WIS-CS-177-73
The University of Wisconsin
Computer Sciences Department
1210 West Dayton Street
Madison, Wisconsin 53706

Received March 20, 1973

A SURVEY REPORT ON COMPUTER NETWORKS

by

Michael Schneider

Technical Report #177

March 1973

TABLE OF CONTENTS

ABSTRACT

This paper first defines what is meant by a computer network and then looks at the useful user services that can be provided through such a network. It then reviews the design alternatives, both logical and physical, that are available to a network researcher. Six actual networks--ARPA, CANUNET, MERIT, CYBERNET, OCTOPUS, and DCS--are briefly discussed and compared. Finally, the most current research problems are outlined and some possible approaches taken by the designers of the above networks are contrasted.

1. INTRODUCTION

Computer Network research has only recently been defined as a separate area of Computer Science research that overlaps such other diverse areas as Operating Systems Design, Programming Languages, Hardware Design, Electrical Engineering, and Communication Theory. The tremendous interest shown recently in network development has come about because of the (not so profound) realization that a network of computers could provide a user community with far greater local computing power than a single machine at, hopefully, a reasonable price.

It is natural to think of a computer network as "...two or more computers communicating meaningful information." (BELL69) However, for the needs of this paper that is too broad a definition and admits too wide a range of structures--both from the point of view of the nature of the interconnection and in terms of the type of information communicated. A much better definition is the one suggested by Roberts and Wessler in (ROBE70) which defines a computer network as a "set of two or more autonomous, independent computer systems interconnected so as to permit interactive resource sharing between any pair of systems." By autonomous and independent we will mean any computer capable of performing "stand-alone" processing when separated from any other available resources in the network. This eliminates from consideration as a network a configuration of two or more central processors sharing a common primary memory--commonly termed "multi-processing." (However many interesting examples of multi-processor networks are being developed. See, for example, (BELL72).) By requiring the connection to be between computer systems we eliminate those systems whose nodes are only terminals capable of transmitting or receiving information but

not of processing it locally (e.g. remote batch or remote job entry terminals.) By interactive resource sharing we will mean that any node in the network should theoretically be able to access the resources (hardware, software, data) of any other node just as if it were a local resource.

This paper will examine both the immediate and long-term goals that networking hopes to achieve. It will take a look at the different design philosophies that have been used in creating existing or planned networks, and will briefly survey six actual computer networks that have been implemented or are in the design stage. It will discuss the more currently pressing problems facing the network designer and will look at how the creators of some of the six surveyed networks either solved or avoided these problems. Finally it will conclude with a discussion of what directions future research will probably take, what needs to be done, and what can be expected in the future.

2. SERVICES PROVIDED BY NETWORKING

The particular services that a network designer hopes to provide obviously depend both on his particular design and on the specific needs and desires of the community of users that will be accessing the network. However, the primary goals of networking can be generally classified into six areas.

2.1 Load Levelling

By load levelling we mean the transmission of both a program and its associated data base to another node in the network for the purpose of equalizing the computational load between machines (CANU72-Appendix K). In just about any network with a fairly large number of nodes and a wide geographical base the work load between nodes will probably be in imbalance. This might be due to a particular node being in an area with a much larger user population. It is possible that one node might have an extremely desirable resource that generates an inordinate amount of traffic. Such is the case of the ILLIAC-IV node of the ARPA network. Or it might be the case that because of time zone differences one node of a network might be running fully loaded during a prime-time shift while one node is virtually empty because it is late evening. This might be the case, for instance, in the proposed CANUNET network which, in the process of spanning the width of Canada, will cover five time zones.

However even after mentioning the frequency with which imbalance occurs and realizing the obvious advantages to be gained through load levelling, it turns out not to be a major goal in the majority of network designs. The sharing of computational load implies homogeneous computers since the problems involved in load

levelling between heterogeneous machines are enormous and presently not worth the effort. This immediately eliminates most networks from consideration since most networks are heterogeneous they allow a wide range of different and generally incompatible machines. Even in a homogeneous network there are still vast problems involved. How does a computer dynamically determine if another computer and certain specific resources are available? Even if they are available how can that system determine if the load at the distant node is actually lighter than the load locally? We must also consider that even if we can transmit our job to a lightly loaded node might the cost of the transmission there and back end up being more costly than just waiting and processing locally? Finally, what do we do about the differing command languages and options available on different systems? We will need some type of translator to arbitrate these differences or impose some kind of universal command structure on the network. Because of these and other problems load levelling is considered a minor part of network design and, in fact, only two networks (CYBERNET,DCS) of those discussed in section 6 consider this service to be an integral goal of their design. (FARB72a,ROBE67)

2.2 Program Sharing

Program sharing is the term used to mean the transmission of a data base, under user control, to a remote node where it will be operated on by a resident program with the results returned back to the source (FARB72a,ROBE67). This is a very desirable service to provide a user because it allows him to make use of already existing software. It can create tremendous savings by reducing

the amount of programming effort and reducing the amount of duplication of work in getting a job "on the air". It is a feature that is provided by every major network. It is not "automatic" though, and in heterogeneous networks it requires a careful and usually difficult design problem of conversion between differing modes of storage, differing codes (e.g. ASCII, EBCDEC, Fieldata), and filing conventions used on differing machines. Depending on the complexity this conversion could be carried out in an "ad hoc" fashion by the transmitting node, the communications interface, or the receiving node. A more generalized solution is to have all information conversion handled by a completely separate and centralized Data Reconfiguration Service (Fig. 1) through which all data exchanges pass. The Data Reconfiguration Service would then be responsible for conversion of the information to a format acceptable to the program at the receiving node. Examples of this approach can be seen in (ANDE71, SCHA70).

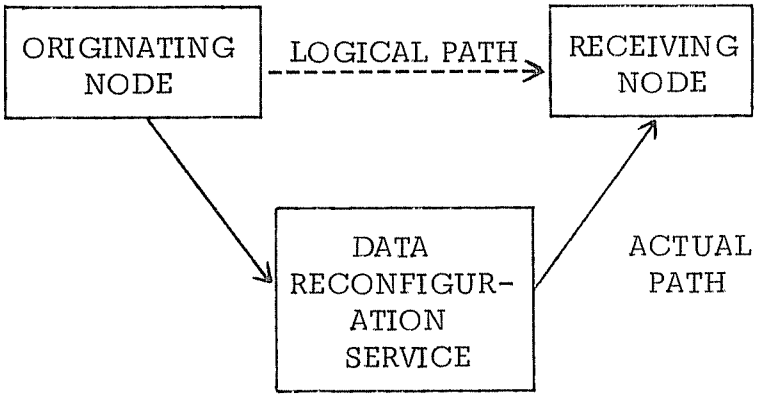


Figure 1. Data Reconfiguration Service

2.3 Data Sharing

This term usually refers to the service in which a program is transmitted under user control to a node containing a specialized data base upon which the program will operate (FARB72a, ROBE67, CANU72-Appendix A). Again, as in program sharing, this is a very useful service to provide to a user. If one node of a network has a very large data base (e.g. 10^9 bits) then it is usually much more economical to send the program to the data than to do the reverse. The existence of these large, specialized data bases is quite common. During the design phase of the CANUNET network literally dozens of large data bases were identified as being important enough to offer to the national data processing community on a network basis (CANU72-Appendix K). Virtually every network offers this service. Again, however, we must realize the design implications of providing this type of service. The sharing of object code is totally unrealistic except between identical machines. The sharing implied in this service usually implies programs on the source language level where the receiving node performs a separate compilation to its own machine language. However even such "standardized" languages as FØRTRAN and CØBOL contain enormous syntactic differences between manufacturers and we must realize that frequently these source programs do not "travel well". The main incompatabilities are (ELIE70)

a. Differing Vocabulary -- The collection of source statements recognized by one compiler may differ from one installation

to the next. This might be due to lack of standards or particular manufacturer's features (e.g. Burroughs ALGØL).

b. As the semantics of most languages are not formally defined the interpretation of the same statement might vary between installations.

c. Some features of a compiler might be machine dependent such as the storage of multi-dimensional arrays.

d. The handling of alphanumeric information and the way it is stored internally and packed/unpacked is not standardized.

e. Job control statements will vary between installations.

To resolve these differences we must either require some human re-programming effort (unacceptable in a real-time environment) or create some kind of network "filter" that will recognize and correct these differences on an "ad hoc" basis. Hopefully the industry-wide standardization of languages will help to alleviate this problem somewhat.

2.4 Dynamic File Access

This feature means the ability of a program to access a remote data set as if it were local. This allows a program to operate on a distributed data base without any special planning.

2.5 Remote Job Initiation

In this class of service both the program and the data reside on a remote node and the local node is used solely to initiate a job step and possibly transmit a small amount of parametric information. Basically this type of service is equivalent to the terminal/central machine relationship in a time-sharing environment. This service

would allow a computer in the network to look like a simple terminal to another computer for the purpose of job initiation or query.

2.6 Hardware Sharing

Most of the above features have stressed the sharing of software, either programs or data bases. However, as common as software sharing is the case where the resource to be shared is a hardware device that might not be available locally. Examples of this might be an I/O peripheral (plotter, high-speed line printer), an unusually large core memory, or a unique hardware resource not available anywhere else (e.g. the parallel processing abilities of ILLIAC-IV).

As can be seen in all of the above services, the common thread in all these goals is the idea of sharing. Each computer is trying to make every one of its local resources available to any other user or computer throughout the network. This idea of resource sharing will hopefully have a beneficial side effect--the fostering and growth of a way of thinking that unfortunately is not widespread presently, the concept of a "community of users". The idea here is to investigate what has already been done by those belonging to your network and to eliminate expensive duplication of effort or facilities by utilizing these efforts through interactive resource sharing. This cooperation between users in making known what is currently available and how it can be utilized should greatly increase the productivity of all users in the network by making a wide range of resources immediately available. The end result will then be the simple but powerful idea of "using what has already been done."

3. NETWORK DESIGN OBJECTIVES

When building a large system, whether it be a supervisor, data base manager, or computer network, too often the objectives of the system designer are given top priority. While the designer's goals and wishes must certainly be considered important in the overall design schema, the interests of the potential user must always be of the prime importance, for in the long run, he is the individual who will either make or break the efforts of the designer. The objectives of a network designer then must be to attempt to create a network which will allow us to achieve the services described in Section 2 but do it in an environment that is attractive enough to potential users to make the network actually utilized. Some of the important design objectives and "human engineering" problems in network design are discussed in this section.

3.1 Ease of Use

A user should find the network resources no more difficult to use than, let's say, a local peripheral. The command language should be easy to learn and manipulate; it should be possible to do "simple things simply". Also the network and its multiple layers of software and hardware should be as transparent as possible to the user. The network design should not impose any restrictions on the allowable patterns of information that can be transmitted through the network. For example, if the user wants to transmit raw binary data between nodes then the network design must allow for some kind of "transparent" mode of transmission so that certain patterns of data bits are not mistaken for control commands (e.g. EØT--End of Transmission) with disastrous side effects (EISE67). The network

designer should create some kind of HELP mode to allow a user who is either unfamiliar with the network or just plain lost to use the system itself to recover quickly and easily.

3.2 Reliability

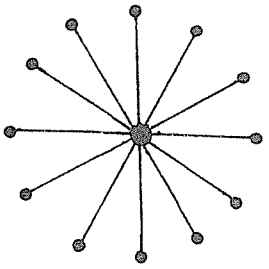
Network reliability is basically a measure of the difficulty in isolating a working node from the rest of the network. There should be at least two or more physical communication paths to every node so that a failure of one path will still leave that node with access to the rest of the network.

The reliability of a network is primarily a function of its topology and the probability of failure of its individual links and nodes. Given a value for the probability of a node failure, p, and the probability of a line failure, q, a network designer can use either classical analysis or simulation to begin to get an estimate of the reliability R of a proposed network design. An interesting example of this, shown in (HANS71), is the determination of the reliability of various ways of connecting twelve remote nodes to a centralized facility. In this case reliability is defined as the probability that a working node will be able to communicate with the central computer.

In the ARPA Network the design constraint on R is that it be approximately 1.00. That is, if an IMP is operable then it should always be able to communicate with any other operable IMP in the network (ROBE70,FRAN72). This is achieved by lowering the possibility of an IMP failure by physically "ruggedizing" the hardware as much as possible and by providing a sophisticated topology which allows for multiple physical paths between nodes (HEAR70).

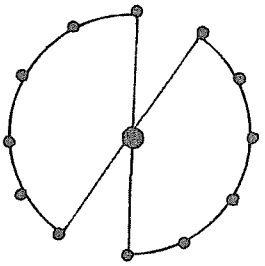
probability of a node failure = 0.01
probability of a line failure = 0.10

a)



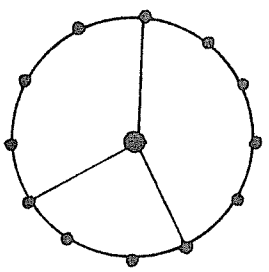
Star Configuration
R = 0.88

b)



Loop Configuration
R = 0.91

c)



Spoke Configuration
R = 0.94

Figure 2. Reliability Values for Three Different Node Networks

The cut-sets of a node, the set of links which must be broken to isolate a node from the network, all contain at least two elements.

