M.I.T. Laboratory for Computer Science Request for Comments No. 228

Computer Systems Groups July 16, 1982

ANNUAL REPORT, 1981-82, COMPUTER SYSTEMS AND COMMUNICATIONS

by J. H. Saltzer and D. D. Clark

Attached is a preliminary version of our group's annual report. This report is a little more extensive than usual, since several projects got to a point where a summary is appropriate. Please look it over for completeness and correctness, and to see what other people are doing. Also please check attributions of who did what. If you did something that is mentioned here but not attributed to you, don't feel slighted--it's merely a screwup, nothing personal, so report it. Thanks for any help you can provide.

COMPUTER SYSTEMS AND COMMUNICATIONS

Academic Staff

J.H. Saltzer, Group Leader
D.D. Clark
F.J. Corbato
M.V. Wilkes

Research Staff

J.N. Chiappa
M.B. Greenwald
E.A. Martin
C.M. Novitsky

Graduate Students

R.W. Baldwin
G.H. Cooper
S.R. Curtis
D.L. Estrin
J. Frankel
K. Koile
L.N. Lopez
V. Singh
L. Zhang

Undergraduate Students

L.W. Allen
D.A. Bridgham
M.D. Cunningham
D.C. Feldmeier
D.E. Goldfarb
F.S. Hsu
L.J. Konopelski
C.V. Ludwig
F. Meier zu Sieker
R. Myhill
M.A. Patton
J.L. Romkey
R.S. Teal
D.L. Wilson
K.D. Wright
C.M. Zeitz

Support Staff

D.J. Fagin
N. Lyall
M.F. Webber

1. INTRODUCTION

The Computer Systems and Communication Group of the M.I.T. Laboratory for Computer Science does experimental research on the integration of computer operating systems with data communication networks. Its current projects are in four related areas: alternative local area network technologies, high-performance communication protocols, local network interconnection, and prototype experiments for future portable personal computer terminals. The work of this group is closely coordinated with that of the Computer Systems Structures Group led by Professor David Reed, and somewhat more loosely coordinated with that of the Programming Methodology Group led by Professor Barbara Liskov. Because experimental work in computer systems often requires trying ideas out in realistic application situations, some of the activities of the group mix research with the provision of service facilities to the Laboratory as a whole. The four major project areas are discussed individually in the following sections.

2. NETWORK TECHNOLOGY EXPLORATION

The underlying goal in exploring alternative network technologies is to exploit modern, low-cost digital electronics to provide relatively high-speed data communication among groups of desktop computers and related, specialized, service-providing computers. This basic goal includes the extension of high-speed data communication to personal computers located at home. One such technology, the contention-controlled bus, has been extensively developed by the Xerox Corporation under the name Ethernet. We are comparing the Ethernet with an alternative approach, the ring of digital repeaters using a token for access control. From a theoretical point of view one cannot identify persuasive arguments favoring either one of these technologies over the other. Instead it appears that the primary differences are in things best tested in the field, such as ease of trouble-isolation and repair, frequency of failure, and cost of pre-wiring a building. On each of these issues, there is reason to believe that the ring is a superior strategy.

To gain a better feel for these issues, in 1979 we installed a one megabit/second ring network, a design originally developed by the University of California at Irvine for experiments in distributed computing systems. This ring, which grew to a size of eight nodes was operated side-by-side with a locally-designed variant of the Ethernet, known as Chaosnet. Based on that experience, in 1980 we began work with a subcontractor, Proteon Associates, to develop a 10 megabit/second ring network with a simplified design. This newer ring network, known as the LCS Version 2 ring (and now available from the subcontractor under the trademark ProNet) was the target of a major software support effort of Elizabeth Martin and Larry Allen, after which it began to replace the older ring for some hosts in November, 1981. The Version 2 ring has gradually been expanded to connect five nodes. Equipment is currently on order to extend this expansion to eighteen nodes, and plans are being made to increase the number to thirty or so, with the installation of several VAX 11/750 computers for personal computing experiments. Simultaneously, Proteon Associates began commercial delivery of the design, and has completed installation of rings at ten other sites. Host-specific ring interfaces have been designed for computers using the Digital Unibus and Qbus, the Intel Multibus, the S-100 bus, and the LCS nu-bus.

So far, experience with the Version 2 ring in comparison with the Chaosnet and a more recently-installed Xerox experimental Ethernet is largely anecdotal: all these networks work well, they fail to work only rarely, and we believe that the ring is proving easier to repair. To allow us to move from anecdotal to statistical evidence, a network monitoring station is under development. We expect that

in the coming year the work in this area will comprise mostly adding stations to the ring and statistics gathering.

As an aside, public interest in ring networks received a strong boost this year with the presentation by the IBM Zurich Research Laboratory of three papers describing a ring network quite similar in design to the LCS Version 2 ring. Although no commitment has been made by IBM to use this approach, a related presentation by members of the IBM Communication Products Division to the IEEE 802 local networks standards committee on the subject of token access rings fueled speculation that IBM may eventually adopt this technology.

A second area of network technology exploration began this year: extension of megabit-per-second communication to computers installed at home. The technology being tried for this project is known in the industry as "broadband", using radio-frequency signalling over coaxial cable with components developed for two-way cable television applications. The long-range goal is to make use of the modern, high-capacity (in channel numbers) two-way CATV systems currently being installed in many large metropolitan areas, including Boston and some of its suburbs. Initially, such communications would be among a few small home computers and larger systems, such as the file-storing hosts at M.I.T. But as the number of home-sited personal computers grows, we envision the arrival of community-located services, such as laser printers and online storage, and an increase of communications among the home computers themselves, to exchange messages, programs, and for multi-player video games. The potential growing demand for such local data communication services has inspired a cooperative venture between our group and the Newton, Massachusetts cable operator, Continental Cablevision. So far this cooperative venture has been limited to discussion of the possibilities and a search for suppliers of suitable hardware.

