to be coming true. But this is another of those debating-society issues.

The carriers are digging in for a long fight. It isn't merely a matter of matching their services to those the independents are making available. To quote myself from a near-forgotten *Datamation* article, "If MCI and Datran didn't exist, we would have had to invent them to keep AT&T honest." What has happened is that the carriers have reacted savagely enough to make the future bleak for all but the best financed independents. Datran is gone, even though it cost AT&T $50 million in settlements. By its own admission MCI is only marginally profitable. The regional carriers are gone. Telenet was acquired by GT&E.

The traditional carriers are defending on many levels. First, they take every legal action possible to delay the implementation of new services by independents. Second, they are providing their own competitive services. Third, they are attempting to remove from service functional capabilities already in general use—for example, distance pricing and private line networks. Finally, they are always just a little slow in connecting local loops and repairing outages.

All of this means that the natural, normal, and highly proper convergence of data processing and data communications is a good deal more difficult than it should be. The problems are political and economic, not technical. This book describes systems in being and how they are built. Neither it, nor any other book on the market, can forecast the political climate that may well undercut a sound technical performance.

The clash between a slow-moving, regulated industry and a very rapidly changing, thoroughly unregulated business is of increasing concern to the user community. Knowing how to build a system is of little value if a very large traffic cop keeps saying, "You can't do that" at regular intervals. The cop wears many disguises. He may appear as a British labor leader who has instructed his men not to install non-GPO modems, or as a Nigerian minister who refuses to allow any encoded data on public transmission facilities, or as a Canadian telecommunications minister who believes that all necessary transmission can be accomplished on public networks.

Who suffers from these actions? Not the mythical general public whose interests are allegedly being protected. The loser is the end user at whose behest remote, terminal-based systems are being constructed. The solution is political. Corporations don't have the influence when it comes to a clash between the need for more effective business communications and an impact on the plain old telephone service.

A sense of outrage continues to exist in the computer community over the restrictions, real and apparent, arbitrary and legislated, that grow ever more widespread. The game is at a high level because the revenues involved are very large, billions of dollars. Even the most astute technical people are likely to be blind to these affairs; their interest is in getting the job done.

Before 1976, the direction and pace of the data processing and data communications convergence were fairly clear. The technical job to be accomplished was, and still is, becoming ever clearer. Books like this describe what has to be done. But the whole effort will fall apart if the industry's technical experts do not take the time and energy to come to grips with political and legislative forces. Things are happening that may undermine even the best planned efforts. It was previously considered reasonable to assume that telecommunications regulations would be eased. In actuality, matters have gone quite the other way. The French PTT still wants to pull cable in an era when satellite transmission is completely routine. Fiber-optical transmission is regarded with suspicion by those who grew up with twisted pairs of copper wires. Digitally based communications equipment cannot approach peak effectiveness when inhibited by operating parameters in a thirty-year-old analog carrier. Voice, data, and image transmissions are still seen as separate entities when all three have become little more than the movement of digital bit streams.

The technology of communications-based systems is expanding at a rapid pace. Each new development, however, is delayed by the defenders of the past. The excuses vary but the most commonly heard are: "unproven technology," "too great an impact on existing services," and "no tariff has yet been filed."

The common carrier attack on remote systems has been sustained, but at worst their efforts can only slow down implementations. But there is another emerging threat that has the possibility of totally destroying effective remote computing—the concept that data should be held within national boundaries as a national resource.

Remote computing systems by their very nature weaken the idea that all data should be maintained locally. In fact, while much of day-to-day operational data should properly and for good technical reasons remain local, there is a valid set of aggregate data that make little sense locally. How is a modern corporation to optimize its use of scarce resources if arbitrary, often capricious, regulations are placed on its ability to move the data to the location where it is needed?

The issue of transborder data flow is rapidly growing in its impact upon remote computing systems. It is beyond the scope of this book to discuss whether it is a real issue or one dreamed up by a tiny minority of social activists as another means for controlling large corporations. However, it is an issue with a major potential for mischief.

Of still more recent generation is the issue of national vulnerability to computer/communication outages. The thesis as evolved by a Swedish governmental committee revolves around the notion that each national society must have all of its own facilities for keeping the computer/communications complex going. Why? Because the big, bad wolf, presumably an American-made wolf, will come along and reduce a nation to third-rate status by failing to provide spare parts or the technical know-how to keep the complex functioning. It is hard to take such scare tactics seriously. Yet it would be foolish to assume that the threat will simply disappear.