3.3 Error Rate Reduction

Because of the noise on communication lines, especially impulse (or white) noise, there will always be the possibility of frequent errors in data transmission along communication lines. Many measurements have been made of the frequency of transmission errors on standard sub-voice grade and voice grade communication lines to determine approximate error rates. The general rule of thumb figures are (MART70):

Type of Channel	Baud Rate	Error Rate (Approximately)
Telex Sub-Voice Grade	50	1 in 50,000
Public Sub-Voice Grade	150	1 in 100,000
Public Voice Grade	1200	1 in 200,000
Publice Voice Grade	2400	1 in 100,000

Figure 3. Approximate Error Rates on Public Communication Channels

An error rate of 1 in 10^5 cannot be ignored in network design and still be acceptable to most users. At this rate if a user were transmitting continuous streams of bits at 1200 baud about 43 bits would be erroneously transmitted in one hour's time. However, using such simple checks as vertical and horizontal odd-even parity can reduce the undetected error rate to about 1 error in 10^8. At 1200 baud this would be about one undetected error per day. The user pays for this, however, with a reduction in throughput and

line utilization due to the additional characters needed to do the checking. Using horizontal and vertical parity checking and 7-bit ASCII the percent degradation is:

$$\text{Degradation} = 100 * \frac{N+7}{8*N} \quad \text{where} \quad N = \text{number of chars/block}$$

With $N = 100$ this amounts to over 13% of all the information transmitted. Using even more sophisticated error checking procedures such as cyclic and spiral error checking characters or polynomial codes the undetected error rate can be lowered to as little as 1 in 10^{13}. This would correspond to about one error per century. A good description of the use of these codes and the undetected error rates achievable is given in (MART70).

The network designer must decide the acceptable level of undetected errors in the network that represents a reasonable compromise between the desire for high reliability and maximum line utilization. Then he must design an error detection scheme that achieves that level. In CANUNET the acceptable level of undetected error was set at 1 bit/year in the entire network (CANU72). In ARPA the error rate of about 1 in 10^{12} was decided upon as an achievable goal (FRAN70).

Another aspect of error detection that should be mentioned briefly is error correction. Automatic error correction codes are usually prohibitive because of the tremendous amount of redundancy required. A much simpler solution is the automatic retransmission of erroneous information. This, however, implies that the transmitting node must store the information until it is sure that the message has been correctly received. It also implies that some kind of positive/

negative acknowledgment procedures will be required. All of these things must be anticipated by the network designer and considered in the overall design.

3.4 Responsiveness

Just as the time-sharing user begins to get "itchy" if response times are extremely slow, so will the network user. The network designer must impose a maximum response time constraint on his network design. In the ARPA network a primary design constraint was that the propagation time (transmission and acknowledgment) of any packet in the network be $\leq$ 0.2 sec. (FRAN72). This would allow approximately 1-1$\frac{1}{2}$ sec. for processing the message at the receiving node and still have a total response time under two seconds--acceptable in a conversational mode. The design should also be flexible enough to achieve this response time under wide variations of traffic load without significant variations.

3.5 Capacity

The network must be allowed to expand in a simple way with the minimum of impact on already existing nodes. Ideally, the creation of a new node should be as simple as putting a plug in a socket, once the hardware is purchased. It should not require a reprogramming of software already in the network, aside from possibly making a few new entries in some system tables.

3.6 Cost

Probably the single greatest challenge facing a network designer is to make the network economically attractive to the users. A primary point is that "...although the network may appear economically

attractive on a global or national basis unless it appears economically attractive to the individual user, the network will not be used." (CANU72-Appendix N) The cost of becoming a node in a network must not be so high that a user thinks the money could be better spent in purchasing other equipment or upgrading local facilities. The costs to be considered include both the initial purchase of any necessary hardware and the line charges amortized over all the nodes.

The CANUNET planning group has approximated that the cost of maintaining a node in the CANUNET network, including both line costs and staff, at between $10,000 and $15,000 per node/month (CANU72). The cost of an ARPA IMP or TIP (a Honeywell DDP-516 computer) is about $100,000. A great deal of network research is going on in the area of cost reduction and in bringing network services to the small user with limited resources and funding (DESA73).

4. NETWORK TOPOLOGY

4.1 The Individual Nodes

This section is concerned with describing the various ways that a network can connect together its individual nodes. Regardless of the overall design philosophy used to connect the nodes of the network together, the nodes themselves will always be quite similar in their makeup. A network node will consist of three distinct units: (Figure 4).

4.1.1 The HOST

The nodal HOST is that device located at the node which either contains the resources that will be shared in the network and/or allows a user to access the available resources in the network. This unit is called variously the "Grid Node" in the IBM Network/440, "HOST" in ARPA, CANUNET, and MERIT and "Centroid" in CYBERNET. For this paper we will use the term HOST. Depending on the network design the HOST may consist of either a single computer (Figure 4a) or possibly many (4a, 4b). A HOST might consist of a single node from an entirely separate network (4d). This latter case would allow the creation of a "network of networks". The network design may also possibly allow the direct or indirect connection of devices which in themselves have no resources to contribute to the network resource pool but only allow users access to the other facilities of the network--i.e. a terminal. This is illustrated in Figure 4c.

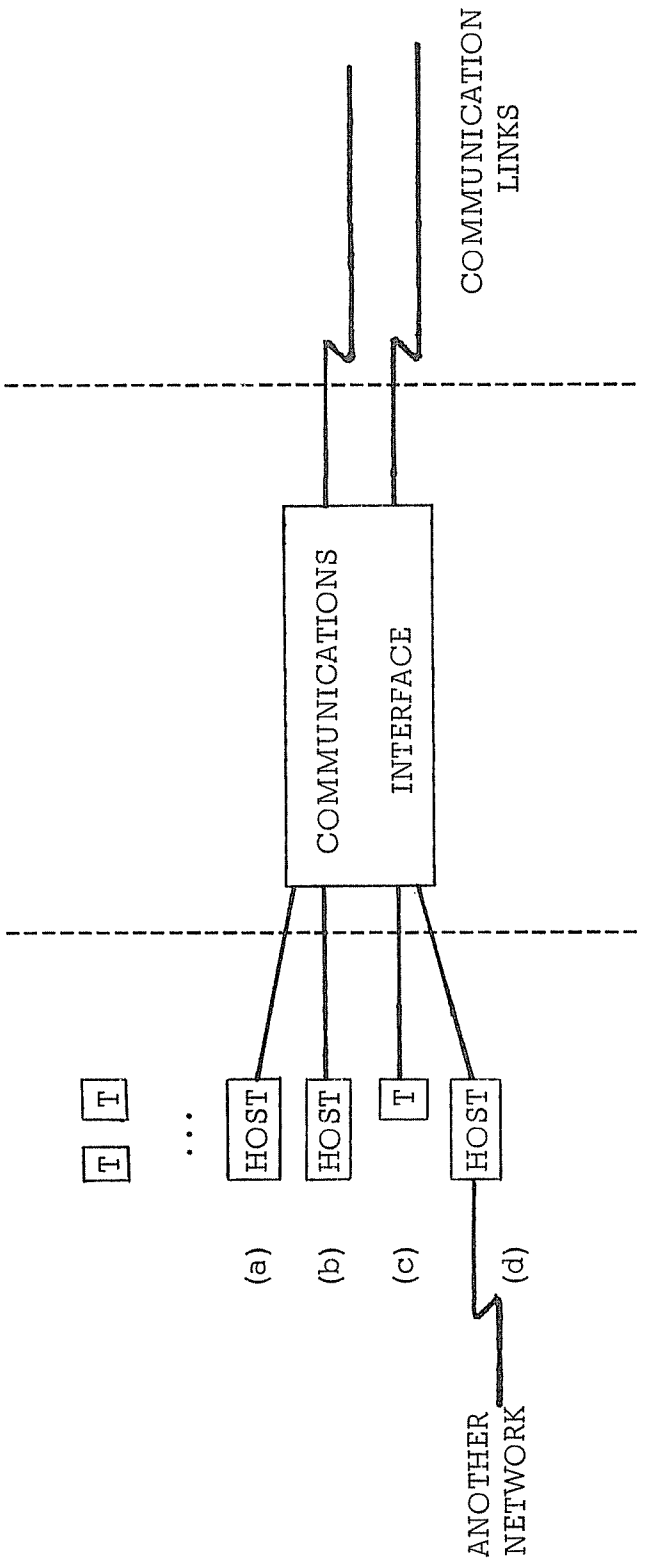


Figure 4. A Typical Network Node

4.1.2 The Communication Interface (CI)

Typically the nodal HOST contains none of the software or hardware facilities required for handling the overall network communications responsibilities. All the communications control lies in a Communications Interface. This device is variously titled the IMP in the ARPA Network, the Communications Computer (CC) in the MERIT Network, the Communications Subsystem (CS) in Network/440, the Node Computer in CANUNET or the Ring Interface (RI) in DCS. In this paper we will refer to it as just the Communications Interface or abbreviate it as CI. Depending on the network design it may be incorporated into the HOST operating system (e.g. Network/440) or be a physically separate unit performing its jobs independently of its associated HOST (e.g. ARPA).

Regardless of where it is actually located the Communications Interface must perform five basic functions:

a. Data Signal Conversion--The CI performs the signal conversion functions such as modulation/demodulation or serialization/deserialization.

b. Message Handling and Buffering--The CI is responsible for assembling and disassembling the arriving messages into packets for transmission through the network. It must handle the storage of these packets, if necessary, until the receipt of a successful arrival.

c. Error Control--The CI is responsible for accumulating any error checking characters required by the message transmission protocol. It is also required to check the

corresponding characters on the incoming messages for the detection of errors in transmission. It is responsible for the negative acknowledgment of erroneous messages. If automatic retransmission is required the CI must handle it. It must also be checking for other possible errors such as duplicate messages, messages received out of sequence, and infinitely looping messages or packets.

d) Flow Control--A network in which messages may easily enter and exit can become locally congested. It can even come to a complete halt under worst case conditions. It is the responsibility of the CI to prevent these situations from occurring and recovering from them if they do occur.

e) Routing--It is (usually) the responsibility of the CI to determine the best path to the destination from among all the possible paths.

4.1.3 The Communication Links

The communication links connecting the nodes of a network can be either private lines (usually coaxial cable or microwave), or more commonly, leased from the common carrier communications facilities (telephone, telegraph). The links may be dedicated, point-to-point lines in which case there exists a permanent path between two nodes. Alternatively the links may be circuit-switched, dial-up links in which case the connection between two nodes is made through a switching center and lasts only for the duration of that call. The difference between these modes is illustrated in Figure 5:

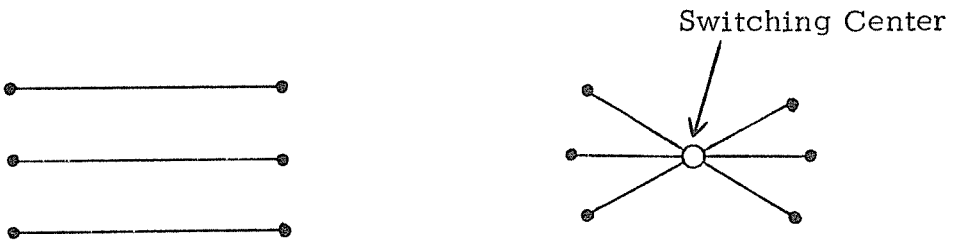


Figure 5. Private vs. Switched

Dedicated lines are more expensive but guarantee the existence of a path when needed. Dial-up lines raise the possibility of a "busy-signal" as well as causing a delay in making the physical connection.

There is also an extremely wide variation in speeds and capacities of lines available through the common carriers. Some examples of what is available and the general categories of lines is shown in Figure 6 (MART70).

Class of Line	Speed Range	Examples
Subvoice Grade	10-300 baud	AT&T Type 1002 (55 baud) AT&T Type 1006 (150) Datel 200
Voice Grade	300-9600	AT&T Type 3002 (1200) AT&T Type 3002 with C4 conditioning (4800)
Wideband	10,000- ?	AT&T Type 8802 (19.2Kb) AT&T Telpak C or Western Union Telpak C (105 Kb)

Figure 6. Categories of Communication Links Available from Common Carriers

The link may be designed to propagate messages in both directions simultaneously (full duplex), only one direction (simplex), or alternately in either one direction or the other (half duplex). A thorough discussion of communication lines can be found in either (MART70) or (MART71).

4.2 High Level Network Design

The set of physical communication paths connecting two nodes together form a logical link or channel. The high-level network design is concerned with the problem of creating channels between all nodes in the network. What is the best way to link up geographically diverse nodes which minimizes line cost and maximizes line utilization and user satisfaction? There are a number of decisions to be made.

4.2.1 Connection Protocols

The network can create links that are united via either a line-switched or message-switched protocol. As mentioned before, in line switching all links pass through regional or local switching centers where a direct physical connection between the two nodes is made. The connection lasts only for the duration of the call. Line switching can create an unacceptable delay (on the order of 10-15 sec.) in a networking situation where many short messages are transferred. Line switching is an acceptable discipline only in those cases where the traffic consists primarily of infrequent but very long messages.