The hard technical problems in using a residential cable television system seem to be two: accumulation of noise on the channel inbound toward the cable television headend, and assuring no interference with a fully-occupied spectrum of entertainment video materials. The first problem may require the use of digital signal regenerators in place of some of the analog amplifiers in the return path, while the second problem seems to constrain modulation techniques and spectrum shaping (at least on the outbound channel) to closely imitate a standard television signal. An interesting possibility that might tackle both these problems is the use of spread spectrum modulation methods (but present techniques would involve higher cost).

Perhaps harder than the technical problems are policy problems arising from use of systems installed for entertainment video purposes. Our studies on this topic are reported later, in the section on internetwork connection.

Finally, to learn more about the technical aspects of data communication via television cable, cooperation has begun between Ungermann-Bass (a local network supplier), and M.I.T. to allow an Ungermann-Bass subcontractor, Bolt Beranek and Newman, to use two channels of the M.I.T. educational cable television facility as a testbed for broadband equipment. Initial communication experiments began in May, 1982, and are expected to continue through the summer and fall.

3. HIGH-PERFORMANCE PROTOCOL STRUCTURES

The second area of research of this group is to understand better the causes of bad performance when computer operating systems are connected to a network. It is by now a common experience to attach a 10 megabit/second local network between two computer systems, fire up some

communications software to transfer a file, and observe an effective data rate in the range of 5 to 50 kilobits per second. In explanation of this phenomenon, the words "software overhead" usually appear, but overhead by itself does not really explain a difference between capability and achievement of more than two orders of magnitude--something more fundamental is wrong. It appears that the primary problem is in the structure of the communication network protocols themselves.

Network protocols, both design and implementation, have evolved until now in a world of telephone lines that are characterized by frequent data errors, point-to-point topology, and most important, a data rate limited to 50 kilobits/second or less--often only one or two kilobits/second. As one might expect, existing protocol design and implementation experience has been shaped by these properties of telephone lines. The thing that is not obvious (until one attaches a 10 megabit/second local network) is just how strong this shaping force has been. Our tentative conclusion from early investigations is that increasing the available data rate by two orders of magnitude so changes the environment that traditional concepts of protocol function, structure, and layers must be largely replaced with lighter-weight approaches. In addition, these lighter-weight approaches must be applied both to the protocol designs and implementations and to the operating system designs and implementations.

One slightly heretical concept that we have explored extensively this year is the complete protocol implementation that is specialized to one application. An example is the file transfer package that in a single small but not very modular program reads data from a file and emits properly-formatted data packets, packets that appear to have been neatly constructed in several layers. Because the only purpose of this package is to transfer files, most of the intermediate protocol layers do not need a full implementation--only the function required for file transfer need be there. Once this simplification has been made, it is often apparent how to reorganize the remaining functions to combine layers or do things in a different order than the layers might imply. (Karl Wright, in his undergraduate thesis describing such an implementation for the IBM Personal Computer describes this strategy as "cradle-to-grave handling".) The result can be a file transfer application package that is not built out of general-purpose subroutine packages, but that may move files at a rate ten times higher than the usual protocols. Two cradle-to-grave implementations have now been completed, the one already mentioned for the IBM Personal Computer, and a TCP/Telnet by David Clark, for the Xerox Alto.

A closely related idea is the restructuring of a protocol implementation to use implementation modularity that does not conform to the layer boundaries of the protocol. A simple example of this approach shows up in construction of a packet for a multilayered protocol. Traditionally, an application layer calls an outer protocol layer with a pointer to some data it would like to transmit. The outer protocol layer constructs a packet header, copies the application data into the packet, then calls a lower network layer. This lower layer in turn constructs a header, copies the higher layer's packet into its own data area, and calls on the next lower layer. Only when the packet finally gets to the bottom layer does anyone discover that the network is busy and the packet has to be queued anyway. After restructure, this scenario would operate quite differently: the call from the application would drop through all the layers to the bottom, leaving only notes along the way that the client has data to transmit. The next time the network becomes available, a series of calls would go backwards through the layers. The lowest layer would create a header and then call the next higher layer to fill in the data area. This series of calls propagates through the layers, perhaps all the way back to the client, creating a packet with no extra data copies and at a time when it is known that it can be dispatched immediately.

Several experimental implementations using ideas such as the one just described have been carried out this year. Liza Martin developed an IP implementation for UNIX, and Dave Clark developed a byte stream protocol implementation for the TRIPOS system while visiting the Computer Laboratory of the University of Cambridge. Larry Allen implemented a file transfer package for UNIX that achieves a useful data rate of 130 kilobits/second as compared with 45 kilobits/second for an earlier implementation with more traditional structure. Geoffrey Cooper started work on a Master's thesis on what he calls "soft layering" of protocol implementation. The idea here is that implementation of a protocol layer follow traditional lines but that it be done with knowledge both of the application that the client layer has in mind and also the implementation strategies of lower layers.

A second area of rethinking of organization applies to operating systems themselves. A high-speed communication network places tremendous stresses on the facilities that an operating system provides for coordinating parallel activities. Demands for attention from a network are usually quite unsynchronized with other activities inside the host (as compared with a disk, which usually asks for attention only after you poke it) and successive demands may arrive with only a millisecond headway separating them. More subtly, a decision to queue a packet for transmission (say in acknowledgement of one that just arrived) sometimes needs to be reevaluated if more data becomes available to add to the packet before the transmission actually occurs. Finally, whatever mechanisms are used for sharing data between parallel activities must be designed with the recognition that simple copying of data from place to place consumes an inordinate amount of time.