The point in both of these cases is the same. It is no longer sufficient for a technician to stay purely technical. The construction of computer-based systems is full of social, cultural, economic, and political questions that can no longer be cheerfully disregarded. Enough good systems have been built for the alarm-raisers to have had their interest awakened. A cynic might suggest that we ought to have failed more often because if that were the case, the politicians would leave us alone. So perhaps some of the energy and effort devoted to trying to decide what a minicomputer is, or what a terminal is, should be applied to the process of maintaining freedom to implement. The definitions are only of importance to lexicographers, but the politics are critical. The deeper people delve into remote computing systems, the more the impact will be if the system proves "culturally unacceptable."

We used to think that technology was hard. It isn't. It's still not easy or off-the-shelf. The multiple forces that pull and tug within a system are not that easy to see, and it is still hard to find all the trade-offs. But Healey and Hebditch have pulled out most of the issues into plain sight and tried to dispel the mystery. It's not a black art, nor does it require implementers capable of leaping tall buildings.

Minicomputer systems can be thoroughly constructive for the corporation in which they are based. They can also be major traps for the unwary. There are rules of the game that must be obeyed even as with large-scale mainframes. Healey and Hebditch detail the structure for minicomputers in the distributed environment. Their game plan is sound. Now it is up to all you implementers to follow their precepts.

PHILIP H. DORN

Preface

This book is the culmination of a discussion that has been going on for some four years between the authors. During the mid-1970s we have seen the steady convergence of computer and telecommunications technologies. In particular, the data processing industry has continued to move away from batch processing to transaction processing, and networks of terminals have improved user access to computer systems.

Conspicuous in these trends has been the emergence of the minicomputer. These compact and powerful processors, which originated in control systems, have found a new role in the commercial field as data concentrators, message switchers, front-end processors, terminal controllers, and special-purpose transaction processors. A common characteristic in these systems is the use of interactive and telecommunications facilities for data transport.

Although the cost/performance ratio and general modularity of minis make them very difficult to ignore for many applications, the problems of system design, development, and implementation can be numerous.

The authors are consultants, one majoring in minicomputers, the other in communications systems, both having a reasonable knowledge of the other's specialization. During many discussions (in just as many watering holes), we managed to convince ourselves that if we had an *average* understanding of these problems, then at least half the data processing profession was really struggling!

Minis are ideally structured for handling data communications and terminals, but that does not mean that they always do it well. This book is intended as a review of minis, communications, and how the two work together. Many products are described by the way of illustration; where a specific device or software package is mentioned, we do not intend to imply that it is any better or worse than comparable components, merely that it is representative of a certain approach.

If this book is for anyone in particular it is for the COBOL programmer whose boss is planning to buy a six-pack of minicomputers on which to base a network of interactive terminals. But we hope that technical managers, designers, systems programmers, and application programmers in user companies, manufacturers, and software houses will find it helpful in the development of effective systems as well as in the development of their personal expertise.

MARTIN HEALEY
DAVID HEBDITCH

Contents

1

The Role of the Minicomputer in Data Communications and Distributed Processing Systems

This chapter reviews the various roles minicomputers play in terminal-based data processing and communications systems to provide a background against which we can consider the technical problems involved in implementing minis in such roles. For the time being we shall define the mini as a small-scale, general-purpose computer such as the Digital Equipment Corporation's PDP-11 or Data General's NOVA series. Small business computers (such as the IBM System/3 and ICL 2903) and special-purpose distributed processing products (like the Burroughs B776 and IBM 8100) are excluded. A more complete definition of the minicomputer will be discussed in chapter 2.

Some Typical Systems

When minicomputers were first developed, they tended to be used in industrial process-control and data-acquisition systems. In the former case, the nature of the application demanded a processor capable of acting in "real time," in the original sense of the expression, a processor that could economically and efficiently interface to a number of local and/or remote telemetry and control devices. Most minicomputers were designed, therefore, to operate in a highly responsive on-line environment. Their development engineers had probably never heard of batch processing (or data processing, for that matter). Universities and research bodies also quickly saw the potential of the mini, which typically is used for one-off jobs generally developed in batch or (more often) interactively, using high-level languages such as FORTRAN or BASIC. One of the earliest uses of minis outside the industrial and research fields was in message switching. Clear-text messages received over telegraph circuits are transmitted to destination teletypewriters/teleprinters according to routing information held in standard or nonstandard headers. Disc or drum storage may be used to implement a delayed-delivery facility.