In message switching there does not need to be a direct physical connection between the two specific nodes. The message carries with it its destination address. The Communications Interfaces are responsible for storing the message in a local buffer and then determining (either via fixed routing tables or via stochastic methods) a good path to the destination. This type of message switched environment is frequently called store-and-forward message handling. Typical delay times for a store-and-forward distributed network are on the order of 100-500 msec. (FRAN72)

A special form of message switching called packet switching has been adopted for use in both the ARPA and CANUNET networks (CARR70, FRAN72, CANU72). The messages that might be transmitted along the network can be widely varying in length from just a few bits up to thousands of characters. This can create a severe problem in the design of an effective buffer allocation scheme. To alleviate this problem all messages are segmented into maximum size physical units called packets. In ARPA the packet in 1000 bits. If a message is longer than this fixed size it is chopped up and sent as a multi-packet message, each packet travelling independently of the others. The Communication Interfaces at the sending and receiving nodes are responsible for the assembly and disassembly of the separate packets back into a single logical message. A good analogy to packet switching can be drawn between the logical unit called "program" and the physical unit used in time-sharing systems called "page of memory".

4.2.2 Network Composition

A network may be composed of connections between HOSTS that are all identical machines or from the same family of compatible machines. This is termed a homogeneous network. When the HOSTS are allowed to be widely varying machines (e.g. speed, size, or manufacturer) it is called a heterogeneous network. While the difference between the two types of networks might not seem so great at first glance, it makes a tremendous difference to both the designer and the user--primarily in the classes of service that can be provided. Load levelling and program sharing are much more difficult in a heterogeneous network. In a homogeneous network, however, the user cannot access as wide a range of unique physical resources--all nodes are typically the same or very similar. It is also a severe limitation on the class of potential network subscribers since it is limited to those users with a particular configuration. A good discussion of homogeneous vs. heterogeneous networks can be found in (ELIE70).

4.2.3 Network Organization

There are basically four possible topologies in network design. There are described below.

a. Centralized Networks

All control features of this type of network (see Fig. 8a) are contained in a single node. Messages between any two nodes in the system must pass through this control node. This design is unacceptable in all but a

few exceptional circumstances. A centralized design usually creates a series of very long (and therefore very expensive) communication lines to connect the central node to remotely located nodes. The failure of any single line in the network automatically isolates a node from the network and the failure of the central facility is catastrophic. Also the central facility can quickly bog down in an environment where many, many short messages are being transferred. Centralized networks are used primarily in those situations where the power or resources of one node are orders of magnitude greater than all others in the network, such as in the LCS network design (DESA73), where the geographical distribution of the nodes is very small, such as TUCC, or where having a single central control point to monitor all network activities is an important factor, e.g. Network/440. There are also the political and administrative problems mentioned in (BROO68) of choosing a single central site from among many competing locations.

b. Decentralized Network

A decentralized network (Figure 8b) can be more accurately considered to be a network of centralized networks. It in an attempt on the part of the network designer to reduce some of the more prohibitive costs involved with the existence of very long and expensive communication links. It also helps to relieve somewhat the problem of the catastrophic failure of the network

following the "crash" of the central facility. The control operations are now spread among a small set of machines instead of a single unit. If one of these centralized machines goes down a large portion of the network is still usable. This type of network design is frequently used for networks of line concentrators.

c. Distributed Networks

In a distributed network (Figure 8c) there is no single control center or even set of control centers. Every node in the network has its own control and store-and-forward capabilities. Using the appropriate design considerations the cost of a distributed network can be kept significantly below the cost of the two previously mentioned designs. The failure of a single line frequently will not isolate an individual node because there usually exist multiple paths between any two nodes. Even if the failure does isolate a node all other nodes in the network can typically remain active. The classic example of a distributed network is the ARPA Network. When there is a point-to-point connection between every node it is termed a fully distributed network (Figure 8d). Examples of this type of structure are the OCTOPUS and MERIT networks. Figure 7 shows a cost vs. delay graph for various centralized and distributed networks (FRAN70).

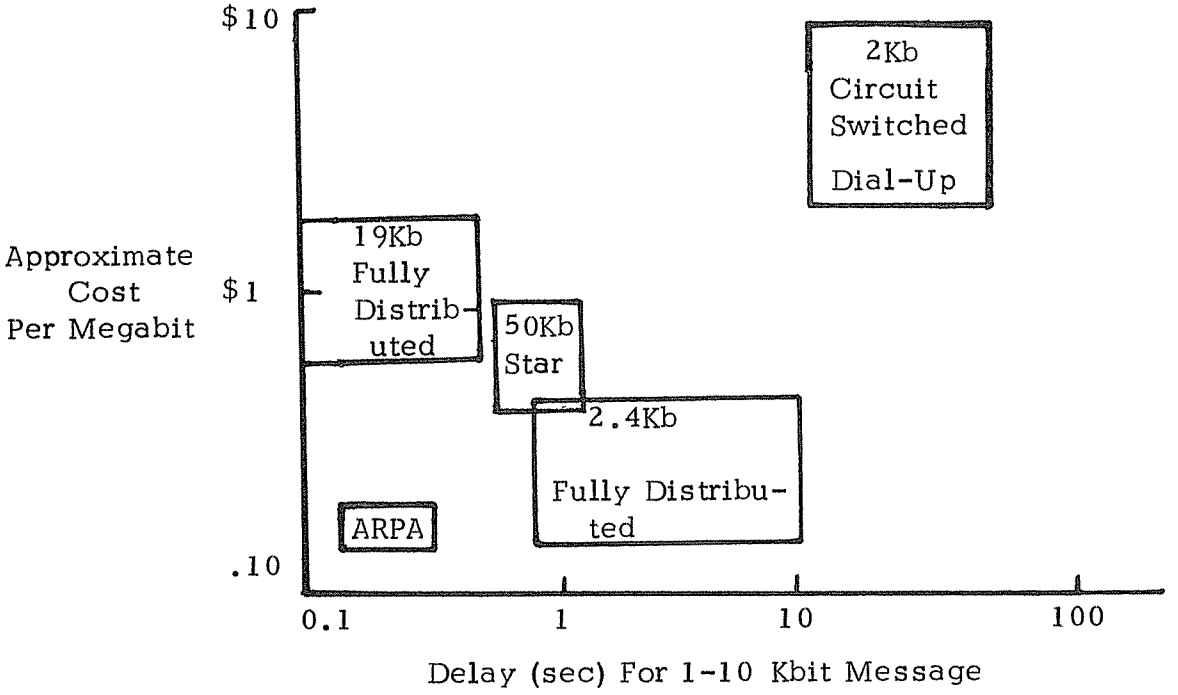


Figure 7. Cost ($) vs. Delay (sec.) for Various Configurations

d. Data Rings

The ring topology is primarily the result of research being conducted by David Farber at the University of California - Irvine (FARB72a, FARB72b, FARB72c). A ring network (Figure 8e) consists of a series of terminals, HOST computers, and other ring networks hooked together via a hardware device called a ring interface to a multi-drop, uni-directional communication line. To send a message on the ring, one addresses another unit by name rather than by address or physical location on the ring. The unit sending the message must wait until there is an

empty block for the message to be placed in. When an idle block comes by the message is placed on the ring where it travels circularly until one of the ring interfaces recognizes the destination name and pulls the message off the ring. The control of the ring structure can either be distributed around to all the elements in the ring or delegated to one unit which typically is called the ring controller. This device must handle such problems as timing synchronization, infinitely looping messages, or malfunctioning ring interfaces.

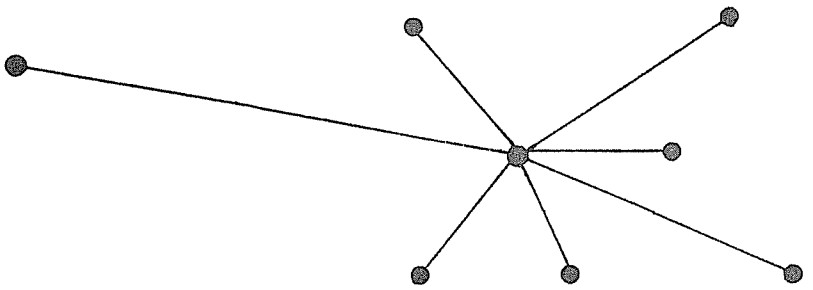


Figure 8a. A Centralized Network

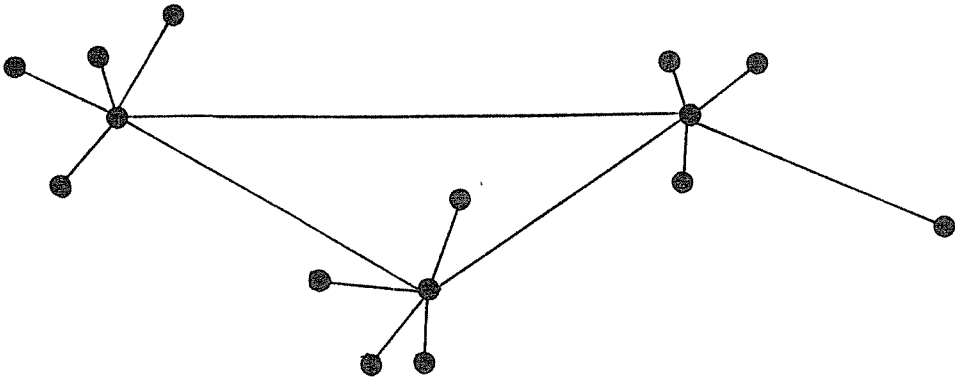


Figure 8b. A Decentralized Network

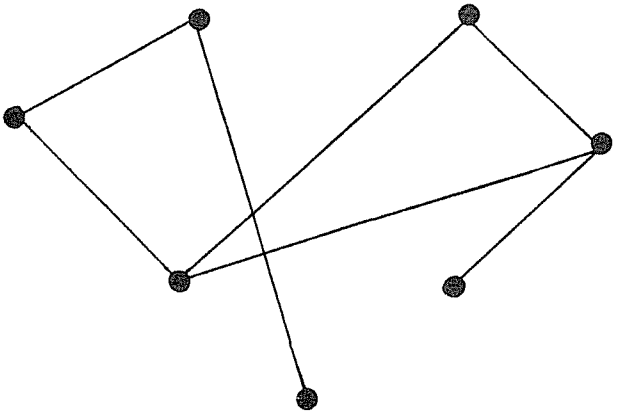


Figure 8c. A Distributed Network

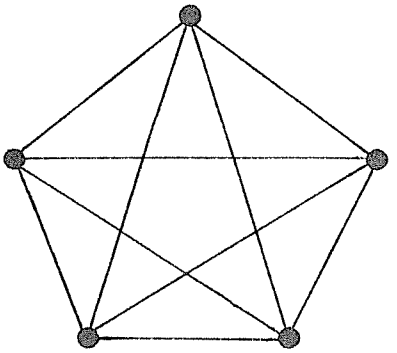


Figure 8d. A Fully Distributed Network

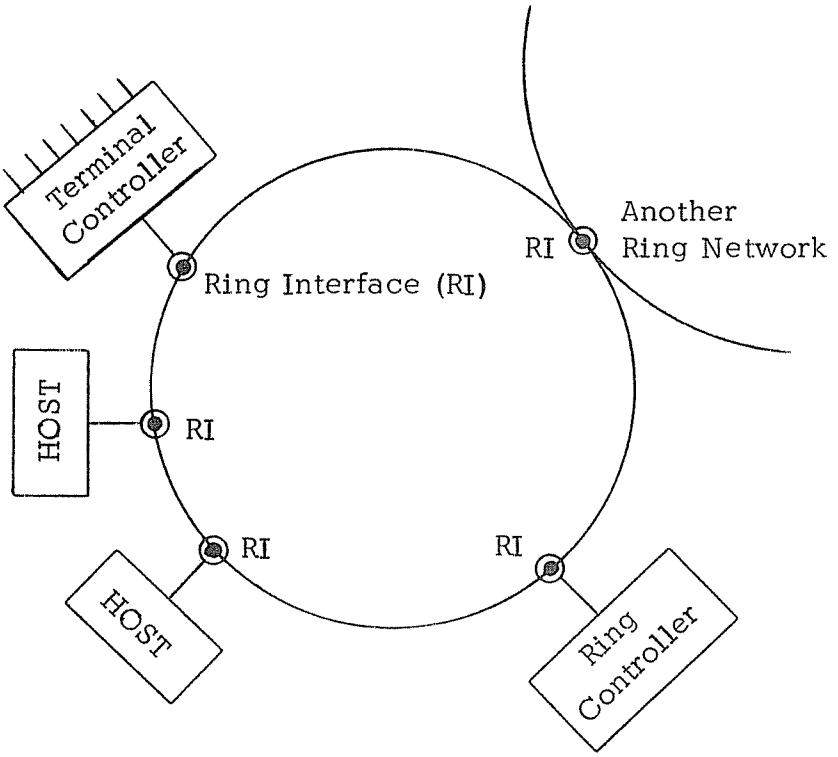


Figure 8e. A Data Ring Structure

5. COMMUNICATIONS PROTOCOLS IN NETWORKS

The previous section was concerned with hardware and the topology of computer networks. However even after a design has been fixed and the hardware chosen another major design effort is still required--deciding all of the various network communications protocols. A protocol is defined as "...the set of agreements on the format and relative timing of messages to be exchanged." (CROC72) More simply it means those particular techniques which will be used to transfer messages back and forth between various points in the network and the division of the responsibilities for doing this between the various network components. Protocols are almost always multi-layered. The high-level protocols are those dealing primarily with communications between individuals or user processes and are concerned with the exchange of textual information. The lower-level protocols are those concerned with communication between computers or communication interfaces and deal primarily with the mechanics of communication. Figure 9, (from CROC72), depicts the relationships between the various layers of protocols.