The folklore in this area says that one should attack these problems by minimizing the number of process scheduling operations. Such a strategy may have the unfortunate side effect of not exploiting some opportunities for parallel activity. That observation in turn leads to the idea that what is really needed is very light-weight parallel activities. We have now completed three experimental implementations of light-weight activities, one for multiple tasks within a process of the UNIX operating system, by Larry Allen, the second for a queue-driven subroutine dispatcher for the PDP-11 MOS operating system, by Noel Chiappa, and the third for a multi-task module for the TRIPOS system by David Clark. The first was the basis for a high performance TCP implementation. The second was the basis for an internet gateway implementation that is still being checked out. (An earlier prototype of this gateway has operated as a link in a chain where end-to-end data rates of over 400 kilobits/second were achieved.) The third was the basis of a revised protocol implementation that achieved a transmission rate improvement of a factor of ten (from 30 kilobits/second to 300 kilobits/second) over the previous implementation.

A third area of organizational rethinking relates to protocols themselves. Professor David Reed of the Computer Systems Structures group has led the development and implementation of two protocols that reduce the use of acknowledgements to a bare minimum. One is used for rapid transfer of files, the other for rapid transfer of bit maps for displays. It was this latter protocol that operated at the 400 kilobit/second rate mentioned earlier. Another performance-oriented protocol concept is that of source routes: the originator of a packet places a complete set of routing instructions inside the packet so that as the packet proceeds through internetwork gateways it can be forwarded without any computation or table lookup by the gateways. Vineet Singh, in a Master's thesis, developed a plan for a service that calculates source routes for use in such a system. His service allows for hierarchical organization boundaries, with a separate route-calculating service in each organization, yet it retains the essential advantage of high-performance transmission of individual packets.

Although not having exactly the same goals, another project was started in the area of protocol performance: the network attachment of low cost personal computers using the same large-scale

protocols as do large mainframes. The goal here is to provide convincing evidence that even a low-performance desk top computer can participate as a full member of a data communication network. Three IBM Personal Computers were acquired, on the basis that that recently designed machine is typical of the latest wave of 16-bit machines, with an address space large enough and a processor fast enough to allow a full protocol implementation. Since the primary interest in this project was to learn about software feasibility, and it seems likely that manufacturers will soon provide off-the-shelf network hardware for this class of machines, we chose to use as the initial hardware connection just the 9600 bit/second RS-232 asynchronous line connection that comes with the IBM Personal Computer. Several such lines are run to a Digital LSI-11 computer that is to be programmed as an internetwork gateway and attached to one of the higher speed local networks.

Much of the first stage of this project consisted in getting support tools in place to facilitate programming for the IBM Personal Computer. Lack of availability of a native assembler and awkwardness in the mechanics of use of the native compilers for the personal computer led to a decision to use a cross-assembler, cross-compiler, and downloading system from our PDP-11/45 UNIX machine, with the result that most of the programs being developed by us for the Personal Computer are in the C language. Two related tools for the Personal Computer were also developed, a terminal emulator and a clock-driven profiler that tells where programs are spending their time. David Bridgham and John Romkey have done most of this work, with both help and guidance from Wayne Gramlich.

The first completed protocol program is the file transfer package using the protocol TFTP/UDP/IP, mentioned earlier. This package achieves a useful data rate of 3500 bits/second over a 9600 bit/second line; most of the difference in these rates can be explained by low performance of the floppy disk software and hardware packages of the Personal Computer. When doing memory to memory file transfer, a data rate of about 12000 bits per second was achieved using a 19200 bit/second line between two Personal Computers. (In both cases the communication line data rate was adjusted to the lowest rate that did not add any bottlenecks.)

Two further protocol implementations are underway for the IBM Personal Computer, a remote login (Telnet/TCP/IP) protocol and an acknowledgement-free file transfer protocol (Blast/UDP/IP).

4. LOCAL NETWORK INTERCONNECTION

The third major research area of this group concerns interconnection of local area networks. Most laboratories exploring this topic have concerned themselves with only one aspect of this problem, namely that available local network technologies are limited in area to one or a few buildings and limited in connectability to a few hundred nodes. These limitations lead to the conclusion that larger networks must be built up by interconnecting smaller ones with forwarding nodes, called gateways. We have added to this consideration a second aspect that we believe is equally important in its technical impact: administrative boundaries within organizations and between organizations also shape the boundaries of local networks and add management considerations such as policy control (e.g., privacy, authenticity, and accounting) on data flow from one local network to another.

There seem to be two models on which the interconnection of a large number of local networks can be based: concatenation of adjacent local networks, or systematic hierarchical interconnection. With the first model, gateways are placed at opportune points where two or three local networks are adjacent in coverage, and communication from one point to another takes place over as many intervening local networks as necessary. In the second model, a higher-level network called a

"spine" or "backbone" is installed, with geographical coverage of the entire community of interest but with attachment only of gateways to the several local networks. With this approach, communication from one point to another goes from a node through its attached local network to a gateway that passes the message to the spine network; it travels across the spine to a second gateway directly into the local network to which the target node is attached.

When administrative boundaries are taken into consideration it is apparent that the model of concatenation of adjacent networks has serious shortcomings. For long distance communication one must depend on correct operation and benevolent administration of all the intervening local networks and gateways. Bandwidth, delay, and reliability of connections are limited by the worst example of each parameter along the path. From the point of view of the manager of one of the local networks, much of the traffic he is carrying may be "tandem" traffic, that is, long distance communication that neither originates nor terminates in his network; this makes control of load, and thus performance, for his own users hard to accomplish. The alternative, hierarchical model, with its backbone network, has corresponding advantages. The backbone network and its gateways can be centrally administered, while the local networks can be locally installed and managed. Responsibility for long-distance communication is clearly defined, which means that trouble isolation, repair, and capacity planning are easier to accomplish.

It is this line of reasoning that has led us to recommend the hierarchical model to the M.I.T. Director of Computing and Telecommunication Resources as the basis for a campus-wide network.