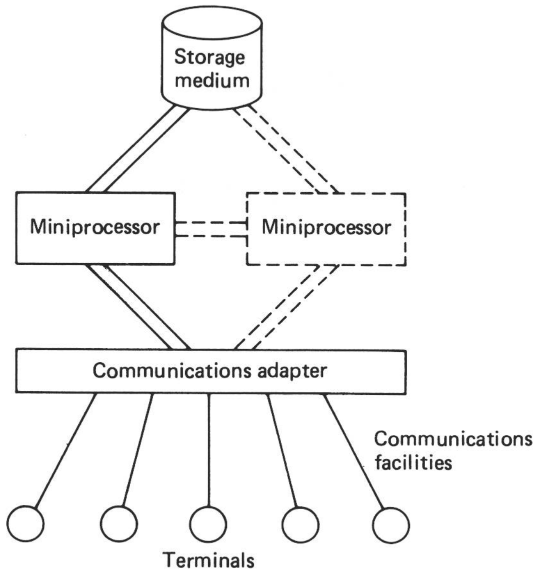

Figure 1.1
The minicomputer as the central site processor (in single or multiple configurations)

All the above systems are typical of the mini used in a stand-alone situation (as shown in figure 1.1). Until recently, the application of the mini as the main computer in commercial data-processing systems was somewhat limited mainly as a result of the lack of suitable software. Today the problem is no longer a lack of software but the choice of a package/language from the plethora of programs now available for business applications. Most such packages assume application as terminal-based transaction processing systems (the structure and relevance of this approach will be considered in chapters 7 and 8). The two earliest and most widespread uses of minis in commercial teleprocessing were as front-end processors (figure 1.2) and as remote batch terminals (figure 1.3).

The first generation of communications control units (CCUs) typified by the IBM 2701, 2702, and 2703 were hardwired devices, inflexible and expensive. In order to achieve a more adaptable, cost-effective solution that would relieve the mainframe of some of its steadily increasing control-program load, many software houses (and some users) developed their own software to work in one of three ways: to emulate the CCU (for example, the IBM 2703); to emulate another peripheral controller (for example, the IBM 3803 magnetic tape controller); or simply to support the mainframe input/output (I/O) channel.

Because of its market size IBM was inevitably subject to attack by this approach: its response was that because central processors and storage were becoming so cheap, there was no point in shifting control program functions into

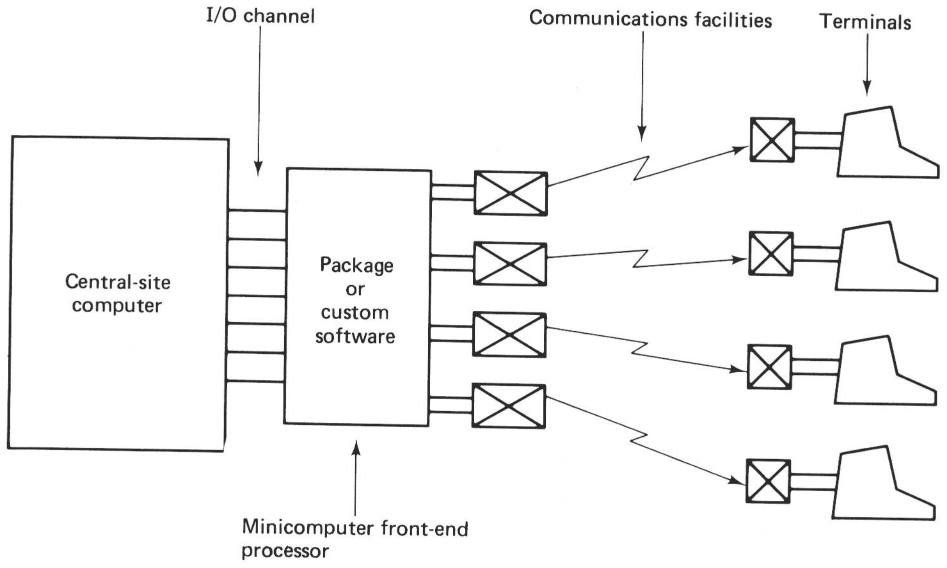
I/O channel
Communications facilities
Terminals
Central-site computer
Package or custom software
Minicomputer front-end processor

