EDP ANALYZER

JUNE, 1980
VOL. 18, NO. 6

IN YOUR FUTURE: LOCAL COMPUTER NETWORKS

Possibly—just possibly—a common reaction to our title this month might be, "*What* is a local computer network? And why should I be interested in it?" Well, it just may be that local computer networks are being installed almost as fast as the term is becoming known. These networks offer a number of benefits that we think you should know about. (Oh, yes, here is that definition: "A communications network that has a length of up to a few thousand feet, to which computers as well as peripheral devices can be attached; it provides communications between any two attached units, does not use common carrier services, and has a transmission speed in the millions of bits per second.")

Boeing Computer Services Company (BCS) is a division of The Boeing Company, the large aerospace manufacturer which has its headquarters near Seattle, Washington. BCS provides the bulk of the computing services for Boeing, and in addition sells computing services and software to outside organizations. BCS employs over 6,500 people.

BCS has three main processing centers—at Renton and Kent, Washington (both near Seattle) and at Vienna, Virginia, just outside Washington D.C. Each center has multiple large CPUs. At Renton, for instance, they have two IBM 3031s and six CDC Cyber 170s (including 175, 750, and 760s).

In recent years, BCS management had become concerned about some of the constraints that are associated with most mainframe computers. For instance, remote batch work has been a rapidly increasing part of the BCS workload. The Cyber machines can perform batch work, but they do not support a remote job entry service that BCS would be willing to use. The IBM machines have an efficient remote job entry facility but cannot easily make use of available computing capacity on the Cybers. Also, the Cybers are somewhat limited in their ability to share peripherals. And so on.

What BCS wanted was a way to bring remote batch work in over their data communications network and then allocate

ISSN 0012-7523.

that work to whichever CPU had the capacity to handle it at the moment.

In early 1977, one of the BCS people read about a new product offering—HYPERchannel™, by Network Systems Corporation (NSC) in Minneapolis, Minnesota. One of the BCS managers visited NSC, saw a demonstration, liked what he saw, and recommended ordering it. HYPERchannel is a local computer network designed primarily for use with the larger, faster computers. BCS entered an order, and the first equipment arrived in May 1978. BCS was one of NSC's early customers, and some of the units received were prototypes. The HYPERchannel network was up and running by the end of August.

BCS uses HYPERchannel in the following way. Remote batch work flows into the Renton center by regular data communications, into an IBM 3031. The 3031 stores each batch job on disk, as it is received. The 3031 is tied to a Cyber 175 by HYPERchannel. At appropriate times, the 3031 asks the Cyber if it is ready to accept another job. When told yes, the 3031 ships the job across to the Cyber; the rated transmission speed of HYPERchannel is 50 million bits per second. The Cyber receives the job and stores it in its input job queue. After the Cyber finishes the job, it stores the results, asks the 3031 if it can send over the results, and does so when it gets the go-ahead. The 3031 then forwards the results to the appropriate user terminal.

So the local computer network has, in fact, given BCS what was sought—an economical, practical way to have the Cyber work on remote batch work.

But an interesting change in viewpoint has occurred at BCS. As important as this accomplishment has been, it now takes second place to another capability. HYPERchannel makes it feasible for BCS to bring in just about *any* computer, without worrying about compatibility. They have already tied a DEC PDP-11 to the network, and will soon be doing the same thing with a Cray-1 and a back-end data storage system.

BCS now has more latitude for selecting the most appropriate hardware for its needs. To put this concept to work, all that BCS has to do is to create the software interface and attach the new equipment to the network.

Xerox Corporation

Xerox Corporation, with headquarters in Stamford, Connecticut, has an annual sales volume of over $7 billion. It is ranked by *Fortune* magazine as the 40th largest industrial corporation in the U.S. In addition to its well-known line of photocopiers, Xerox also manufactures and sells computer equipment, facsimile equipment, and other components for information handling.

James White, of the Xerox Office Products Division, discussed his company's use of local computer networks during a panel session at the 1979 National Computer Conference. The particular local computer network he discussed was Ethernet, developed at the Xerox Palo Alto Research Center in the mid-1970s. A good number of these networks are now in use throughout Xerox, he said, and they provide network inter-connection among hundreds of Xerox mini-computer work-stations. These work-stations use a 16-bit mini with 64k words or more of main memory and 2 1/2 mbytes of disk storage.

Ethernet was originally developed as an experimental network. (The term 'Ether' is taken from the historical *luminiferous ether* through which electromagnetic radiations were once thought to propagate, say the authors of Reference 3; it means the same as trunk or bus.) The network worked so well that it has been put into everyday use within Xerox. Late last year, Xerox announced Ethernet as a commercial product, to go with the company's new 860 office information system. For a very readable description of Ethernet, see Reference 3.