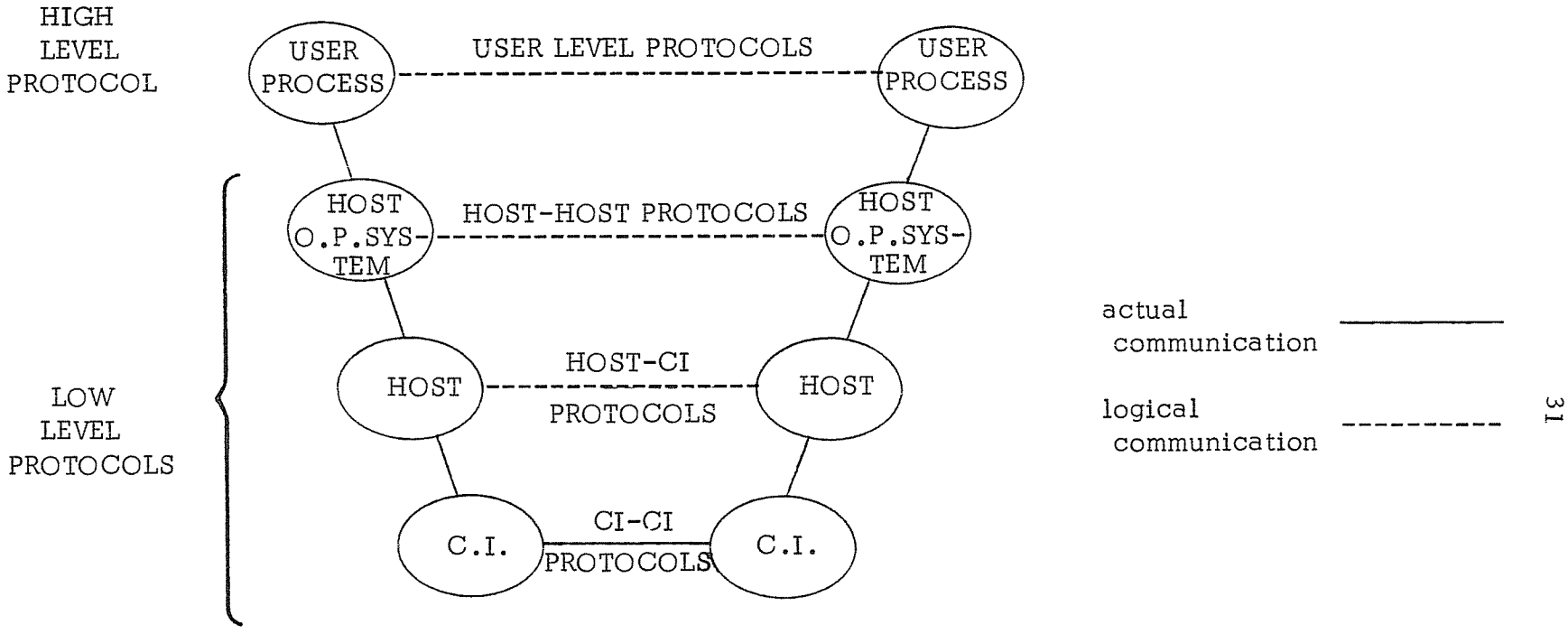


Figure 9. Layers of Protocol

5.1 The CI-CI Protocol

The communications protocol between the communications interfaces is primarily concerned with the problems of routing, flow control, and error control. When a message is to be sent on the network the CI-CI protocol is responsible for determining the way that the communication path will be chosen. This may be done in several ways. The routing may be pre-set by the network designer and be unchangeable except by human intervention. The routing may be allowed to be dynamically modified. If that is the case then the CI-CI protocol must include the description of how this modification will be handled. For example, one possible protocol is to have each CI pass on to its immediate neighbors, at fixed intervals of time, their approximation of the time delay to all the other nodes in the network. This is the actual routing protocol used in the ARPA network and is described more fully in a later section. The CI-CI protocol must also be able to deal with the problem of message congestion. It is possible that messages will be delivered to a HOST faster than they can be handled. This might lead to either buffer overflow and the subsequent loss of information or else a condition known as reassembly lockout. In this situation a CI is filled with many partially assembled messages but has no more room in its buffers to finish the assembly of any single message. The CI is effectively "dead".

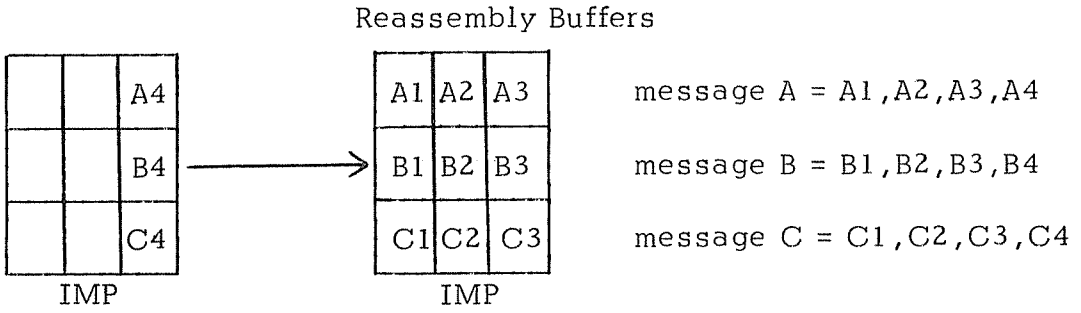


Figure 10. Simplified View of Reassembly Lockout

The CI-CI protocol for the ARPA network handles the problems of clogging and congestion by using the concepts of links (logical paths) and RFNM's (Requests for Next Message). This is described in a later section.

The CI-CI protocol also must handle the error checking of all incoming messages. If a message is found to be correct then a positive acknowledgment must be sent. If a message is found to be incorrect it sends a negative acknowledgment and must initiate retransmission of the message. The protocol must also allow for the possibilities of duplicate messages or incomplete messages due to a missing packet.

5.2 The CI-HOST, HOST-CI Protocols

This protocol is concerned with the transmission of a message from one physical HOST to another. The CI-HOST conventions must include among other things a description of the exact content of headers and trailers that must be affixed to all network messages. There must also be a way for the CI to distinguish different types of messages (e.g. control vs. text) because of the possibly different

handling methods for each type. Additionally, any procedure for returning an acknowledgment to the sending HOST must be agreed upon as part of the CI-HOST protocol. Ideally the CI-CI protocols should be completely transparent at this level. A HOST should be able to present a logical message to its associated CI for delivery to a remote HOST and not be concerned about the communications problems involved in its delivery or worrying about how the response got back.

5.3 HOST-HOST Protocol

This protocol concerns itself with "the set of rules whereby HOSTS construct and maintain communications between user jobs running on remote computers." (CROC72) It is the responsibility of the HOST-HOST protocol to establish and break connections. A connection is a one-way communication path between a process in a sending HOST and a process in a receiving HOST. This protocol must be able to:

a. Initiate a connection - The sending HOST must create a sending port locally and then send a request for a receive port to the receiving HOST.

b. If successful it must establish a connection between the two ports.

c. Break the connection when the communication is done and return the resources (logical ports) to the resource pool.

5.4 User Process-User Process Protocol

This is the highest level protocol in the network. It attempts to interface two user processes in a way that makes all of the

preceding protocols transparent to him. Typically this is done by providing the user with a series of high-level communications primitives which remove him from all the busy work and messy details of operating systems and communications equipment. In most cases these protocols are implemented in some kind of high-level, easy-to-use Network Control Language (NCL). A good example of a User Process-User Process Protocol is the File Transfer Protocol (FTP) of the ARPA network. A user can easily migrate any file to any node in the network via a few simple commands in a language called TELNET:

SENDFILE. Substitute the contents of a designated file for regular control stream input.

SCRIPT. Direct a local file as the output sink for all succeeding output.

The user can handle the movement of the files in a simple fashion completely oblivious to the enormous protocols involved in the file migration.

A complete discussion of all the various protocols involved in network design can be found in (CARR70,CROC72).

6. EXISTING NETWORKS OF COMPUTERS

6.1 The ARPA Network

The ARPA Network represents the largest effort yet in Network Research and Design, and is the most likely candidate for a National Data Network. A general discussion of the ARPA Network can be found in (ROBE70, HEAR70, CARR70, FRAN72, CROC72, ROBE67, COLE71).

ARPA (Advanced Research Projects Agency) is a distributed, heterogeneous network designed to connect together all ARPA-sponsored research centers. The network creates a pool of diverse software and hardware resources which can be called upon by any local program in the network. The topology of the ARPA Network (as of December, 1971) is shown in Figure 11 (ORNS70). The network consists of two subdivisions.

a. The HOSTS. The local HOSTS run the gamut from PDP-11's to a 360/91 at UCLA and the ILLIAC-IV at Ames Research Center (proposed).

b. Communications Interfaces. The interfaces consist of modified Honeywell DDP-516 computers connected via leased 50Kbit communication lines. The machines (called IMPS) handle all the communication responsibilities. They use a store-and-forward message switching protocol. A message is routed from one IMP to another until it reaches its destination. At each step the message is stored until it can be determined whether or not it has been successfully received. The routing procedure is adaptive and is updated every few seconds on the basis of information passed to it by neighboring nodes as well as its own information on the disposition of previous messages put

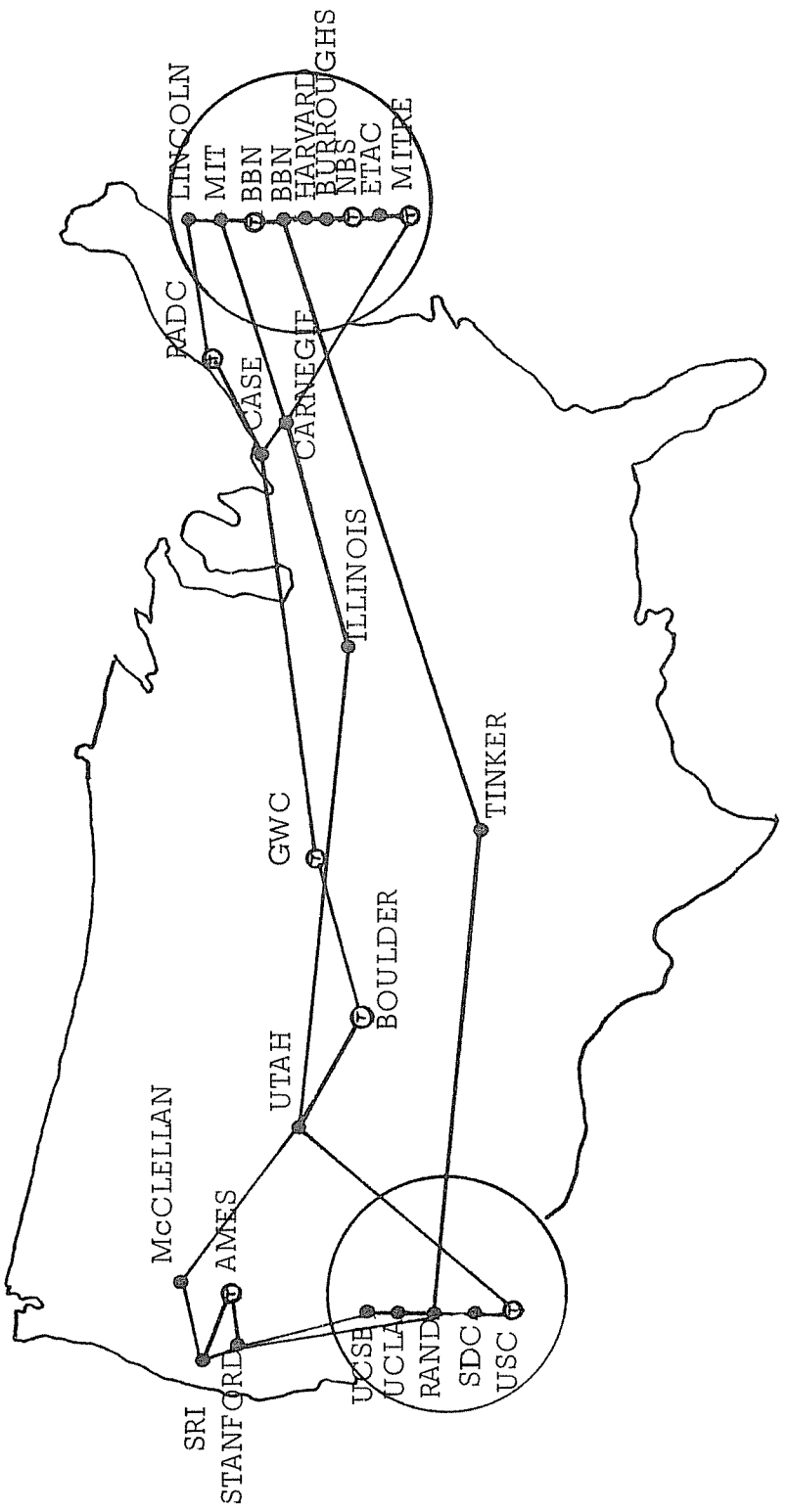


Figure 11. The ARPA Network (December, 1971)

out on a particular link.