A second area of interest in the area of network interconnection relates to the proliferation of different communication protocols. Unfortunately, the many different sources of protocol implementations are not converging to a single approach. Instead, it appears that strong differences in opinion as to relative importance of different issues is leading to send quite different families of protocol design. The differences stem from application differences, environment differences, and sometimes issues of taste and vendor strategy. These families differ in the same ways (and for the same reasons) that computer programming languages differ. For the moment, at least, any forces for standardization appear to be neutralized by the forces of diversity, which means that every attempt at network interconnection inevitably encounters incompatible communication protocols.

At the lowest level, removing a packet from one local network (say an Ethernet) and placing it on another (say a ring), there is generally not a very serious compatibility problem, because although link-level communication protocols are often very different, they are usually very modular and higher-level protocols are often designed in anticipation of using many different link-level layers and they do not usually depend on having a single link from end to end. The gateway between local networks has a problem that is analogous to that of taking letters from a truck and putting them on a bus or handing them to a letter carrier. One must not ask the letter carrier to walk away with a 100 kg sack of mail, but unless the carrier insists he can handle only letters weighing less than 15 grams there is usually not a serious forwarding problem.

The real difficulties arise at the higher levels of protocol--the end-to-end transmission protocols and above. After several not-very-satisfying attempts to build protocol-translating gateway programs we have reached a second important conclusion concerning network interconnections that cross administrative boundaries: having a protocol translator at the gateway between two networks is about as ineffective as having a long-distance telephone operator act as a language interpreter in a call from Spain to Finland (or, more analogously, automatic translation of Cobol programs into PL/I). That is, there are certain stylized applications for which the approach can be acceptable (for example, remote login and mail forwarding), but as a general communication system it is fundamentally unworkable. The reasons are many, but mostly they boil down to non-comparable

semantics in the higher-level protocols. A simple example is translation between two different file transfer protocols, one of which requires that the first fact received be the size of the arriving file while the second allows the sender not to reveal this information until the last packet it sends. To translate between these protocols, a gateway must provide buffer storage large enough to hold the entirety of the largest file that it might be asked to forward. More subtly, since the transmission to the recipient can't begin until the sender has dispatched the last packet, the sender's protocol may, by timeout, expect commitment of the transaction long before it is clear that the recipient can make that commitment. If the translating gateway returns an early acknowledgement to the sender committing acceptance of the file, then the gateway must honor that commitment with the same degree of reliability as the intended recipient; this commitment means that the gateway must have reliable storage and recovery procedures equal in quality to those of the most sophisticated recipient system.

Even when two different protocols appear superficially similar enough to allow on-the-fly translation, often they are built on different assumptions about network delay, probability of specific kinds of error, or recovery technique; in these cases a translating gateway may have the difficult-to-predict property that it works only when traffic load is light and no errors occur; less-favorable conditions lead to communication breakdown and application failure.

The alternative to translating gateways is end-to-end agreement to use the same protocol. This approach has two implications, one for hosts and the other for gateways that connect networks. For hosts, it means that two different hosts cannot communicate unless they have some common protocol. This implication leads to multiple protocol implementations and to application-specific translation programs. For gateways, it means that packets of several different end-to-end protocols may require forwarding. Since the forwarding strategies of different protocols can be quite different, the gateway must therefore be organized to deal with multiple routing strategies, multiple addressing schemes, multiple error handling and recovery procedures, and multiple examples of local state information.

Realization of this requirement has led us to design and implement a multi-protocol, multi-network gateway package for use in network interconnection. This package, developed by Noel Chiappa, is written using the C language and the MOS operating system, and operates on Digital LSI-11 computers. It has been designed with the ability, at the lower level, to connect to any local or long-haul network; drivers for the LCS Version 1 and Version 2 rings, the Xerox experimental Ethernet, the Chaosnet, the Xerox-Dec-Intel 10 Mbit/second Ethernet, the ARPANET (by Robert Baldwin), and low speed telephone lines (by David Bridgham) have been developed or are planned. At the higher level, internetwork forwarding packages for IP and Chaos protocols have been implemented, and packages for X.25, DECNET, Xerox NS, and IBM SNA forwarding strategies have been considered as candidates for implementation. Experiments in source-route forwarding will probably also be carried out using the multi-protocol capability of this gateway package. As of this writing the package has been completed and has operated as an IP gateway between the ARPANET and the version 1 ring network.

Another example of a service built on the principle of avoiding on-the-fly protocol transformation is a multi-protocol mail forwarding facility installed this year on the Multics system. Although at the highest level there is agreement among several communities on the format of an electronic message, there are several different protocols for transfer of a message between two hosts: old and new ARPANET protocols, Chaosnet protocols, and some local variants. In some cases, the protocol that is required differs depending on the port over which the message is dispatched. The forwarding service on Multics will accept a message from any source host in any of the various protocols. It will then queue and remail the message to the intended target, using whatever mail-forwarding protocol that host requires. Michael Greenwald did most of this year's work on this facility.

The earlier section on protocol performance mentioned a thesis by Vineet Singh on the design of a service that calculates routes to be placed in a packet at its source. That work has important application for network interconnections that cross administrative lines as well, since an important virtue of a source route system is the ability to control exactly the path taken by a packet. Such control allows a packet to be directed along a path that has appropriate data rate, reliability, delay, or security, and that meets policy requirements (e.g., the packet shouldn't pass through country X or private company network Y). It also allows trouble isolation (by sending packets that have a route that takes them out to a gateway and back to the originator) from any node, a useful facility to reduce finger-pointing as the prime trouble isolation method in multivendor network paths.

As the scale of interconnection of local networks grows, and links extend from one organization to another, a problem arises of identifying the individual users of the network. Proper identification is needed to send electronic mail, and to authorize use of data or other services; it may also be needed to account correctly for use of services. Traditional authorization and naming systems have operated entirely within a single computer system, and have used techniques satisfactory for a population of from a few dozen up to maybe a thousand or so users. When connected to a network, the name of the host computer usually appears as part of the identification, say, when sending mail to someone. These schemes begin to break down when larger numbers of computers each with a smaller number of locally-assigned users appear, and as the cost of computing declines to the point that almost everyone in an organization becomes a computer user. Problems are especially apparent when user names are expressed only as three initials or some locally-known nickname.