Current uses of Ethernet include the following, said White. The networks: (1) connect work-stations to larger time-sharing computers, (2) transfer files between work-stations, or between a work-station and central data storage, and support (3) electronic mail, (4) distribution of software to the work-stations, and (5) experiments with distributed computing. Of these, the second and third constitute the bulk of the use, said White.

Structure of Ethernet. The original goals of the Ethernet project were to create an inexpensive local computer network that would be highly reliable. The reliability was to be accomplished by simplicity of design, which included a passive transmission medium; the trunk can be a coaxial cable, optical fibers, or even twisted pairs of wires.

An Ethernet consists of a single trunk that is up to one kilometer in length, and to which as many as 256 stations can be attached. The trunk—a coaxial cable, say—has terminators at each end to prevent the reflection of signals; signals are entered into the trunk, are received by all stations attached to the trunk, and are then dissipated when they reach the two ends of the trunk. There is therefore just a single path between a source work-station and the destination station.

The transmission speed on the trunk is 3 million bits per second.

To expand a local network—for attaching more stations, or for handling more traffic, or for attaching to, say, an external network—two or more Ethernets can be inter-connected. Such inter-connections involve the use of active elements (repeaters, gateways, or filters) in the transmission medium; other than this, the transmission medium is passive in the same sense that a piece of cable is passive.

Each work-station taps onto the trunk via a commercial CATV (community antenna TV) tap and a relatively simple transmitter and receiver (transceiver). When a work-station has a message to transmit to another station, it breaks its message into packets and transmits the packets one at a time. Each packet has a header section; the first 8-bit byte gives the address of the destination station, the second byte gives the address of the sending station, then a packet sequence number, and finally the data. Each packet is just injected onto the trunk, once access to the trunk has been obtained. All stations receive each packet, but only the destination station copies the packet; the other stations discard it.

Access to the trunk is by contention. Before a station begins to transmit, it senses the trunk. If there is transmission already on the trunk, the station's transmission is delayed. If two stations start to transmit essentially simultaneously, a *collision* results; it is sensed quickly, both stations abort their transmissions, and then use a rather simple but effective re-try scheme that involves different length delays. Each packet has a 16-bit cyclical checksum attached to it. This checksum is re-computed at the destination and compared with the one received. If there is a discrepancy, the packet has been garbled and no acknowledgment is sent. The sending station, not receiving an acknowledgment within a designated period of time, automatically re-transmits the packet.

White reported on some performance measurements that were made on one of the Ethernets. This net had 120 stations attached to it, and each station used the network sometime during each working day. An average of 300 million bytes of traffic were transmitted each 24-hour day on this one net. But during the busiest hour, only 4% of the theoretical capacity of the network was used by the transmissions; during the busiest minute, 17% of the capacity was used, and during the busiest second, 37% of the capacity was used. Over 99% of the packets were transmitted with no delay at the transmitting station due to the trunk being busy. Of the less than 1% that encountered a delay, only a very small number (0.03 of 1%) involved collisions.

Errors were detected in an average of 1 packet in 6,000 transmitted.

Because of the simplicity and low cost of the design, the designers of Ethernet say that the network cannot *guarantee* error-free delivery of any single packet. Ethernet does achieve economy of transmission and high reliability averaged over many packets. But packets may be lost due to collisions with other packets, or from impulse noise on the trunk, or because of an inactive receiver at the destination, or such. So user protocols are needed to detect and correct such occurrences, as dictated by user needs.

The cost of a coaxial cable version of Ethernet, say the designers, is insignificant relative to the cost of the distributed computing systems that are supported by the network.

TRW Systems Group

The TRW Defense and Space Systems Group (DSSG), a part of Cleveland-based TRW, Inc., has its headquarters in Redondo Beach, California. DSSG is engaged in the development of complex hardware/software systems for customers, primarily the U. S. government. A large number of these systems are for military command and control applications.

We talked to engineers at DSSG's mini-computer and information technology laboratory about their use of a local computer network they had developed. The first components of the network—the trunk and two CPU interface adapters—were received from Network Systems Corporation in mid-1978. A third processor adapter was installed last year.

DSSG had a number of reasons for designing and implementing the network; however, two were paramount. First, they foresaw a need for a network test-bed to use as a tool in doing further research and development concerning computer networking. In addition to in-house research and development, they were also interested in implementing such networks in systems that they build for customers. Further, the network would provide training for their own engineering personnel in the design, installation, and operation of computer networks.