There also exists a special form of IMP called TIP (Terminal Interface Processor) which allows terminals to have direct access to the network. The TIP performs all the IMP functions but also has an additional 8K of core and a terminal interface (called a multi-line controller-MLC) which has hardware provision for hook-up to sixty-four terminals of widely varying characteristics. The MLC can handle asynchronous (start-stop) terminals with speeds ranging from 75 bit/sec. to 19.2 Kbits. It can handle any synchronous terminal with a rate up to 19.2 Kbits. A description of the characteristics of a TIP can be found in (ORNS70, KAHN72).

6.2 CANUNET

In November, 1970 the Inter University Communications Council (EDUCOM) queried all Canadian Universities about their possible interest in joining the ARPA Network. From this initial investigation came a national proposal that a Canadian University Computer Network should be established independent of ARPA to allow Canadian Universities to inter-communicate without having to resort to the North-South links of an American Network.

In March, 1971 the University of Quebec began a study to initiate work of CANUNET -- an all Canadian inter-university computer network. Four study groups: Utilization of Networks, Network Design, Communications, and Institutional Framework were created, and by March, 1972 the initial network concepts had begun to take form (CANU72). A proposed eighteen node network topology is shown in Figure 12.

The Network design follows very closely that of the ARPA Network. The communications interfaces will be physically separate machines (called Node Computers (NC) in CANUNET terminology). The communications protocol will be store-and-forward message switched with adaptive routing. The type of communication links have not been decided upon but simulation studies are being performed on various combinations of line speeds from 4800 baud all the way to 50 Kbaud. The major differences between ARPA and CANUNET which have so far been identified are (CANU72):

a. Division of functions between the software of the HOST Computers and the node computers (or NCs) is more clearly defined in CANUNET. This will simplify the node computer and its software without significantly increasing the work of the HOSTs.

b. The interface between the HOST and the node will be microprogrammed to provide maximal efficiency and speed of information transfer.

c. Terminals will be attached to the node through a separate mini-computer called a "pseudo-host." These mini-computers should be much cheaper and have less complex software than the TIP processor in ARPA.

d. CANUNET, which is being developed about four years later than ARPA, should be able to take advantage of advances in hardware during that period (e.g. mini-computers, modems).

CANUNET represents another major national effort to develop a large general purpose Computer Network which will make a wide resource pool available to a large user population. Other national

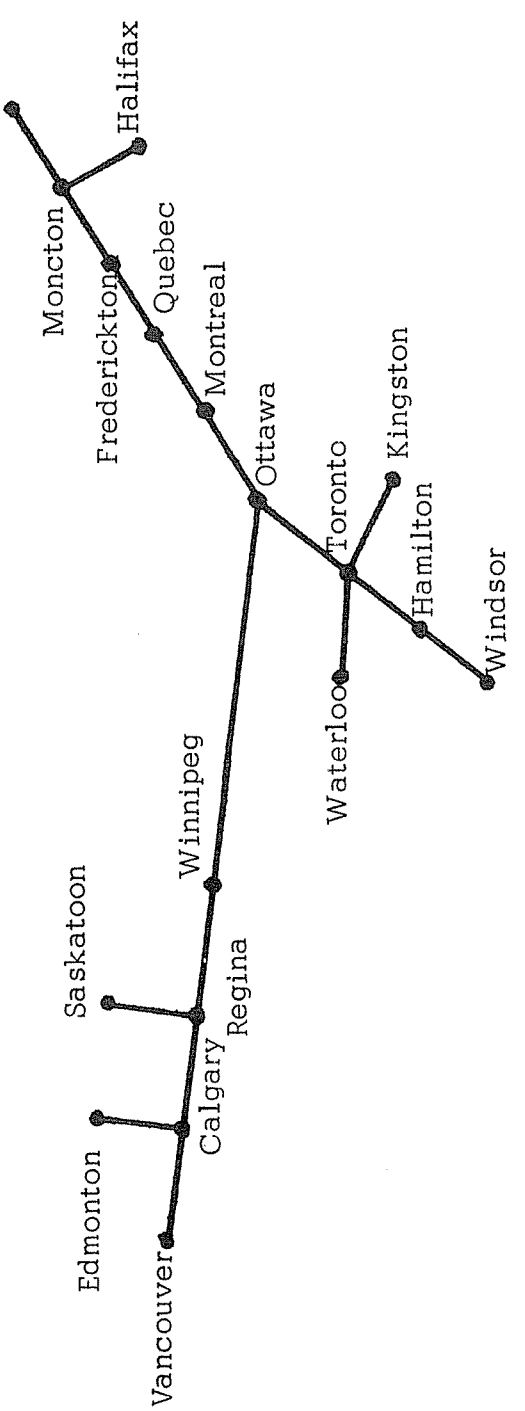


Figure 12. A proposed 18-node CANUNET Configuration (CANU72-Appendix M)

networks being planned or developed include the NPL Network (National Physical Laboratories) in England (DAVI68, BART67, SCAN71) and the European Computer Network (BARB72). Both of these are very much "ARPA-like" in their design and philosophy.

6.3 The MERIT Network

The MERIT Network (Michigan Educational Research Information Triad) is a cooperative network effort between Michigan State University, University of Michigan, and Wayne State University. (Figure 13) (HERZ72, AUPE72, COCA72).

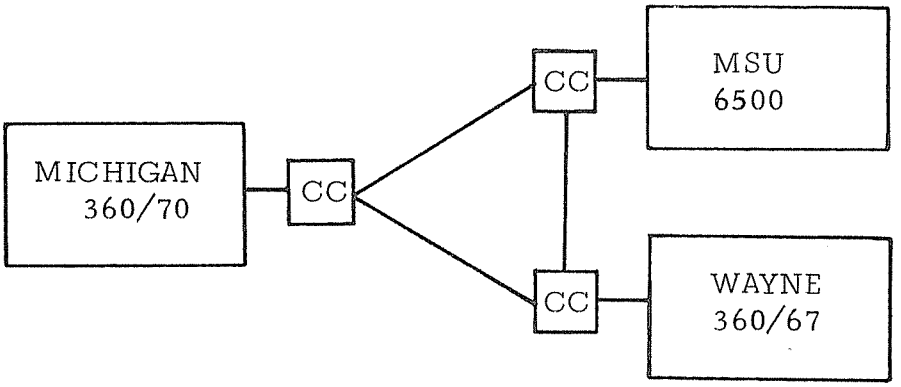


Figure 13. MERIT Network

Its primary aim is to create an educational computing network to allow the computing resources at the member schools to be shared while not jeopardizing the existing institutional autonomy of the separate centers. The MERIT design philosophy attempts to make a minimum of impact on the HOSTs themselves.

The HOST computers consist of 360/67s running under MTS at Michigan and Wayne and a CDC 6500 operating under SCOPE at MSU.

The communications aspects of the Network are handled by four distinct components as shown in Figure 14.

a. The communications computer (CC) is a standard "off the shelf" 16K PDP-11/20. Its functions are similar to those mentioned in Section 4.1.2.

b. The communication links are standard 2000 baud full-duplex dial-up lines. There was a major question in the network design stage on whether to use voice-grade or wideband facilities. A significant argument in favor of wideband facilities is the elapsed time required for transmitting a record over the link. A 125 character record (1000 bits) would require about 1 second for a round trip transmission over a 2Kb line, quite discernible to a frequent interactive user. However a 50Kb line would have only a 40 msec. delay -- completely unnoticeable. The final decision was made, however, strictly on the basis of economics and the 2Kb voice-grade lines are presently being used.

c. The line interfaces are Bell 201A Line Interface Units (4 per node). The 801C is an Automatic Calling Unit (ACU). The ACU allows the CC to dynamically vary the communications capacity as the need arises. The CC will be continuously monitoring the average traffic load between nodes and can automatically dial-up another full-duplex line. Thus the number of actual lines between any two nodes can vary from none up to a maximum of four.

d. The HOST interface allows the CC to look like a multitude of peripherals to the HOST computer. Even though there is

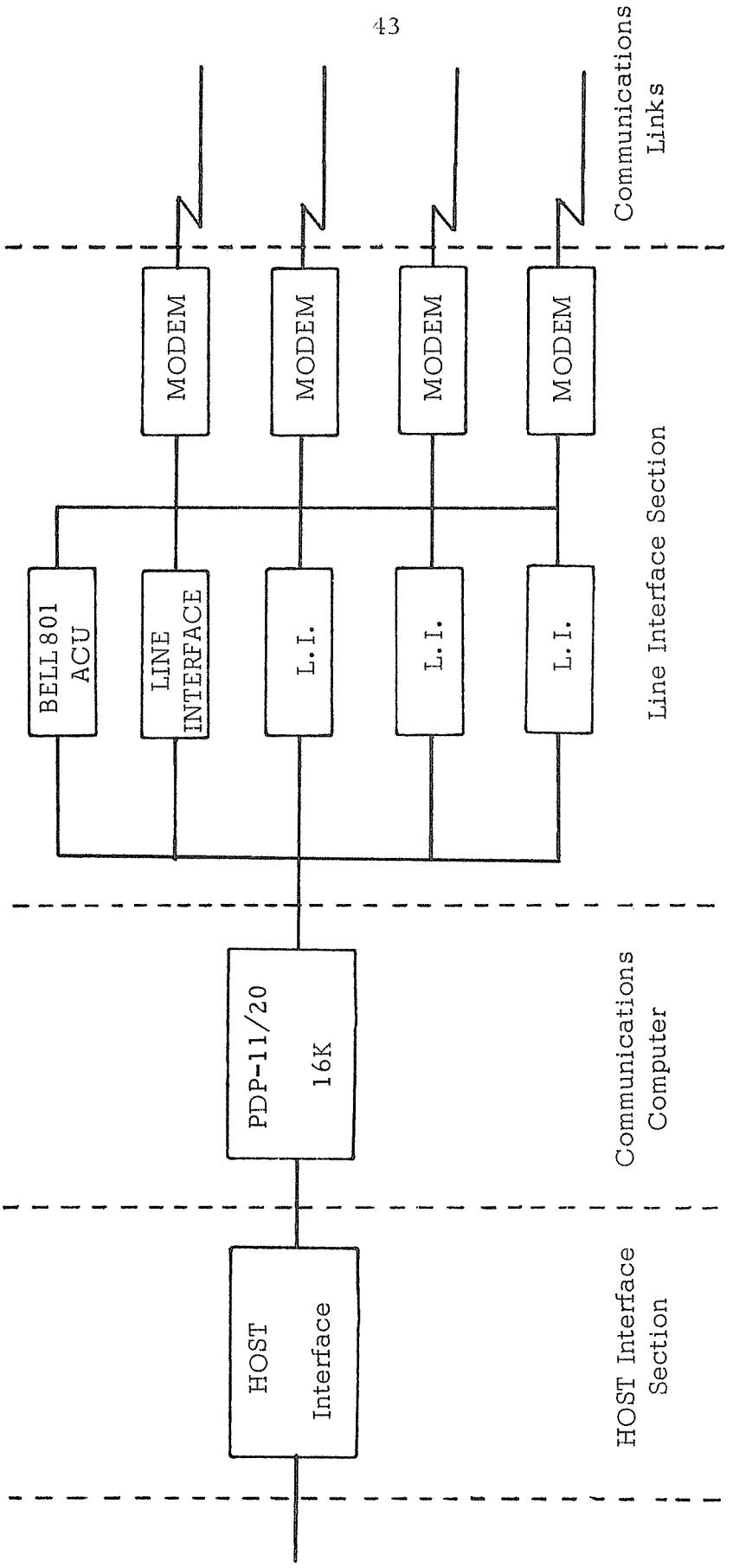


Figure 14. Communications in the MERIT Network

only one actual physical interface, the unit will respond to a whole series of different addresses. Thus the CC acts like similar but independent devices. The individual programs operating in the HOST can each have their "own" CC with which to communicate with other nodes without worrying about other processes fighting for this physical resource. This is of great advantage in the time-sharing environment of the HOSTs and has allowed a minimum of impact to be made on the HOSTs' operating systems.

6.4 CYBERNET

The CYBERNET Network is a true representative of a currently operating commercial (available to the public) network (LUTH72). The network was originally formed with the idea of connecting all the existing Control Data Corp. data centers in order to:

a. gain reliability through the availability of alternate machines

b. level the computing load between unbalanced nodes

c. share the available resources.

CYBERNET has now grown to 36 nodes offering a "kaleidoscope" of communications facilities and computer resources -- all of which are accessible to the general public.

The central resources in the network are called centroids. Presently they consist of CDC 6600 machines only, but will eventually be expanded to include 7600s as well. The centroids are used primarily to perform high throughput batch jobs submitted by a remote user as well as doing some interactive computing.

The communications functions of the network are handled by the nodes. Presently these consist of CDC 3300 and 6400 computers. The nodes are responsible for all the regular functions of a general CI as well as handling the job of line concentration to the centroids and sending jobs to nodes or centroids that are presently unloaded. Also the nodes, unlike ARPA, have a stand-alone computing capability. The nodes themselves can execute both batch and interactive jobs submitted by other points in the network.

The communication lines available are 2000 baud dial-up for remote terminals, 2400 baud dedicated lines between terminals and nodes or between terminals and centroids and 40.8Kb wideband facilities between nodes or centroids themselves.