This year we began a project to implement an on-line directory service for the M.I.T. community. The goal of this project is to explore techniques for naming people at the scale required at M.I.T. in the future--a community of 15000 people with turnover of perhaps 2000 faces (and names) each year. Two specific services are being developed by Kimberle Koile, with assistance from Felix S. Hsu. First is an interactive directory assistance service that accepts a partial, potentially ambiguous name of someone thought to be at M.I.T., and responds with a set of possible correct identifications, along with confirming information such as title or department, and office or term address. When the correct person is found, associated with it will be a standard form of that person's name, for use in sending mail and for authorization and accounting. One of the questions to be explored is the extent to which the standard form can be exactly the person's name as he is commonly known--typically a first name, middle initial, and last name.

The second service, based on the file stored by the first service, is a mail forwarding service that eliminates the need to know on which computer system at M.I.T. a person's electronic mailbox resides. Instead, one directs the mail to the person's standard name (perhaps as discovered once before by use of the directory assistance service and then tucked away in a private list of nickname--standard name pairs.)

Operation of these two services at the scale of the entire M.I.T. campus demands that much of the information base be automatically derived from other files: those managed by the registrar of students and the staff personnel department. The project began by obtaining machine readable copies of directory listings from both those sources; continued cooperation with both those organizations as well as the telecommunications office (which publishes telephone books and operates the telephone-based directory assistance service) is anticipated. Also, because the data held by these services is personal in nature, all plans for this data base are being reviewed by the M.I.T. Privacy Committee for suggestions and to spot possibly troublesome points.

As an experiment in practical problems of interconnection, Fredrich Meier zu Sieker, in his undergraduate thesis, designed a gateway to the commercial Telex service, to be usable via the local

networks by any authorized person at M.I.T. This project helps focus attention on user identification (for poorly addressed incoming messages) and accounting and authorizaton for use. An implementation of the service is underway, by Robert Myhill and Lixia Zhang.

As part of our interest in network interconnection, we have maintained a strong interest in the Internet protocol family being developed by DARPA. During this year David Clark took over the job of technical coordination of the architecture of this protocol family, and he is chairing the Internet Configuration Control Board with the responsibility for decision making this this area.

As part of this DARPA Internet project, he has prepared a number of notes which together comprise an informal implementor's guide to the protocols, with particular attention on how to produce a simple implementation that performs well. These memos will be distributed by DARPA in July.

A final project in the area of network interconnection has already been mentioned under the heading of network technology exploration--the use of cable television as a data communication medium. The interconnection aspect of this project is its policy component. There are issues of allocation of cost, of control of access to a limited resource, of provision of related services, and of separation of control over content and carriage. Communication regulations are generally formulated with two distinct classes of service in mind: common carrier, and broadcast. Policy regulations pertaining to the first focus mostly on tariffs while for the second they focus on program content: fairness, community service, etc. Using a cable for data communication involves elements of both kinds of policy regulation, requiring some innovation to deal with the situation sensibly. In addition, in the United States, cable policy regulation is typically handled at the community level, rather than state or federal, and most communities have inadequate resources and expertise to deal with either kind of policy regulation, let alone their interaction.

To shed some light on this area Deborah Estrin just completed a Master's Thesis on the policy problems of using cable television for data communications. The main contribution of the thesis is to outline the range and depth of the policy problems that are involved. In addition, one chapter offers specific suggestions to community policy makers on how to proceed.

5. PORTABLE PERSONAL TERMINAL

During the year, the group has done a preliminary study on the design of a portable terminal, small enough to be carried in the briefcase, if not the pocket. We were interested in exploring ways in which such a terminal could be made realistically useful, given that the small size must involve a severe restriction in the basic capabilities of the terminal. We assumed that individual applications would have to be reprogrammed so that they knew about and could take advantage of the specific features of the terminal. This kind of specialization has been very important when moving from traditional ASCII terminals to more sophisticated bitmap displays; it was our believe that the same specialization would be important for a terminal with restricted rather than enhanced features.

We had in mind two sorts of terminals for this project. The smaller realization consists of a single line of alphanumeric text and an undersize, but traditionally organized keyboard. This terminal configuration appealed to us because products are now appearing on the market with this approximate hardware configuration. The more sophisticated version of the terminal would have a keyboard and a multi-line display, perhaps packaged so that the display could fold up and sit at a traditional angle to the keyboard. The display technology for such a packaging is beyond our ability

to fabricate, but we thought it worth exploring the implications of this degree of sophistication, because the range of applications that could be supported was much greater than with a single line of text.

We had in mind particular applications which seemed suitable for a portable terminal. The first application studied was sending and receiving mail, which clearly benefits for a greater flexibility and freedom in the patterns of accessibility. Other applications which would be suitable for such a terminal include a portable appointment book or a database query facility. These alternative applications were not considered in detail.

The first phase of this project concentrated on the simpler of the two terminals, with a single line of display. The first question was how the limited functionality of this terminal should be organized in order to make it maximally useful. Simple experiments with traditional patterns of text display, in which the text is streamed from right to left as a continuous ribbon, quickly eliminated this pattern of display. Because of refresh limitations in the display, this pattern was limited to a very slow reading rate, which the user quickly found frustrating. In a Bachelor's thesis, Russell Houldin experimented with a number of alternative reading modes, using a simulation of an LCD display which he and David Goldfarb programmed for the bitmap display of the Alto computer. This simulation allowed us to experiment before we had implemented any prototype hardware for the terminal itself, and led to the conclusion that a rather specialized reading mode was the proper display pattern for a single line terminal. The reading mode involves dividing the text into small chunks, no more than ten to fifteen characters long (except when a single word is longer), and displaying these chunks in rapid sequence centered in the display. The reader never moves his eye, and can easily be trained to read five to ten chunks per second. In this manner, large quantities of text can be perused quickly.