The second reason for this network research and development was to further their expertise in fiber optics technology. In addition to installing a coaxial cable trunk for their network, they also installed a fiber optic trunk for data transmission.

In DSSG's view, the networking of computer systems to perform distributed data processing tasks is the here-and-now of computer technology. Large problems are being sub-divided into a series of smaller sub-problems which can be handled on mini-computers. Real economies can be achieved by this sub-dividing approach—for instance, as in the case of back-up computers. If they create a command and control system for a customer that uses a large central computer, then that computer will have to have another large computer just like it as back-up, at a hefty additional cost. But if a network of minis is used, a few extra minis can provide the needed back-up for the whole network.

In their own mini-computer systems facility, they foresee a network of minis, where user programs that reside on one mini can communicate with co-operating user programs residing on any other mini in the network. Currently, they have several types of computers connected to the network: Interdata 8/32, DEC PDP-11/34 and 11/04 minis, and a large main frame Cyber 175. In addition, they foresee connecting additional mini-computer types into the network.

DSSG has developed network support services software to use with this network. This includes higher level protocols and functions by which a program in one computer can call upon the services of a program in another computer, as well as an input-output driver which facilitates input-output from computer to network and vice versa.

As previously mentioned, one of the network project goals was to enhance their expertise in fiber optics technology. One of the two network trunks that DSSG has developed employs a fiber optic transmission medium; the other trunk utilizes a coaxial cable (coax) medium. Fiber optics trunks have several advantages over coax, one of which is security. There are no electromagnetic emanations from the fiber optic material, which prevents surreptitious data acquisition by an unauthorized source. Also, fiber optics trunks can transmit signals satisfactorily over greater distances without regeneration than can coax. Coax is a good transmission medium up to about 2000 feet or so, before signal regeneration is necessary. A fiber optics trunk may double or even triple this distance. Thirdly, fiber optics trunks potentially have a higher data transfer speed than coax. Today's coax trunks and adapters can operate reliably at an upper limit of 50 million bits per second. While fiber optics trunks can operate at higher speeds, the network adapters (interfaces) will have to be re-designed to accomodate the full potential of fiber optic trunks.

So TRW's Defense and Space Systems Group is putting their test-bed local computer network to a number of interesting uses.

Iowa State University

The computation center at Iowa State University, Ames, Iowa, has developed and put into use a network of mini- and micro-computers, for experiment control and data acquisition. This network is also tied to a central DEC PDP-11/34 that operates under the UNIX operating system. Over 25 of the micros are Commodore PET personal computers. One goal of the project is to make both the network and the small computers easy to use by inexpert users.

This network, like the one just discussed, has been a research and development project not aimed at daily productive use. But, in fact, it is being used daily in a productive manner by such departments as home economics, veterinary medicine, industrial engineering, and electrical engineering. Further, other departments want to start using the network.

We will give a brief description of the network. For further details, see Reference 1, paper by A. V. Pohm.

In structure, this Iowa State network is a recirculating ring or loop. It has four types of stations attached to it. (1) Central communicating stations can handle up to eight mini- or micro-computers each; they are the prime stations for attaching personal computers to the network, for providing communications between the small computers and the PDP-11; (2) general purpose stations provide communications between a terminal and any other station on the network, not just the PDP-11; (3) central stations interface the PDP-11 to the network, and (4) the master station provides the clocking, error logging, automatic shutdown when errors become excessive, and so on.

The trunk is a coaxial cable, about one mile long, that goes between buildings on the campus in an underground steam tunnel. As mentioned, this trunk is in the form of a ring in which information re-circulates at an effective rate of over 3 megabits per second. The signals are regenerated as they pass through each station interface. The re-circulated information is divided into 256 'slots' of 12 bits each—very much like a freight train with 256 cars that continuously circles a track. Of these 256 slots, 254 of them are dedicated to specific stations attached to the network. As the 'train' of slots goes by, a station can pick up 12 bits of information from, and/or insert 12 bits of information into, its assigned slot. The timing is such that the eight small computers that can be tied to a central communicating station can each be served at the rate of 1200 bits per second.

The personal computers can operate in either of two modes on this network. One is a terminal mode, where the computer

acts as a time-sharing terminal with the PDP-11; a 2k byte program in the personal computer is used for this purpose. The other mode is a stand-alone compute mode but with access to a central shared disk storage system.

Compilers and cross-assemblers are available on the PDP-11 by which programs can be developed and stored centrally and then down-loaded to the small computers. This facility is available not only for the PET computers but also for computers that use the Intel 8080 and MOS Tech 6502 micro-processors.

According to Pohm, the cost of this network has been "modest."