The terminals allowed in the network are virtually unlimited. CYBERNET has over 600 registered names in its terminal directory. They all fall into four general categories however:

a. Conversational Terminals -- Typically a keyboard/printer device similar to a model 33 or 35 TTY.

b. Medium Speed Peripheral Processors -- Typically these are processors driving line printer/reader/punch devices such as the CDC160A. Speeds can range from 2400 to 9600 baud.

c. Satellite Computers -- These remote computers differ from a peripheral processor in that they have a stand-alone computing capacity. However, they can appear to the network as just simple terminals or peripheral processors. Examples are CDC1700 or CDC3150 computers.

d. Time-Sharing Computers -- Computers with their own terminal facilities such as CDC3300 can still be considered simple terminals to the network.

Unlike the ARPA network with its rigid HOST-IMP link, the ways of connecting up these three components -- terminals, nodes centroids -- are quite wide. Terminals may be connected to either nodes or centroids. Nodes may be connected to either other nodes or centroids also. The entire network is much too complex to display in its entirety but a sample, reproduced from (LUTH72), is shown in Figure 15 to give an idea of the interrelationships between components. The network makes frequent use of the "look-alike" technique in which an entire class of devices is made to look like a particular standard device which that unit can communicate with. For example a CDC 3300 can be made to look like a 200 User Terminal. This has allowed each central site to be programmed independently of the others. CYBERNET is presently operating as a commercial entity and is offering general computational services to its users.

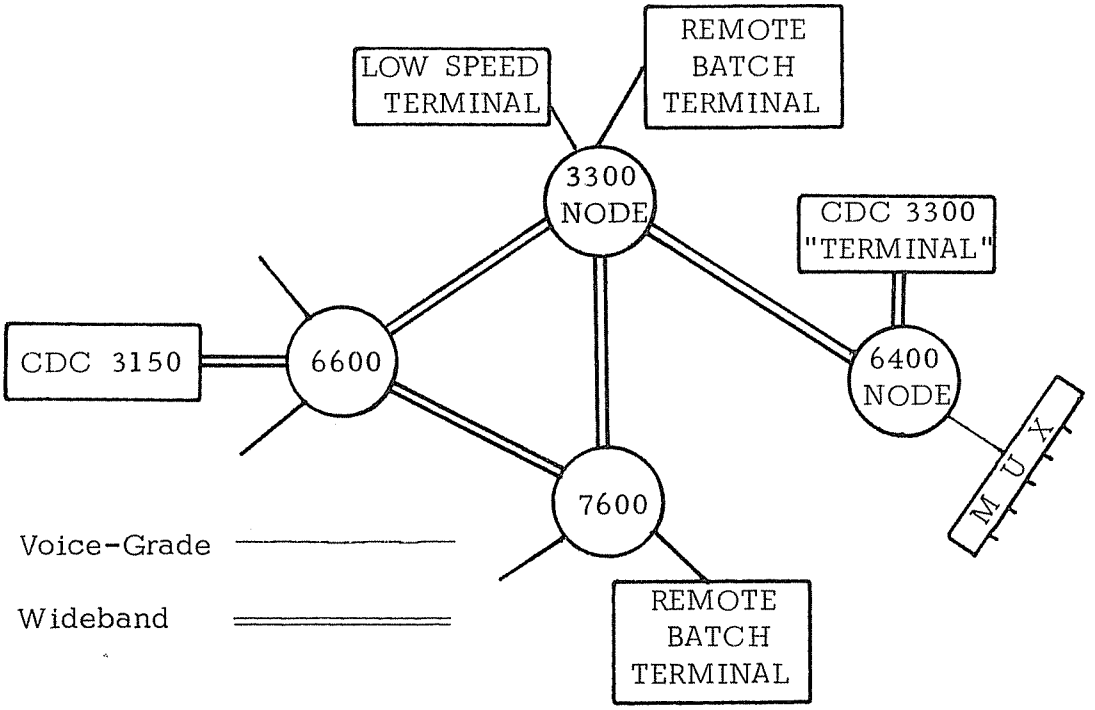


Figure 15. Sample of the CYBERNET Network

6.5 OCTOPUS

The OCTOPUS network is a large, distributed, heterogeneous network currently in operation at the Lawrence Radiation Laboratory (MEND72). It is composed of semi-independent sub-networks which handle the separate functions of data base sharing, remote job entry, and terminal access to the network resources. The configuration as of Fall, 1970 is shown in Figure 16.

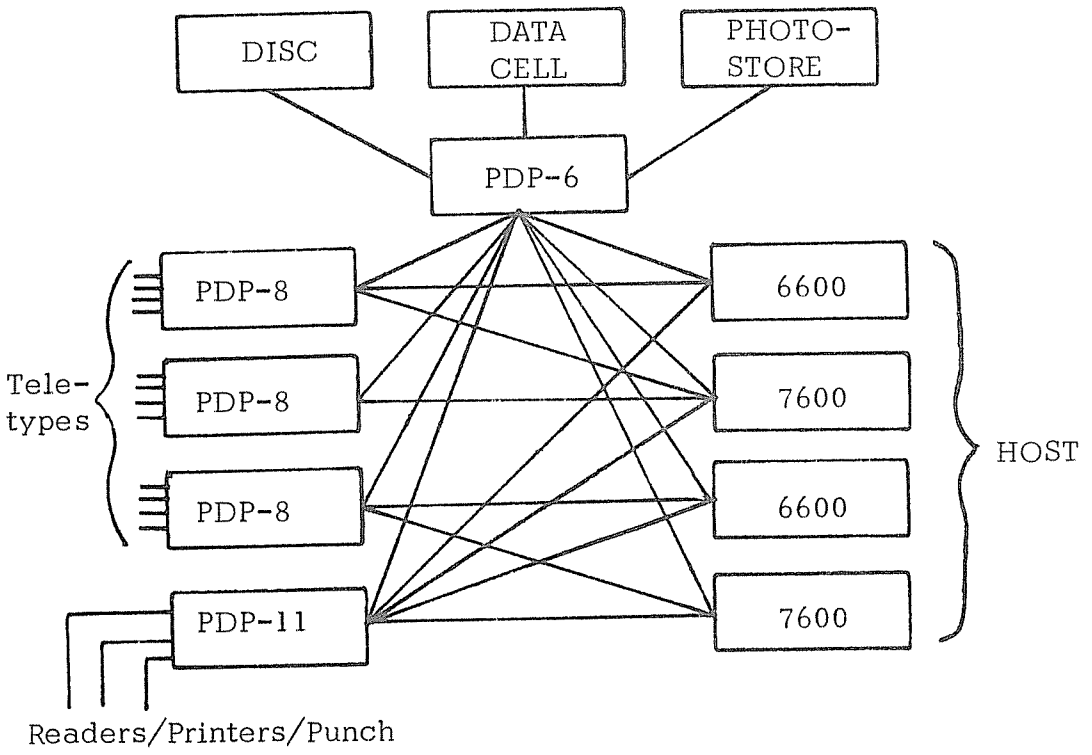


Figure 16. The OCTOPUS Network

a. Teletype sub-network -- This network consists of three PDP-8's which can each handle up to 128 TTY's. Through this sub-network the terminals can be connected to any HOST machine. The PDP-8's handle all the communication responsibilities including multiplexing, serialization, packing/unpacking of messages, buffering, and routing.

b. File Transport sub-network -- The PDP-6 computer handles the management and maintenance of the network's shared data base. The hardware consists of a 10^9 bit fixed head disc, a 10^9 bit IBM 2321 data cell and a 10^{12} bit IBM 1360 Photodigital store. Data can flow between the data base and any HOST in the network at a rate of about 10 megabits. The basic unit of transfer is the file which can be of varying length up to a maximum of 64K.

c. Remote Job Entry Network -- A Remote Job Entry Sub-Network of dual PDP-11's is planned for operation in Fall, 1971. One PDP-11 will be used to connect the sub-network to the HOST machines and the file management sub-network via a selector channel. The second PDP-11 will be used for line concentration of up to eighteen remote job entry terminals. Each terminal will typically consist of a PDP-8/L with a 600 lpm printer and 300 cpm reader. The HOSTs consist of CDC 6600 and 7600 machines and (proposed) a CDC STAR. The communications are done via hardwired 12 megabit coaxial cables because of the physical proximity of all the HOSTs.

Essentially OCTOPUS is a network of sub-networks superimposed on each other, each one independent and performing a specific function.

6.6 DCS

The DCS network is an experimental network being build at the University of California-Irvine (FARB72b, FARB72c). Its primary function will be experimental -- to investigate the nature of data ring networks. The topology of DCS is similar to that of Figure 8e.

Messages circulate uni-directionally around the ring. The units of transmission of the ring are called frames. The frames consist of about 300 data bits, about 9 bits of addressing, and a busy/not busy bit. To transmit, a node puts its message into a shift register and watches the frames coming by. When an idle frame comes by (not-busy bit set) the message is shifted onto the ring. The addressing is done by processor name rather than location on the ring so processes and processors can migrate in their location on the ring and still be successfully located. A device called the Ring Interface (RI) connects the processors to the ring. The RI handles the routing by always maintaining an up-to-date list of the names of all active processes in its attached processor. As a message frame passes by, the RI compares the destination name with its own list and accepts the frame if there is a match. There will be three kinds of ring interfaces -- one to attach a terminal controller, one to attach a processor, and one to handle the attachment of another ring. The latter will allow the creation of networks of local rings.

By distributing much of the control responsibilities into the nodes themselves or into the ring interfaces it is hoped that DCS will achieve another important aim -- a simplified design and a modest start-up cost.

6.7 Other Networks

A summary of the characteristics of the networks described in this section is presented in the Appendix. This table is in part a reproduction of Table 1 from (FARB72b).

Other networks that should be mentioned briefly because they represent significant steps in the area of Computer Networks are listed below:

a. IBM Network/440 - A centralized network being designed at the IBM Research Center in Yorktown Heights. The central node is an IBM 360/91 with twelve other nodes presently in the network (MCKA72).

b. TSS/360 (WEIS72, ELIE70) - This is a nine node distributed, homogeneous network of 360/67 computers. Some of the nodes include Carnegie-Mellon, Princeton University, and Bell Laboratories.

c. TUCC - (BROO68, PARK69) This is a centralized, homogeneous network linking together three major Southeastern universities -- University of North Carolina, North Carolina State, and Duke University. The central node is an IBM 370/175 at the TUCC Research Park.

d. BROOKNET - (DENE68) This is a network created by the Brookhaven National Laboratories. It is a centralized network connecting a single large machine (CDC 6600) and several small and mini-computers such as PDP-11 and Sigma-7 computers. The network allows the sending and receiving of files between the central and remote nodes and allows the remote initiation of program execution from a central file.

e. TYMNET - (BEER72) A commercial network of the Tymshare Corporation in which Varian 620 mini-computers are used as intelligent "front-ends" to a group of XDS-940 HOSTs.

7. RESEARCH AREAS IN NETWORKS

7.1 Interprocess Communications

There are basically two differing approaches to utilizing a network of computers (CANU72 - Appendix K):

a. Terminal-Computer Communications - In this approach one machine, M_1, is considered only as a terminal for submitting a job or data file to another machine, M_2. The process is run entirely on M_2 and the results returned back to M_1. In this type of communication the relationship between nodes strongly resembles the terminal/computer relationship in a time-sharing environment. However, this mode of operation is satisfactory to achieve most of the services described in Section 2.

b. Computer-Computer Communications - In this approach disjoint parts of a process are run concurrently on different machines in the network. This mode of operation allows more sophisticated usage of a network such as dynamic run stream modifications and speed increases through some degree of parallelism. However, this type of communication raises the thorny problem of interprocess communications. In general the problem can be divided up into four distinct parts:

   i) Establishment of a communication path between processes (START) -- There must be some process existant in all HOSTs whose sole purpose is to recognize initial requests for connection.

   ii) Blocking/Unblocking of processes (SLEEP, WAKEUP) -- We must have the ability to block processes until the

occurrence of an external event and unblock them when that event has occurred.

iii) Actual transmission of data between connected processes (RECEIVE, SEND) -- Every HOST in a heterogeneous network has its own internal naming scheme for processes. It is not practical to impose a uniform naming scheme on every HOST so we have the problem of how to create some kind of intermediate "network address". We must also be able to then map this intermediate address into the internal process identifiers of the HOST.

iv) Termination of a connection --(END) We must be able to break the connection at the end of the call and return all resources required to make the call to the resource pool.

The following will be a description of how interprocess communication is handled in the ARPA and NPL networks. A further discussion of these methods can be found in (CARR70, CROC72, SCAN71, ELIE70, WALD72).

In the ARPA net a process communicates with a foreign process through a program called the Network Control Program (NCP) which is resident at each node. It is the function of the NCP to establish connections between cooperating processes. As mentioned before, it would be unfair to impose a common process naming scheme on all HOSTs. Instead an intermediate name space is used and it is the responsibility of each individual HOST to map from these network addresses into their own local process identifiers. The elements of these network addresses are called sockets. A socket forms one end

of a connection. The socket syntax is:

(socket)	:=	(send socket)/(receive socket)
(send socket)	:=	(socket body) 1
(receive socket)	:=	(socket body) 0
(socket body)	:=	(host #) (user #) (port #)

(port #) is a 7-bit value thus allowing 128 sending and 128 receiving sockets. A connection is a 1-way communication path between a send socket and a receive socket. For full-duplex operations two connections and four sockets are required.