In order to support this reading mode, the terminal required two special features. First, it required special keys to control the rate of text to display. The reader must easily be able to slow down, or back up and reread something. This requirement in turn implied that the terminal must have an internal buffer, rather then simply displaying text as it came down the telephone line. In fact, the buffer was additionally required in order to permit a peak reading speed which exceeded the data rate of the telephone line. We therefore developed a proposed buffer scheme, in which the application understood about the management of the buffer, and attempted to keep the buffer full, so that as the reader advanced through the buffer, the desired text was always there.

In order to explore some of these specialized features, and to learn something about the actual operation of LCD displays, Clifford Ludwig, in a Bachelor's thesis, undertook the development of a prototype terminal. His goal was not to produce something which was in any sense properly packaged for portability, but rather to produce something which contained hardware and software of the correct power. The prototype contained a Z80 microprocessor, a suitable amount of memory for buffering, and a special display peripheral which consisted of one line, 36 characters long, of LCD display. The prototype was implemented, and certain of the experiments which had been performed on the Alto simulator were recreated on the actual display.

This initial prototype taught us a number of things. First, LCD displays, although clearly desirable from a point of view of power consumption, are marginal for this purpose, because they cannot alter the display fast enough to suit the peak reading speed of a trained user. Blurring and cloudiness occur, even with very careful tuning of the electrical parameters of the driving circuitry. Therefore, more experimentation with the detailed characteristics of LCD displays are required. Second, in order to drive the display properly, specialized display chips are required, which are now available only in a form too bulky to be properly packaged. A commercial version of this terminal would require specialized LSI. Third, a Z80 is clearly powerful enough to provide the display, keyboard, and telephone line controls needed for the terminal.

A related project in this area explored the idea that speech input and output could be an important part of this terminal's function. The intention was not to rely on speech recognition or synthesis, two very difficult problems, but to use speech storage as a way of passing information through the terminal from one human to another without the necessity of dealing with a restricted display or an undersized keyboard. For example, in the context of mail, the user might employ the keyboard to specify the recipient and subject matter of the message, but might dictate the message itself. Alternatively, the terminal could be used as a dictating machine, with later playback and transcription of the information into ASCII representation of the text. Two projects were undertaken to explore the feasibility of integrating speech into this terminal. First, a study was undertaken of speech digitization and storage techniques, to find out how many bits per second of digital information would be required in order to store intelligible speech. Much previous work has been done in this area, but the specific issue we were concerned with was the fact that speech had been transmitted through a telephone line before being digitized, which changes its spectral characteristics. In particular, efficient storage algorithms such as linear predictive coding are unsuitable. We thus did a preliminary experiment with delta modulation techniques, to determine whether they could produce intelligible speech with a reasonable bit rate. In a Bachelor's thesis, David Teller clearly demonstrated that delta modulation is a suitable technique in this context, but he also demonstrated that succesful utilization of that strategy in this context would require the development of specialized hardware for the purpose, as opposed to the converter card which we had purchased for initial trials.

The other experiment related to speech involved the development of a specialized modem to connect the terminal to the host. Since we desired to send both digitized information and analog speech over the phone line, it was necessary that the modem be switched off and on by computer control to permit the telephone line to be used alternatively for both digital and analog transmission. Further, at the terminal end, it was necessary to have some form of microphone and speaker which the human could use for sending and receiving speech. Since a telephone line was being used for the transmission of this information, we explored whether a telephone handset could be plugged into the terminal for this purpose. Carol Novitsky did a preliminary study in this area, with the particular goal of determining whether any of the specialized low-speed modem chips which are now available could be used at a bit rate above 300 bits per second. This involved the careful design of high order analog bandpass filters. The results of this experiment were somewhat encouraging, but it is clear that to achieve high bandwidth over a phone line with a very small space and power available in the terminal represent a substantial analog engineering job. We feel that pushing further in this direction should be delayed until we have a better idea of the actual bandwidth requirements.

The final study in this area was an evaluation of mail on the more sophisticated version of the portable terminal, in which many lines of display might be available, perhaps as many as on a traditional CRT. In this case, the study focused on a rather different area of terminal function. For the single line display, it had been assumed that the terminal would be connected to the host using a telephone line whenever the terminal was in use. Because of the space limitations in the terminal, there was no other useful way to imagine utilizing the terminal. However, if the terminal is somewhat larger, it is possible to imagine enough buffering in the terminal that it could serve as an independent computer, being connected to the host only now and then to refresh its internal storage. This is a particularly appealing pattern for mail, since the terminal could be connected to the computer a few times a day to receive any new messages, and then could be carried off to permit the reading of mail whenever convenient. The problem which arises from this pattern of use is one of management of duplicate copies of information. In particular, when a host transfers a piece of mail to the terminal, it is not reasonable for the host to discard its copy of the mail. The terminal, being portable, could be lost or damaged, and it would be inappropriate if mail were lost under these circumstances. Therefore, a synchronizing strategy is required between the copies of the mail stored in the terminal

and the host, to ensure that mail is not lost. In fact, it is very important that the host keep a copy of any mail delivered to te terminal, because, as a result of reading the mail, the user may instruct that the mail be disposed of in various ways, perhaps being saved or forwarded or used as the text of a reply. Such functions are much more convenient if the host has its own copy of the mail. Michael Patton, in a Bachelor's thesis, did a first study of the way in which the host and the terminal would cooperate to provide this kind of functionality.

This particular distributed database is a very interesting one for study in general, because it has a very different usage pattern from those traditionally considered in research on distributed systems. Most researchers assume that the database is connected by communications facilities almost all the time, but occasionally partitioned. In this case the database is almost always partitioned, but very occasionally connected. In general, such a partitioned database could not be made to work well, but in the case of mail, because of its particular characteristics, the partitioning does not seem to be a difficult problem. This is an interesting observation, which suggests that more study of application-dependent distributed systems is appropriate.