Local computer networks

Of the four examples of local computer networks just described, two use a commercially available product (the HYPERchannel of Network Systems Corporation) and two were developed in-house.

All four of the networks—BCS, Xerox, TRW, and Iowa State University—are being used to tie together dissimilar computers. With HYPERchannel, these can be the largest and fastest of today's computers—but the networks can also have mini- and micro-computers tied into them. In the case of Xerox, the networks tie together clusters of work-stations, plus a DEC PDP-10 and some Data General Novas.

These are only four of the examples that we could have cited. Some of our listed references include papers about other local computer networks, particularly the proceedings listed under References 1 and 2.

As one reads these references and attends conferences on this subject (for instance, we have attended sessions at two recent computer conferences, plus one conference devoted entirely to local networks), it is apparent that this is an area of high activity. Whether one is concerned with large computers and/or with small ones, the reaction of the people working in this field is the same: local computer networks are a wave of the future.

But we found, during our study of the subject, that some data processing people are as yet unfamiliar with local computer networks. It is appropriate, then, for us to give a few basics.

What is a local computer network? As we indicated at the beginning of this report, the term is hard to define. The term implies that two or more computers, located in fairly close proximity, are connected together. But it can be more than computers that are tied to the network; there can also be peripherals (disks, tapes, printers, etc), as well as connections to other networks. The term also implies that the units are rather close together. In fact, they may be a few feet apart or thousands of feet apart. The term, by itself, does not make it clear that local computer networks typically use their own transmission media, such as coaxial cables or fiber optic cables and interface units, rather than conventional common carrier circuits and conventional data communications modems.

In practice, local computer networks usually will tie together computers (and perhaps peripherals) within one building, or in reasonably adjacent buildings.

How did they evolve? The first type of data communications for computer use was terminal-to-terminal communications, begun back in the 1950s. The first form of this was punched card transmission. This was soon followed by terminal-to-computer data communications.

The developments up to this point were mainly low speed in nature, with speeds of perhaps hundreds of bits per second. The next step in the evolving data communications brought high speed transmission: computer-to-computer and computer-to-peripheral communications. Further, this development had two main branches—where the units were tightly coupled and where they were loosely coupled. Master/slave computer configurations, where the

two CPUs are connected by cables, are examples of tightly coupled, high speed computer-to-computer communications. Host computers tied to the ARPA Net are examples of loosely coupled, high speed computer-to-computer communications.

These developments set the stage for the emergence of local computer networks. What was seen to be needed was something that had some characteristics of both the tightly coupled and the loosely coupled networks. What was wanted was a means of inter-connecting very dissimilar computers, as in the case of the ARPA Net, but without using conventional data communications, as in the case of master/slave configurations.

The first company to offer a commercial product in this area was Network Systems Corporation, of Minneapolis, Minn. (HYPERchannel, Reference 5). The Minneapolis-St. Paul area has been closely associated with the development of high speed computers—for instance, at Control Data Corporation and in the UNIVAC 1100 series. NSC founders came from this environment.

What NSC brought to the market was (1) a passive trunk network, using coaxial cable, to which CPUs and peripherals can be attached by means of (2) 'adapters'. The adapters handle the problem of physically accessing the trunk, under a contention access method, and the lowest level protocols for the handling of packets.

As of this point in time, NSC has developed adapters for the IBM 360/370 (Models 148 and up) and 303X computers (with the 4300 computers in the offing), CDC 6000, 7000, Star, and Cyber 170 computers, Cray-1, UNIVAC 1100, and the IBM-compatible computers such as Amdahl, FACOM, and Omega computers. Under development are adapters for Burroughs 7600 and Honeywell Level 66 computers. Mini-computer adapters include DEC PDP-11 and VAX, Data General Nova and Eclipse, Modcomp, and SEL; under development are adapters for PDP-10 and Tandem. For directly attaching peripheral controllers to a network, adapters are available for IBM disks, tapes, and printers.

In the words of one user we talked to, "NSC's HYPERchannel offered capabilities that we just did not expect; there was nothing else on the market comparable to it at the time."

Why the need for these networks?

Here are some of the reasons that local computer networks are being enthusiastically received by their users.

Alleviate operating problems. We see two environments in which local computer networks can help alleviate today's operating problems.

Big systems environments. Many users of big computers need to, or desire to, spread their computing equipment among several rooms or buildings, or on different floors of the same building. And they often want to share critical resources—such as a database—among these computers. The users often want high speed communications in these situations—in millions of bits per second; at these speeds, maximum cable lengths are normally a few hundred feet. The users would like to remove restrictions on the amount of sharing that can be done—say, a restriction that says disk storage can be shared by no more than four processors. And users would like to use storage in the network as a buffer, to be able to transmit from CPU to buffer at CPU speed, and from buffer to peripheral at peripheral speed.