(connection) := (my socket body) (B) (your socket body) ($\bar{B}$)

(B) := 0/1 and $\bar{B}$ is the complement of B

All user requests for connections are handled by system calls to the NCP. The NCP is responsible for interpreting these calls and creating the appropriate network control commands. All network commands are in the following form:

$$\text{(control command)} := \text{(op code)} \ \{\text{(parameters)}\}_0^n$$

To initiate a connection between a sending and receiving process the NCP's exchange matching Requests for Connection (RFC) messages. A prospective sender sends out the following:

sender-to-receiver (RFC) := 02 (connection)

The prospective receiver sends back:

receiver-to-sender (RFC) := 01 (connection) (link #)

where the (link #) is chosen by the receiving HOST from those available.

· Either the sender or the receiver can refuse a requested connection or break an established connection by sending:

close command (CLS) := 03 (connection)

When the (link #) is assigned the connection is closed and the transmission of textual information can be performed by the NCP along the assigned link.

The initial dialog between the sending HOST and the receiving HOST is always to a program resident in each HOST frequently called the "logger". It is a function of this program to monitor all requests to socket 0 (by convention). The logger's function is to record the user number and HOST number, and close the connection by creating a receive socket and assigning a link to the connection. It must also return these values to the sending HOST. The logger can also iniate a request for a second pair of sockets thus enabling full-duplex transmission. All succeeding communications will be via these links and sockets.

In summary then, to have a process communicate with another the following steps are handled by the NCP:

a. A process call by a currently running process is translated to a system call to the NCP.

b. The calling process is then blocked.

c. A request for a receive socket is transmitted to the logger in the receiving HOST via receive socket 0.

d. If the logger accepts the request it generates a receive socket and chooses a link. It then transmits these values back to the sender.

e. The logger then executes its own request for a receive socket at the sending HOST.

f. If accepted there now exists a full-duplex connection between the sender and the receiver and the initiating process can be awakened.

g. The logger will now interrogate the initiating process and create a new process in the receiving HOST. Communications will be via the established links.

The interprocess communication facility in the NPL network (called IPCM) is similar to the ARPA design. The sockets are termed channels and connections are called calls. There exist 256 logical channels for each network node. Connections are created by CALL messages to a logger process. This message contains the sending channel number and sufficient information to identify the target process. The logger creates a receive channel and returns the information to the sending node. Communication proceeds via these channels until the issuing of an END command. The only basic differences between IPCM and NCP is that the channels used in IPCM do not have to be designated as receive or send. They may function as both. Also IPCM allows the automatic switching of channels. Thus the logger process can automatically switch a process to another channel in such a way that the sending HOST is completely unaware of the switch. In ARPA this switching between sockets must be handled by the NCP in the sending HOST. Other approaches to communications between cooperating processes can be found in (SPIE69, WALD72).

7.2 Flow Control

In general flow control means the "...co-ordination of the dis-

patch and arrival of messages with the availability of buffer space and/or process availability" (KAHN71). Anytime that you have messages freely entering a distributed network you have the possibility of clogging and local congestion. An example of this was shown in Figure 10. Present research has concentrated on three approaches to the problem.

a. Discard Mechanism - This is basically an avoidance of the problem instead of a solution. In this approach the partially assembled packets are discarded by the clogged IMP and a message to that effect is sent to the source IMPs of discarded packets. These IMPs can then attempt to re-transmit the packets at a later time.

b. Link Mechanism - To prevent the situation where messages are being sent to a receiving HOST faster than they can be accepted, the initial design of the ARPA network created the "link mechanism" of flow control (HEAR70). In this approach each HOST has a fixed number of <u>links</u> or logical paths. (A link is not related to any particular physical channel). Only one message at a time may occupy a link and that link is blocked until the receipt of a special command called Request for Next Message (RFNM) which frees that link for further use. This effectively limits the traffic between nodes.

c. Pre-allocation - The link mechanism of part (b) worked well in the ARPA Network under conditions of light loading and "well-behaved" nodes. Under heavy traffic though, or through the malicious behavior of a HOST "spraying" messages over many links the reassembly lockup of Figure 10 occurred almost instantly. Simulation studies conducted at Bolt, Beranek, and

Newman showed that the lockup condition will probably occur when as few as ten links are transmitting to an IMP. The new scheme designed to eliminate this problem involves the pre-allocation of reassembly buffer storage prior to transmission (MCQU72). When an IMP receives a multi-packet message it sends a short control message to the destination IMP to request a buffer large enough for this message. The request is queued until the storage is available at which time a return message is transmitted saying the allocation was successful. The fact that a multi-packet message is never allowed to enter the network until the storage has been allocated prevents the problem of reassembly lockup. It is at the price, however, of a slight decrease in bandwidth caused by the necessary control messages.

Another serious problem in flow control research is called store-and-forward lockup. An example of this is shown in Figure 17 from (KAHN71):

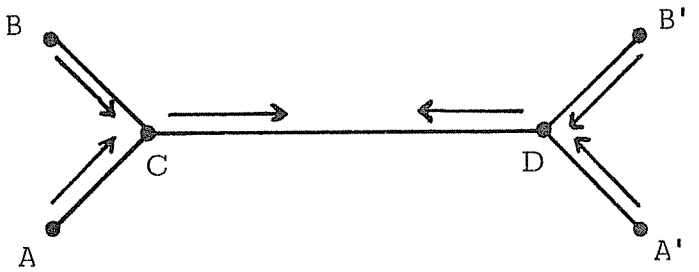


Figure 17. Store-and-Forward Lockup

Assume nodes A and B are transmitting to nodes A' and B' respectively. Node C can become clogged with messages for D and node

D can be clogged with messages for C. Neither can transmit and all traffic is effectively halted. No elegant solution to this problem has been found but a simple solution is to use the idea of overflow buffers (KAHN71). An overflow buffer pool is maintained at all the IMPS to insure that in the event of a traffic flow halt at least one packet will always be able to reach its destination. In this way we can guarantee that some fraction of HOST traffic will always reach its destination. Unfortunately we do this by dedicating a small amount of the most precious and in-demand resource the IMPs possess -- buffer storage. A further discussion of flow control can be found in (FRAN72, DAVI71).

7.3 Routing

A great deal of research has gone into developing routing techniques. A routing technique is a "...technique to decide what path a message will take through a distributed network." (ELIE70) There exist three basic classes of routing techniques:

a. Fixed Routing Techniques - These methods assume a fixed network structure where the path between a pair of nodes is predetermined and unchangeable except by outside intervention.

b. Adaptive routing (local) - Each node estimates the time required to reach a destination from information carried in its own messages and/or passed it by its immediate neighbors.

c. Adaptive routing (global) - One routing center collects all information about the status of the network and dynamically determines the routing tables of all nodes.

Local adaptive routing has attracted the most interest in network research. The so-called "Hot Potato" technique used in ARPA and

proposed for CANUNET is described in (ELIE70). Each node keeps a routing table which lists the order of the most desirable path between nodes.

This table is updated in a synchronous fashion. Every period of time (e.g. in ARPA 0.5 sec.) every node sends to its immediate neighbors an estimate of the shortest delay time to pass a message to each destination. The local node adds to these values the estimated transmission delay from itself to that neighbor. This allows the creation of a delay table (Figure 18a). The routing table (Figure 18b) is constructed from it by the simple expedient of sorting the links by transmission delay time.

	Link 1	2	3	4
A	22	∞	12	10
B	5	3	2	2
C	7	8	13	9
D	7	10	12	14

Figure 18a
Delay Table

	Choice 1st	2nd	3rd	4th
A	4	3	1	-
B	3	4	2	1
C	1	2	4	3
D	1	2	3	4

Figure 18b
Routing Table

However using adaptive routing techniques care must be taken to eliminate the "ping-pong" phenomena in which a message bounces back and forth infinitely between two nodes. So far the only implemented solution to this is to keep in the message header a "node count" field. Every time the message passes to another node this field is incremented. If it ever reaches a maximum value then it is assumed to be irrevocably lost in the network and is returned back to its sending HOST. Additional information can be found in (HEAR70,

FRAN72, KARP71).

7.4 Network Command Language

There are basically two differing philosophies in the design and implementation of a Network Command Language (NCL) (CANU72-Appendix K).

a. You can require a user to learn all the conventions of the command language for the computer he is about to use as well as a separate NCL which allows him to control his call. This is by far the easiest means of implementing a NCL since all that is required is a small layer of language interpretation on top of the already existing command languages. From the user's point of view, however, this can be a severe requirement unless the class of services utilized on the distant HOST is very small or the receiving HOST has only very limited resources (e.g. a PDP-11 doing BASIC). Designers using this approach usually realize these limitations and include some type of HELP or TUTORIAL commands to familiarize the user with the local language environment.

   An example of this type of language design approach is the TELNET language of the ARPA network. An annotated example of an ARPA dialog between a PDP-10 at Utah and an XDS-940 at Stanford Research Institute using TELNET is given in Figure 22 from (CARR70):

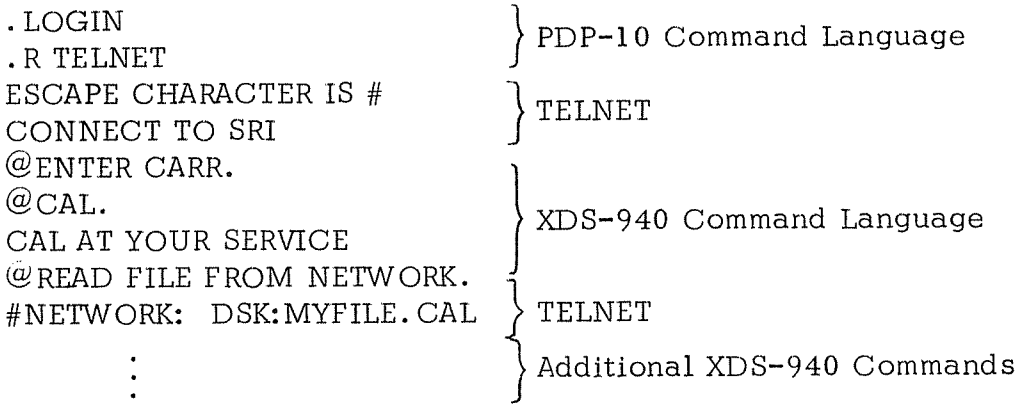


Figure 22. Typical TELNET Dialog

b. Learn an overall NCL which will allow him to exploit all the services offered by any computer in the network in some uniform command language structure.

The development of a very good all-purpose NCL (i.e. one that isn't just a union of all accessible command languages) may well be almost impossible because the various computers are so different in scope and operation that their needs and their services might not be able to be brought under the umbrella of a single NCL.

An example of this approach is the ACL control language for the IBM Network/440 (MCKA72). Some sample commands from ACL are shown in Figure 23. All commands to the network are phrased as ACL command language statements. A network controller resident in the central system interprets these program statements and issues the appropriate commands to manage the transfer of jobs and data through the network.

Regardless of the approach any NCL should have certain characteristics (CANU72 - Appendix P):

```
label:   ROUTE n FROM unnl TO unn2
         OUTPUT n FROM unnl AT unn2
         EXECUTE n FROM unnl AT unn2
         READ dsn, var
         WRITE dsn, var
         IFXX varl, var2, label        (XX = EQ,NE,LT,GT,GE,LE)
         GOTO label
         ASSIGN varl, var2             where  unn = users network node
         START                                  n = name of process or
         END                                        data file
                                              dsn = destination
                                              var = variable
```

Figure 23. ACL Syntax

a. Should allow for the creation and definition of programs and data files.

b. Should allow access to all the available network resources and allow a user to capture any resource not currently assigned.

c. Should make the network protocols relatively transparent to the user. That is, the resources of the network should be as easily accessible as any local resource.

d. It should be able to be invoked at any time during the life of a process. This would allow it to activate network resources before, during, or after the execution of processes.

e. Should be programmable. That is one should be able to dynamically alter the run-time control stream using run-time variables, labels, and conditional and unconditional branches. A good example of a programmable command language is shown in (MINS70).

It is obvious that an NCL cannot be implemented without becoming deeply involved in the architecture and philosophy of the network. The

ease with which it can be understood and used will probably influence as much as anything else the overall network usage.