6. XEROX GRANT ACTIVITIES

An incidental activity of this group is administration of a University Grant of the Xerox Corporation, consisting of 18 Alto personal computers, a Dover laser printer, a file server, an experimental Ethernet local communications network, and related supporting software and facilities. These facilities are now an integral part of the resources of the laboratory as a whole. Eleven of the Altos are currently used in direct support of research projects in four LCS groups, while six are assigned to different groups throughout the laboratory primarily to provide experience with the cultural effect of having high performance personal computers nearby. (The 18th Alto is used as a maintenance spare and for overflow usage by the other groups.) This report briefly outlines the ways in which the Xerox grant facilities have been used. More extensive project descriptions will be found in the annual reports of the individual groups doing the projects.

The Computer Systems and Communications group has undertaken three separate programming projects on the Alto, all related to its protocol effectiveness research. The first, by David Clark, was to create a Dover spooler service, a program that accepts files in the ARPA TFTP/IP protocol, and relays these files to the Dover laser printer in the Xerox Pup protocol. A second project, also by David Clark, was implementation of a very small and fast user Telnet/TCP package that permits the Alto to be used as a remote terminal via internet gateways. Both these systems are now in production use; the Telnet/TCP was recently used to log into a Digital PDP-11 computer located in Norway, with several networks and gateways in between. The third project, by Geoffrey Cooper, was an implementation of the ANGEL protocol, an IP-based reliable datagram transfer service that uses the soft layering ideas of Cooper's S.M. thesis.

During the past year,the Computer Systems Structures group has used the Altos in three ways. First, the Swallow distributed data storage system repository was built using Mesa on an Alto. The repository is now operational in a test configuration. The availability of Altos was a crucial factor in being able to pursue this work. Second, the group began to use Altos to provide remote, network-attached bitmap displays for single user VAX-750's, using special, high-speed protocols. Third, they used Altos as high performance network terminals, providing 80X60 character, network-attached access to hosts using TCP on the ARPA Internet.

The Systematic Program Development group decided to abandon its attempt to use an Alto as the front end of its specification editor. This decision was based upon the prohibitively bad performance of a prototype implementation, unhappiness with the Mesa 5 programming environment, and a reluctance to increase an investment in what appears to be (for LCS) a dead-end line of machines. Members of the SPDG did get considerable use out of the text preparation facilities of the Alto. Compatibility with text preparation facilities at Xerox PARC greatly facilitated cooperative work with James Horning.

The Functional Languages and Architectures group is developing an advanced computer architecture and a hardware prototype based on "dataflow" principles. That group has found that the Xerox Alto computer is an asset in carrying out its research and in particular they have used the Alto in the following ways:

1) The Draw program has been invaluable in producing high quality figures for many designs. Members of the group have commented that the flexibility of the Draw system has led to qualitative improvements in their designs and documentation;

2) The group is using a stripped-down M68000 microprocessor as an I/O and control processor for each processor of its prototype dataflow machine. By connecting a M68000 to the Alto via an 8-bit parallel port they have been able to quickly tap onto the Alto's existing file system and Ethernet without a large software/hardware effort. A student is working on microcoding the Alto parallel port software to improve the performance of the link.

The group is also considering using the Alto programs Sil, Analyze, Route, and Build to generate wire lists for the prototype. Route's capability to generate Multiwire lists is very attractive since they are planning to use Multiwire boards in the prototype.

James Frankel, a Ph. D. candidate at Harvard University, has been using the Alto's as a distributed computer system on which to implement a prototype of some software ideas in his dissertation. The dissertation, "The Architecture of Closely-coupled Distributed Computers and their Language Processors," deals with the hardware and software design of a shared memory multiprocessor.

The system implemented on the Alto's, a parallel-executing Pascal compiler, compiles syntactic structures of the source code concurrently. Thus, the prototype begins by compiling a program on a single processor. That machine compiles all declarations and then initiates the compilation of nested procedures on other machines passing to them the symbol table that was generated. The initial processor then compiles and produces code for its code block. The machines on which compilations were spawned perform the same sequence of operations in turn.

The generalization of these ideas is that the parallelism for multiprocessor computer systems should come from the data flow rather than the control flow present in programs. Furthermore, the premise is that "coarse data flow," data flow where the data is larger than a single word (for example, the size of a procedure, statement, or expression, for a compiler), allows conventional processors to run in a multiprocessor configuration, reduces communications and, thus, contention for shared memory, and does this without the drawbacks present in data flow machines (inability to deal directly with data structures, difficulties with recursive and reentrant procedures, etc.).

A number of tools were developed during the work on the project. With the cooperation of Roy Levin and other members of the Xerox PARC Computer Science Lab, Mr. Frankel wrote a package to create "boot" files from Mesa 5.0 programs. This package was distributed to all of the universities in

the Xerox University Grant Program. The Diagnostic Memory Test, DMT, program was modified to allow an Alto to be down-loaded over the EtherNet only if there are no disks ready in the Alto. A page level file server was written to act as shared memory over the 3 MHz Experimental EtherNet. A client interface to the page server and a stream interface to the client interface were written. When bound with these programs, any Mesa program that was written to access files from disk will access those files from the page server.

The Pascal compiler itself is written in Pascal and compiles into a byte coded instruction set called Pascal Byte Codes. These byte codes are interpreted by the Pascal Byte Machine, PBM, which is itself written in Mesa.

More detailed information is available in Mr. Frankel's dissertation to be completed in August, 1982.

The Dover laser printer is one of the most popular facilities in the laboratory, receiving wide use from almost every computer system of both the Artificial Intelligence Laboratory and the Laboratory for Computer Science. An indication of its popularity is the rate that it consumes paper: 250,000 sheets/month at present. Almost all technical reports, technical memoranda, and submissions of papers to journals from the two laboratories are prepared with the help of this printer.