Further, they would like to be able to bring in a new type of computer and easily have it share these critical resources. They would like to overcome the compatability problem--having to force new technology to be compatible with their existing hardware and operating systems.

Small systems environment. Users of small computers (minis and micros) would like those computers to be able to share important resources, such as large volume data storage, a database, high speed print-

ers, and/or a time-sharing service. They would like to be able to distribute new software to all of the small computers, quickly and easily. They would like to provide a central discipline and control over these small computers, to discourage users from developing incompatible data and programs.

Local computer networks have real benefits to offer for overcoming these common operating problems, as we have indicated.

Integrated information handling. But in addition to using local computer networks to solve common operating problems, users see them as important components of future systems. Factory automation computers need to be tied into companies' information handling networks. The same is true of laboratory data collection and processing. Office automation is arriving; the many work-stations need to be tied into an overall company network.

Local computer networks offer a way to tie a company's many information handling functions into a cohesive whole.

Why not conventional data communications? You might ask: Why is something new needed? Why not solve these problems with conventional data communications? Well, conventional data communications (with modems, telephone lines, etc.) are generally much slower in transmission speed, are not as flexible for attaching new technology, are more expensive if common carrier circuits are used, and have higher error rates due to the analog technology that is used.

Direct cabling of the units is not the answer either. The cables can be bulky, cumbersome to move, hard for operators to connect and disconnect when switching units, and expensive. And, as mentioned, the maximum cable lengths are quite short for high speed transmission.

All in all, it looks to us as though local computer networks have important benefits to offer.

Components of a network

While there are numerous elements that make up a local computer network, we will single out a few disparate ones for comment.

The transmission medium. The trunk (or bus) can be active or passive, can have a line structure or a loop structure, can be made of coaxial cable, optical fibers, or even a twisted pair of wires. If the trunk is active, it has elements in it that amplify the signals; the obvious advantages of regeneration can be offset because if one of those elements fails, the trunk may become inoperative.

The line structure is simply a continuous piece of trunk medium, such as coaxial cable, that may or may not allow branching but which does not connect back on itself. The loop or ring structure, on the other hand, does connect back to itself. Both types of structures have been used successfully.

Whatever the type of trunk, all signals are 'broadcast' onto the trunk and are received by all stations connected to the trunk. However, only the addressed destination station copies the message addressed to it; the other stations ignore the message. Most networks do allow true broadcast messages, which are copied by *all* stations.

Access method. There seems to be even more ways to access a trunk than there are types of trunks. The most common method employed so far is the contention method, we gather. A station that wishes to transmit must first check to see if a transmission is in progress. If so, it waits; if not, it begins to transmit. If two stations begin to transmit simultaneously, means are needed to abort the transmissions.

Metcalfe and Boggs, in Reference 3, give a good discussion of contention access. Kanakia and Thomasian, in Reference 1, give a good coverage of various ways of accessing a ring network. These ways include both synchronous and asynchronous

time division multiplexing, contention, dynamic assignments by demands made on a central ring arbitrator, and priority assignments.

Interface and taps. Television technology provides effective taps by which stations can tap onto a coaxial cable trunk. Tapping onto optical fiber circuits probably is more of a problem but—with all of the attention being given to it—this should be solved when optical fiber trunks are offered in the market place, we would think.

The interface units that are needed pose more of a problem. They have to physically match the unit (CPU or controller) to the trunk, provide some buffer storage, convert bytes from parallel to serial bit streams and vice versa, perform the accessing of the trunk, etc. Further, they should be designed such that, if an interface unit goes down, only its station is affected; the rest of the network should continue to operate.

Software. Ah, here is the rub—and you knew it was coming. It is always the software that is the main problem.

Local computer networks on the market provide for physical communication with the network and the lowest level protocols for using the network. But these are not sufficient. In order to use the network, some enhancements are needed to today's operating systems for the handling of remote resources. We are indebted to James White of Xerox, J. E. Donnelley of Lawrence Livermore Laboratory, and the people at Network Systems Corporation for ideas on what these enhancements will be. Following is our understanding of their ideas.

Local networking is so new that few of today's operating systems have either been designed or modified to function in a local network environment. Generally, the operating systems deal with *local* resources—such as programs, data, and terminals. Some, of course, do handle remote resources, as in the case of time-sharing operating systems that handle remote terminals, and remote batch systems that handle remote batch terminals and file transfers. Note, too, that time-sharing may be provided by an addition to a conventional operating system, as in the case of IBM's TSO.