Figure 1.2
The minicomputer used in a communications control unit (front-end processor)

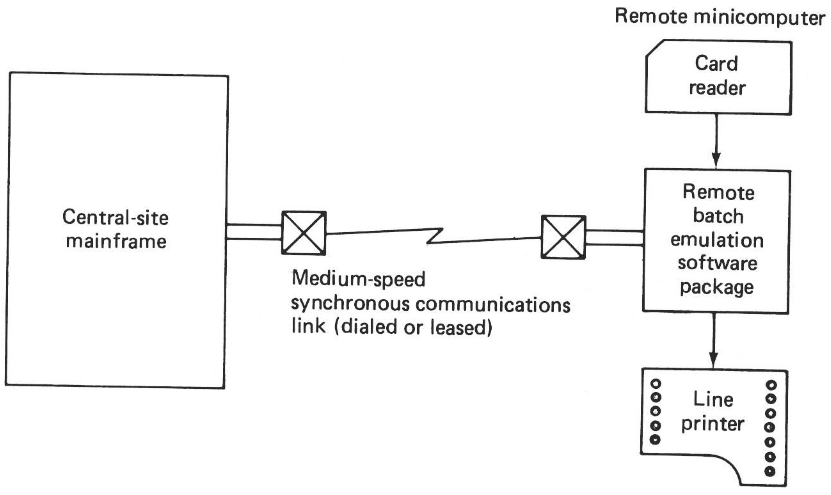
Remote minicomputer
Card reader
Remote batch emulation software package
Line printer
Central-site mainframe
Medium-speed synchronous communications link (dialed or leased)

Figure 1.3
The minicomputer used to emulate a special-purpose remote batch terminal (potential distributed processing system)

the communications control unit. Soon after that, IBM announced the 3704 and 3705 programmable control units—not necessarily because the company had changed its mind about front-ending, but because it was becoming impossible to fulfill users' increasingly heterogeneous teleprocessing needs in a hardwired controller. In fact, since the 370X was introduced, the control software in the central processing unit (CPU) has become larger, not smaller. In spite of the 370X, companies still find it cost-beneficial to front-end their mainframes with a "foreign" mini. Other suppliers who have been front-ended include Control Data Corporation, Univac, and Burroughs. (Honeywell is front-ending with its own minis.)

The minisuppliers and systems houses have also focused their attention on another communications product: the remote batch (or remote job entry) terminal. Originally the RBT was also a hardwired device, typically providing communication with remote input/output peripherals such as card readers and line printers. Independent suppliers soon found that they could put up an alternative package that included similar peripherals but used a mini-based controller and a program that emulated operations used by the mainframe manufacturers' products. Such emulators have been produced for the IBM 2780/3780, ICL 7020, CDC UT-200, Univac DCT-2000, and others.

One advantage of the mini solution is that merely by changing the emulator program one can work to a Univac data center for part of a day and to an IBM center for the rest. Moreover, once the mini is installed other peripherals (including interactive terminals) can be connected for applications. Even in its remote batch configuration, the system can be used for local vetting of the input. This is the beginning of distributed processing, which will be discussed later.

Figure 1.4 illustrates the use of minis in another type of terminal; the interactive visual display unit (VDU) or keyboard printer. In such devices the miniprocessor was closely integrated into the product. In practice many commercially available units (Datapoint, for instance) employ specially made processors, often micro-based. The processor and storage are used to refresh the cathode ray tube (CRT) and to implement various facilities (variable and protected fields, selective transmission) as well as to control the communications function to the CPU. The minis still are relatively low in power because of the limited demands made upon them. As with RBTs, such programmable terminals (especially the VDUs) were provided with emulators so they could be sold in competition with the equivalent units from the mainframe company. Such units included the IBM 2265, IBM 3275, and ICL 7181 stand-alone displays. Peripherals were generally limited to compact media such as cassette tapes and discettes (floppy discs).

Many remote terminal products use the cluster concept in which a number of terminals located in one office block or plant site can share the logic of a single local controller (see figure 1.5). Rather than using expensive modems, the terminals are connected locally via coaxial or multi-core cable (V24 or current loop). Such cheaper circuits generally give a maximum terminal-to-controller distance of less than 1000 meters without amplification or regeneration. The