Publications

1. Clark, D., "Local Networks," to be published in COMPUTER, accepted April 1982.

2. Corbato, F., "An M.I.T. Campus Computer Network," Campus Computer Network Group Memo No. 1, M.I.T., Cambridge, MA., 1981.

3. Corbato, F., "Time Sharing," Encyclopedia of Computer Science, A. Ralson, Editor, Second Edition, van Nostrand Reinhold Co., New York, in press.

4. Estrin, D., "Data Communications via Cable Television Networks: Technical and Policy Considerations," MIT/LCS/TR-273, MIT, Laboratory for Computer Science, Cambridge, MA., May 1982.

5. Saltzer, J., "Communication Ring Initialization Without Central Control," MIT/LCS/TM-202, MIT, Laboratory for Computer Science, Cambridge, MA., December 1981.

6. Saltzer, F., "On the Naming and Binding of Network Destinations," International Symposium on Local Computer Networks, Florence, Italy, April 1982, pp. 311-317.

7. Saltzer, J., Clark, D., and Pogran, K., "Why a Ring?" Seventh Data Communications Symposium, Mexico City, Mexico, October 1981, pp. 211-217.

8. Singh, V., "The Design of a Routing Service for Campus-Wide Internet Transport," MIT/LCS/TR-270, MIT, Laboratory for Computer Science, Cambridge, MA., August 1981.

9. Wright, K., "A File Transfer Program for a Personal Computer," MIT/LCS/TM-217, MIT, Laboratory for Computer Science, Cambridge, MA., April 1982.

Theses Completed

1. Baldwin, R., "An Evaluation of the Recursive Machine Architecture," M.S. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., June 1981. (Also S.B.)

2. Estrin, D., "Data Communications via Cable Television Networks: Technical and Policy Considerations," M.S. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., May 1982.

3. Houldin, R., "Formats and Controls for a One-Line Computer Terminal Display," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., August, 1981.

4. Ludwig, C., "A Personal Portable Terminal," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., February 1982.

5. Martinez, D., "A Central Switcher for Message-Oriented Computer Input/Output," M.S. thesis, MIT, Department of Electrical Engineering and Computer Science, May 1982. (Also S.B.)

6. Meier zu Sieker, F., "A Telex Gateway for the Internet," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., May 1982.

7. Patton, M., "Integrating Disconnected Personal Computers into an Electronic Mail System," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., May 1982.

8. Powell, R., "Microprocessor-Based Floppy Disk Controller," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., May 1982.

9. Singh, V., "The Design of a Routing Service for Campus-Wide internet Transport," Ph.D. dissertation, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., August 1981.

10. Teller, D., "Efficient Storage of Digitized Speech," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambirdge, MA., May 1982.

11. Tou, F., "Information Retrieval in a KL-ONE Data Base," S.M. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., June 1982. (Also S.B.)

12. Wright, K., "A File Transfer Program for a Personal Computer," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., April 1982.

13. York, W., "Command Completion in the Multics Environment," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., June 1982.

Theses in Progress

1. Cooper, G., "An Argument for Soft Layering of Protocols," M.S. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., expected September 1982.

2. Greenwald, M., "Operating System Support for Closely Cooperating Talks," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., expected December 1982.

3. Koile, K., "An On-line M.I.T. Directory Service," M.S. thesis, MIT, Deparmtent of Electrical Engineering and Computer Science, Cambridge, MA., expected June 1983.

4. Konopelski, L., "Implementing Internet Remote Login on a Personal Computer," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., expected August 1982.

5. Lopez, L., "Gateway Congestion Control," M.S. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., expected January 1983.

6. Roush, P., "Computerized Scheduling of Intramural Sports," S.B. thesis, MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA., expected August 1982.

Talks

1. Chiappa, N., "Developments in Fault Tolerant Local Networks at M.I.T.," South Padras Island, Texas, February 1982.

2. Clark, D., "The LCS Ringnet Project,"

   University of Strathclyde, Glasgow, Scotland, April 1982;

   University of Kent, Canterbury, England, April 1982;

   Siemens, Munich, Germany, May 1982.

3. Clark , D., "Problems in Local Network Interconnection," Los Angeles, CA., November 1981.

4. Clark, D., "The ARPA Internet Project,"

   Computer Laboratory, Cambridge University, Cambridge, England, May 1982;

   IBM Zurich Research laboratory, Zurich, Switzerland, May 1982.

5. Corbato, F., Participation in the Communications Policy Seminar on Ad Hoc Networks of Microcomputers, M.I.T., Cambridge, MA., April 15, 1982

6. Martin, E., "Network Handling on a Small Computer Running UNIX," Internet Working Meeting, Cambridge, MA., March 25, 1982.

7. Saltzer, J., Session Chairman, "Local Networking Approa<ches," ACM 81, Los Angeles, CA., November 1981.

8. Saltzer, J., "Technology, Bureaucra<cy Avoidance, and Distributed Computer Systems"; "Intriguing Ideas About Computer Systems"; and "Heretical Ideas About Local Networks," 1982 George Forsythe Lecturer, Stanford University, Stanford, CA., January 1982.

9. Saltzer, J., "Technology, Bureaucracy Avoidance, and Distributed Computer Systems," invited lecture, First Colombian conference on computer science, informatics and related sciences, Bogota, Colombia, South America, March 1982.

Committees

1. Clark, D. D., DARPA/TCP Working Group (Chairman)

2. Chiappa, J. N., DARPA/TCP Working Group

3. Martin, E. A., DARPA/TCP Working Group

4. Saltzer, J. H., DoD/DDRE Security Working Group Member

5. Saltzer, J. H., Chairman, 9th ACM Symposium on Operating Systems Principles

6. Saltzer, J. H., Program Committee, IFIP/TC6 Working Conference on Interconnected High Performance Personal Computing Systems