In local computer networks, the localness or remote-ness of a resource should be invisible to the user and the user's software. The operating system will have to be modified to provide for handling remote resouces almost as though they were local. This action will result in *network operating systems*. Here are some features one will expect to find in these new operating systems.

Operating systems typically provide *device drivers* for each class of supported device—such as hard disks, line printers, terminals, etc. These device drivers are the lowest level of software that interface with each particular class of hardware. For local computer networks, *network device drivers* will be needed for interfacing the operating system to the network hardware.

Some examples of the functions to be performed by the software at this level are the following: (1) transmission set-up, to make sure that the receiving station is ready to receive, (2) flow control, to keep track of how full a receiving buffer is and to turn the flow of data on and off, as required, (3) packet control, to make sure that packets have been received in proper sequence and are acknowledged, and (4) end-to-end error checking. And there are other functions to be performed at this level.

Most of today's operating systems provide inter-process communication, for processes *within* a host—such as passing data between two user programs. For network use, the operating system must also provide host-to-host *inter-process communication*. The host processors can be main frames, mini-computers, work-stations, or almost anything with processor capability. This inter-process communcation capability is the most fundamental network ser-

vice, upon which the other services are built. As an example, a user program in one host must be able to call on, or pass data to, a program in another host.

Still another necessary function for network service is *file transfer.* Ideally, perhaps, a program should be able to treat remote files as if they were local. Frequently, though, file transfer will be used to bring the remote file to the local environment for processing.

Another necessary function, of course, is the ability to handle remote terminals as if they were local, similar to what is done in today's time-sharing systems. The operating system might be enhanced to provide this capability, or a time-sharing operating system might be used.

A number of higher level protocols are needed for these types of services. These include sophisticated error checking, network monitoring, network diagnostics, and even dynamic network re-configuration when an inoperable unit is detected.

These operating system enhancements will be for the purpose of attaching *host processors* to the network. The subject becomes even more complicated when one considers attaching peripheral units directly to the network (via their controllers), in order to share them among a number of hosts. We will have more to say about this additional complication shortly.

This, then, is our understanding of some of the characteristics of the forthcoming network operating systems. As the use of local computer networks grows, the needed operating system enhancements will become commercially available. In the meantime, users may have to develop at least some of these enhancements on their own.

These are still the early days of local computer networks. But it looks to us as though interest in, and use of, these networks will grow dramatically in the next few years. We would expect to see substantial improvements in network operating systems in the not-distant future.

What users are doing

Based on our interviews and research for this issue, here is our understanding of what local computer network users are currently doing with their networks. And, almost as important, we will indicate some significant uses that they are not yet attempting.

Big system environment. One main use of a local computer network is to provide high speed communications between two CPUs, which can be either compatible or non-compatible. In this situation, each CPU has its own peripherals attached to it. Communication is handled mainly by services already available in the operating systems.

Another use in this environment is to provide communications between a CPU and a set of non-compatible peripherals. An example of this might be the use of some IBM peripherals by some other brand of computer—say, a Control Data Cyber.

Still another use is for providing communications among multiple non-compatible CPUs, each with its own peripherals.

Small system environment. The typical network here seems to be one that has multiple small computers connected to it, and where each computer has its own peripherals connected to it—say, floppy disks and a character printer.

One use of such a network is to allow each of the small computers to act as a terminal for a larger time-sharing computer that is connected to the network. In fact, the time-sharing computer can be at a remote site, as long as its network is interfaced with the local computer network.

Another use is to allow these small computers to use important resources, such as large volume disk storage and/or high speed printers. These resources, at this point in time, are usually connected to a larger CPU which in turn is connected to the network; the resources do not connect (via their controllers) to the network.

Local networks for small systems are also being used to inter-connect office work-stations, for performing automated office functions. These functions include message services, conferencing, management report distribution, and so on.

What users aren't doing. With all the work that is going on at many locations, in the use of local computer networks, it is hard to say that something is not being done. However, based on discussions with a number of people working in the field, we think that the following two significant uses of local computer networks are not yet being realized. *Automated office tests* are not being conducted with the local networks that tie high speed computers together. We expected to at least hear of plans for connecting office work-stations to such nets. Also, we came across no users who are *directly attaching peripherals* (via their controllers) to the networks, in order to share them among multiple CPUs.

This latter use (direct attachment of peripherals) is one of the sales points made for local networks. The difficulty, we gather, lies in today's operating systems. Let's look at this in a bit more detail.

Network access of peripherals

J. E. Donnelley (in Reference 1) points up two dichotomy problems that arise when trying to use today's operating systems in a local computer network environment.

Dual access dichotomy. This problem arises at the CPU in which a user's program resides, when that program calls for a resource—say, data stored in a disk file. If the disk unit is local (attached to that CPU), the file is accessed by means of an operating system disk call. But if the disk unit is remote, the file must be accessed by a network communication call. So the user's program must know if this data resource is local or remote, and must handle the two cases differently.

Dual service dichotomy. This problem is the other side of the coin—making a resource available to remote users via the network. Typically, the local computer services requests that are received over the network by means of 'server' processes, says Donnelley. These servers receive the requests and turn them into local requests to the operating system. The results go to the servers which then send them to the requestors via the network.

In this situation, two service codes are needed for each resource, says Donnelley. One service code is needed in the operating system, for handling local service. The other is in the server processes, for handling the network requests.

In a discussion, Donnelley pointed up another problem with today's operating systems that inhibits the attachment of *peripherals* to the network, so as to share them among multiple CPUs. For instance, to share a data storage resource (disks or tapes) among several CPUs, not only is physical access to the resource needed (which the local computer network provides) but also *logical* access is needed. That is, the CPU that wishes to use the storage must have access to the tables that define the storage area and how it is allocated. These tables today are stored within one operating system. If it is another CPU that is accessing the storage, it must make its request through the operating system that has the tables.

What is really needed, according to Donnelley, to solve the dichotomy problems is an operating system design that treats both local and remote requests in the same way. That is, it handles only one type of call, and that one type is a message. The operating system just passes the message to the routine that will provide the requested service. Local and remote requests would have identical types of messages; they would differ only in their origin or destination.

But there are problems with this concept. One main one is: how to keep the overhead in this type of system at least as low as the overhead that is found in typi-

cal third generation operating systems? Two authors cited by Donnelley—Fletcher and Watson—point out that reliable message passing requires end-to-end protocols, as in a packet switched network. The most common approach to message passing today is the virtual circuit approach. This approach, in turn, can require the exchange of *five or more* messages in order to reliably send *one* message. Two messages are needed to set up the logical connection, then the message is sent, then an acknowledgment may be returned, and then two more messages are needed to close the connection. Virtual circuit systems, of course, do not typically send just one message at a time; if several messages are sent after the connection has been set up, then the overhead is spread over these several messages. But if only one message is sent, the overhead may be too high.

What is being sought, then, is an end-to-end protocol for reliably and efficiently passing single messages.

Donnelley's own organization—Lawrence Livermore Laboratories, where they have at least one each of every super-computer—is working on the development of such an operating system. This project is being done in connection with a new in-house time-sharing system they are developing. And they are finding the project very challenging, Donnelley reports. If and when they are successful, their design should pave the way for future operating systems that work efficiently in a local computer network environment.

As for the question of attaching peripherals directly to the network (via their controllers), in order to share them among multiple CPUs, Donnelley sees no easy solution. His organization is considering doing a limited version of this in the next year or two. They are thinking of putting some public files—which any user program can access and which are read-only or execute-only files—on a disk unit that attaches to HYPERchannel. Even this limited approach will require changes to the operating systems, says Donnelley.

Somehow the tables—that define the allocation of the storage space for these public files—will have to be shared among several CPUs. The manner of this sharing must be found, along with the related question of updating all copies reliably.

These problems of network access to peripherals will be solved in time, we suspect, and possibly along the lines that Donnelley describes. In the meantime, local computer networks can be put to good use even without this facility.

What does this mean for you?

The way it looks to us, local computer networks are coming, and coming rapidly. Most data processing management might well include some aspect of these networks in their longer range plans.

For big computers. There are already a good number of installations of local networks for connecting large, fast computers. Most of the installed networks have used HYPERchannel components, but other suppliers are entering this field.

These networks are being used for load balancing among multiple CPUs, for adding new technology easily, and for opening up the market place to more suppliers.

For small computers. We gather that there are not, as yet, as many local networks for small computers as there are for the larger ones. But we suspect that the small computer networks will far outnumber the others not too many years in the future.

These networks are being used to tie small computers and work-stations together for a number of reasons. One reason is to provide access to important resources, such as large volume data storage, or high speed printers, or a time-sharing computer. Employees can exchange messages easily among themselves via such a network. New software can be distributed via the network. And a measure of discipline and control can be exercised over the

users by requiring that their programs meet network standards.

It seems clear, too, that local networks of small computers will be used in distributed systems, in automated offices, and in tying factory and laboratory information systems into corporate information systems.

As we say, local computer networks are coming. Now is a good time to factor them into your longer range plans.

REFERENCES

1. *Proceedings of 4th Conference on Local Computer Networks*, October 1979, No. 276; order from IEEE Computer Society (5855 Naples Plaza, Suite 301, Long Beach, Calif. 90803), price $16.00.
2. Some copies of the proceedings of three previous conferences on local computer networks, held in Minneapolis, Minn., are available. Contact University Computer Center, University of Minnesota, Minneapolis, Minn. 55455.
3. Metcalfe, R. M. and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks," *Communications of the ACM* (1133 Avenue of the Americas, New York, N.Y. 10036); July 1976, p. 395-404; price $5 prepaid.
4. The U. S. National Bureau of Standards has held two symposia on local area networks, one in August 1977 and one in May 1979. Proceedings of the first one are available from the U.S. Government Printing Office, Washington, D. C. 20402; Stock No. 003-003-01918-6; price $2.40. For information on how to obtain a copy of the proceedings of the second symposium (which was held in May 1979 and which was co-sponsored with the Mitre Corporation), write U.S. National Bureau of Standards, ICST, Room B226, Building 225, Washington, D. C. 20234.
5. Mr. James E. Thornton, President of Network Systems Corporation, has delivered a number of conference papers on local computer networking. For copies of such papers, plus information on HYPERchannel, write Network Systems Corporation, 7600 Boone Avenue North, Brooklyn Park, Minn. 55428.

SUBJECTS COVERED BY EDP ANALYZER IN PRIOR YEARS

1977 (Volume 15)

Number

1. The Arrival of Common Systems
2. Word Processing: Part 1
3. Word Processing: Part 2
4. Computer Message Systems
5. Computer Services for Small Sites
6. The Importance of EDP Audit and Control
7. Getting the Requirements Right
8. Managing Staff Retention and Turnover
9. Making Use of Remote Computing Services
10. The Impact of Corporate EFT
11. Using Some New Programming Techniques
12. Progress in Project Management

1978 (Volume 16)

Number

1. Installing a Data Dictionary
2. Progress in Software Engineering: Part 1
3. Progress in Software Engineering: Part 2
4. The Debate on Trans-border Data Flows
5. Planning for DBMS Conversions
6. "Personal" Computers in Business
7. Planning to Use Public Packet Networks
8. The Challenges of Distributed Systems
9. The Automated Office: Part 1
10. The Automated Office: Part 2
11. Get Ready for Major Changes
12. Data Encryption: Is It for You?

1979 (Volume 17)

Number

1. The Analysis of User Needs
2. The Production of Better Software
3. Program Design Techniques
4. How to Prepare for the Coming Changes
5. Computer Support for Managers
6. What Information Do Managers Need?
7. The Security of Managers' Information
8. Tools for Building an EIS
9. How to Use Advanced Technology
10. Programming Work-Stations
11. Stand-alone Programming Work-Stations
12. Progress Toward System Integrity

1980 (Volume 18)

Number

1. Managing the Computer Workload
2. How Companies are Preparing for Change
3. Introducing Advanced Technology
4. Risk Assessment for Distributed Systems
5. An Update on Corporate EFT
6. In Your Future: Local Computer Networks

(List of subjects prior to 1977 sent upon request)

PRICE SCHEDULE (all prices in U.S. dollars)

	U.S., Canada, Mexico (surface delivery)	Other countries (via air mail)
Subscriptions (see notes 1,2,4,5)		
1 year	$48	$60
2 years	88	112
3 years	120	156
Back issues (see notes 1,2,3,5,)		
First copy	$6	$7
Additional copies	5	6
Binders, each (see notes 2,5,6)	$6.25	$9.75
(in California	6.63, including tax)	

NOTES

1. Reduced prices are in effect for multiple copy subscriptions and for larger quantities of a back issue. Write for details.
2. Subscription agency orders are limited to single copy subscriptions for one-, two-, and three-years only.
3. Because of the continuing demand for back issues, all previous reports are available. All back issues, at above prices, are sent air mail.
4. Optional air mail delivery is available for Canada and Mexico.
5. We strongly recommend AIR MAIL delivery to "other countries" of the world, and have included the added cost in these prices.
6. The attractive binders, for holding 12 issues of EDP ANALYZER, require no punching or special equipment.

Send your order and check to:
EDP ANALYZER
Subscription Office
925 Anza Avenue
Vista, California 92083
Phone: (714) 724-3233

Send editorial correspondence to:
EDP ANALYZER
Editorial Office
925 Anza Avenue
Vista, California 92083
Phone: (714) 724-5900

Name_____

Company _____

Address _____

City, State, ZIP Code_____