7.5 Others

The preceding paragraphs tried to highlight important areas of ongoing research in computer networking. However, it is by no means intended to be an exhaustive list. There are far too many current research problems to discuss them all at length. This section attempts just to list, for completness' sake, other important research topics and give the reader references to further information.

a. Optimization of Data Networks (KLEI70, FRAN72, BRWE71, SPRA71, FRAN72b, FLEI72). The overall cost of a distributed computer network is almost entirely dependent on the network topology and the class of communication lines used to link the nodes. The major problem then is to find a network topology which minimizes transmission delays and maximizes throughput for the least cost. The two basic approaches have been:

   i) Perturbation Method (BREW71, FRAN72b) - Given some "intelligent" starting guess, the model makes a series of minor network changes (e.g. add or delete a line) to see if it decreases delay or cost. This is essentially a hill-climbing technique.

   ii) Queueing Theory (KLEI70, HAYE71, SPRA71, KLEI72) - This method attempts to create a mathematical model of the network, usually using classical queueing theory and the assumption that a network can be factored into a set of disjoint single-server queues. For example Kleinrock in (KLEI70) describes a network problem in

the following form: The overall transmission delay T in a network with exponentially distributed message length, Poisson arrivals, and fixed routing procedure:

$$T = \sum_i \frac{\lambda_i}{r} (T_i + 10^{-3})$$

where $$T_i = \frac{1}{\mu C_i - \lambda_i}$$

and

T_i = average delay over the i^{th} channel (sec.)
C_i = capacity of the i^{th} channel (bits/sec.)
λ_i = average number of messages/sec flowing over the i^{th} channel
$\frac{1}{\mu}$ = average message length (bits)
r = total input data rate
10^{-3} = fixed nodal processing time

The problem is the choice of $\{C_i\}$ which minimize T subject to a fixed cost constraint and the existence of such C_i. The solution is shown in (KLEI72).

b. File Migration - There is a large on-going effort to construct some kind of generalized File Transfer Protocol (CROC72, ANDE71, SCHA70) which will allow the exchange of structured files between widely differing file systems with a minimum of user intervention. The protocol should also allow listing remote directories and adding/deleting records, as well as just sending/receiving.

c. Terminal Access to Resource Sharing Networks (ORNS70, KAHN72) - There is a basic problem inherent in all resource sharing networks - how to allow the connection of a device which in and of itself has no resources of its own to contribute to the pool.

i) Direct dial to the appropriate HOST. This is essentially an avoidance of the problem since the network lines are not used to facilitate the initial connection.

ii) Access the network via the closest available node. This is infeasible since each different HOST has restrictions on the terminals allowed access to it. In addition, it introduces the problems of burdening that HOST with all the overhead of terminal handling.

iii) Access the network via a special node designed specifically to allow terminal access to network resources. This approach is being tried at the University of Illinois via their ANTS system (ARPA Network Terminal Service). This system allows a user anywhere in the country to dial-up ANTS and then use ANTS as his terminal handler to connect with any other node in the ARPA Network (DENE72).

iv) Create a special terminal interface which performs all jobs of a regular communications interface but for the special case of terminal access.

Method (iv) is essentially the solution being used in almost all cases where terminal access to the network is allowed. ARPA calls their interface a TIP (Terminal Interface Processor) and integrates it as just an extension of the IMP (ORNS70). CANUNET proposes that the terminal interface be separated from the responsibilities of the communications interface. This terminal handler is referred to as a pseudo-host (CANU72 - Appendix L). CYBERNET has the most flexible design and allows any of i, ii, iii above since terminals can connect to either nodes or centroids themselves (LUTH72).

8. SUMMARY

This work is intended to be an introduction to the growing area of computer networks. A tremendous amount of effort is presently being expended in designing and actually building general purpose computer networks. This effort is going on at all levels - university (TUCC, MERIT, OCTOPUS, DCS), private enterprise (CYBERNET, NETWORK/440, TSS), federal government (ARPA), and foreign governments (CANUNET, NPL, ECN). Their work is helping to begin the long process of the understanding of the problems involved in network design, and their results are stimulating further research into how to optimize the solutions. One cannot help but come away from this area with the realization that networks are here to stay. But as mentioned by D. J. Farber in (FARB72a) the most important question is how can we best utilize them within our present day corporate and academic structures. This is an unsolved problem.

APPENDIX

	ARPA	CANUNET	MERIT	CYBERNET	OCTOPUS	DCS	N/440
Location	USA	Canada	Michigan	USA	Lawrence Radiation Laboratory	UC/Irvine	USA
Organization	Distributed	Distributed	Distributed	Distributed	Distributed	Data Ring	Centralized
Composition	Heterogeneous	Heterogeneous	Heterogeneous	Heterogeneous	Heterogeneous	Heterogeneous	Heterogeneous
Number Nodes	23*	18†	3	36	11	9	13**
HOST Sizes	Varied	Varied	Large 360/67, 6500	Large 6600, 7600	Large 6600, 7600 STAR	Mini-computers Midi-computers	360/44, 67 360/91
Interface Machine	Honeywell DDP 316 DDP 516	?	PDP 11/20 16K	CDC 3300 CDC 6400	PDP 6 PDP 8 PDP 11	Ring Interface	360/91
Protocol	Message Switched	Message Switched	Message Switched	Message Switched	Point-to-Point	Mixed	Message Switched
Transmission Medium	Leased Lines	Leased Lines†	Voice grade dial-up	Voice grade and wideband leased lines	Coaxial Cables	Twisted Pair - Coaxial Cables	Leased
Rates	50Kb	9.6Kb-50Kb†	2Kb	2000b-40.8Kb	12 megabits	2-5 megabits	2.4-40.8Kb
Routing	Adaptive	Adaptive†	Adaptive using automatic calling units	Static	Static	Static (circular)	Static
Message Size	8,095 bits	?	240 characters	1,024 characters	1,208 or 3.78 million bits	900 bits	

** Early, 1971
* December, 1971
† Proposed

BIBLIOGRAPHY

(ANDE71) Anderson, R. D., et. al., "The Data Reconfiguration Service--An Experiment in Adaptable Process/Process Communication", ACM/IEEE 2nd Symposium on Problems in the Optimization of Data Communication Systems, Palo Alto, California, Oct., 1971.

(AUPE72) Aupperle, E., "MERIT Computer Network: Hardware Considerations" in Computer Networks, ed. Randall Rustin, Prentice-Hall Pub. Co., 1972.

(BARB72) Barber, D. L. A., "The European Computer Network Project", Proc. of the First Intl. Conf. on Computer Communications, Wash., D. C., Oct., 1972.

(BART67) Bartlett, K. A., Scantlebury, R. A., Wilkinson, P. T., "A Digital Communication Network for Giving Rapid Response at Remote Terminals", ACM Symp. on Op. Syst. Princ., Gatlinsburg, Tenn., Oct., 1967.

(BEER72) Beere, M. P., Sullivan, N. C., "TYMNET--A Serendipitious Evolution", ACM/IEEE 2nd Symp. on Problems in the Optimization on Data Communication Systems, Palo Alto, California, Oct., 1971.

(BELL69) Bell, C. G., et. al., "Computer Networks", Computer Science Research Review, Computer Science Department, Carnegie-Mellon Univ., 1969.

(BELL72) Bell, C. G., Wulf, W. A., "C.mmp - A Multi-Mini Processor", AFIPS Conference Proceedings, Vol. 41, 1972.

(BREW71) Brewster, R. L., "Optimizing Data Networks", Data Proc. Magazine, Volume 13, Number 1, Jan. 1971.

(BROO68) Brooks, F. P., "The Organizational, Financial, and Political Aspects of a Three University Computer Network", IFIPS Cong. Proc., Edinburgh, Scotland, 1968.

(CANU72) "A Proposition for a Canadian University Computer Network (CANUNET)", prepared by the Univ. of Quebec and the Advisory Committee for the Canadian Dept. of Communications, March, 1972.

(CARR70) Carr, C. S., Crocker, S. D., Cerf, V. G., "HOST-HOST Communication Protocols in the ARPA Network", AFIPS Conf. Proc., SJCC, Vol. 36, 1970.

(COCA72) Cocanower, A., "MERIT Computer Network: Software Considerations", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1972.

(CROC72) Crocker, S. D., et. al., "Functional Oriented Protocols in the ARPA Network", AFIPS Conf. Proc., SJCC, Volume 40, 1972.

(DAVI68) Davies, D. W., "Principles of a Data Communication Network for Computers and Remote Peripherals", IFIPS Congress Proceedings, Edinburgh, Scotland, 1968.

(DAVI71) _____, "Control of Congestion in a Packet Switching Network", ACM/IEEE 2nd Symp. on the Problems in the Optimization of Data Communication Systems. Palo Alto, California, Oct., 1971.

(DENE68) Denes, J. E., "BROOKNET--An Extended Core Storage Oriented Network for Brookhaven National Laboratories", IFIPS Congress Proceedings, Edinburgh, Scotland, 1968.

(DENE72) Denenberg, S. A., "Getting Started on the ARPA Network", CAC Document No. 42, Center for Advanced Computation, University of Illinois, August, 1972.

(DESA73) Desautels, E. J., Chow, V., Schneider, M., "Loosely Coupled Time-Sharing Systems", University of Wisconsin Technical Report, Dept. of Computer Science, (in preparation).

(EISE67) Eisenbies, J., "Digital Communication Conventions", IBM Systems Journal, Volume 6, Number 4, 1967.

(ELIE70) Elie, M., "General Purpose Networks of Computers", M.Sc. Thesis, Dept. of Computer Science, UCLA, 1970.

(FARB72a) Farber, D. J., "Computer Network Survey", Datamation, Volume 18, Number 4, April, 1972.

(FARB72b) _____, "Data Ring Oriented Computer Networks", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1972.

(FARB72c) _____, "Structure of the Distributed Computer System (DCS)", First Intl. Conf. on Computer Communications, Washington, D. C., Oct. 1972.

(HANS71) Hansler, E., "Reliability Considerations in Centralized Computer Networks", ACM/IEEE 2nd Symp. on Problems in the Optimization of Data Communication Systems, Palo Alto, California, Oct. 1971.

(HAYE71) Hayes, J. F., Sherman, D.N. "Traffic and Delay in a Circular Data Network", ACM/IEEE 2nd Symp. on Problems in the Optimization of Data Communication Systems, Palo Alto, California, Oct., 1971.

(HEAR70) Heart, F. E., et. al., "The IMP Processor for the ARPA Computer Network", AFIPS Conf. Proc., SJCC, Volume 36, 1970.

(HERZ72) Herzog, B., "MERIT Computer Network", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1972.

(KAHN71) Kahn, R. E., Crowther, W. R., "Flow Control in Resource Sharing Networks", ACM/IEEE 2nd Symp. on Problems in the Optimization of Data Communication Systems, Palo Alto, California, Oct. 1971.

(KAHN72) _____, "Terminal Access to the ARPA Network", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1970.

(KARP71) Karp, D., Seroussi, S., "A Communications Interface for Computer Networks", ACM/IEEE 2nd Symp. on Problems in the Optimization of Data Communication Systems, Palo Alto, California, Oct., 1971.

(KLEI70) Kleinrock, L., "Analytical and Simulation Methods in Computer Network Design", AFIPS Conf. Proc., SJCC, Volume 36, 1970.

(KLEI72) Kleinrock, L., "Survey of Analytical Methods in Computer Networks", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1972.

(LUTH72) Luther, W. J., "The Conceptual Basis of CYBERNET", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1972.

(MART70) Martin, J., Teleprocessing Network Organization, Prentice-Hall, 1970.

(MART71) _____, Telecommunications and the Computer, Prentice-Hall, 1971.

(MCKA72) McKay, D. B., Karp, D. P., "IBM Computer Network/440", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1972.

(MCQU72) McQuillian, J. D., et. al., "Improvements in the Design of the ARPA Computer Network", AFIPS Conf. Proc., FJCC, Volume 41, 1972.

(MEND72) Mendecino, S. R., "OCTOPUS--The Lawrence Radiation Laboratories Computer Network", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1972.

(MINS70) Minsky, N., Rotbard, I., "CSL--A Language for Job Control", Hebrew University Technical Report, Hebrew University, Oct., 1970.

(ORNS72) Ornstein, S. M., et. al., "The Terminal IMP for the ARPA Computer Network", AFIPS Conf. Proc., SJCC, Volume 40, 1972.

(PARK69) Parker, L. T., et. al., "Introducing Computers to Small Colleges and Universities--A Progress Report", CACM Volume 12, June, 1969.

(ROBE67) Roberts, L. G., "Multiple Computer Networks and Inter-Computer Communications", ACM Symp. on Operating Systems Principles, Gatlinsburg, Tenn., 1967.

(ROBE70) _____, Wessler, B. D., "Computer Network Development to Achieve Resource Sharing", AFIPS Conf. Proc., SJCC, Volume 36, 1970.

(SCAN71) Scantlebury, R. A., Wilkinson, P. T., "Design of a Switching System to Allow Remote Access to Computer Services by Other Computers and Terminal Devices", ACM/IEEE 2nd Symposium on Problems in the Optimization of Data Communication Systems, Palo Alto, Calif., Oct., 1971.

(SCHA70) Schaefer, M., "DBL--A Language for Converting Data Bases", Datamation, Volume 16, Number 6, June, 1970.

(SPIE69) Spier, M. J., Organick, E. I., "MULTICS Interprocess Communication Facility", ACM 2nd Symp. on Operating Systems Principals, Princeton, N. J., Oct. 1969.

(SPRA71) Spragins, J. D., "Analysis of Loop Transmission Systems", ACM/IEEE 2nd Symp. on Problems in the Optimization of Data Communication Systems, Palo Alto, Calif., Oct., 1971.

(WALD72) Walden, D., "A System for Interprocess Communications in a Resource Sharing Computer Network", CACM, Volume 15, Number 4, April, 1972.

(WEIS72) Weis, A. H., "Distributed Network Activity", in Computer Networks, ed. Randall Rustin, Prentice-Hall, 1972.

BIBLIOGRAPHIC DATA SHEET	1. Report No. WIS-CS-177-73	2.	3. Recipient's Accession No.
4. Title and Subtitle A Survey Report on Computer Networks			5. Report Date March 1973 6.
7. Author(s) Michael Schneider			8. Performing Organization Rept. No. 177
9. Performing Organization Name and Address The University of Wisconsin Computer Sciences Department 1210 West Dayton Street, Madison, Wisconsin 53706			10. Project/Task/Work Unit No. 11. Contract/Grant No.
12. Sponsoring Organization Name and Address			13. Type of Report & Period Covered 14.

15. Supplementary Notes

16. Abstracts

This paper first defines what is meant by a computer network and then looks at the useful user services that can be provided through such a network. It then reviews the design alternatives, both logical and physical, that are available to a network researcher. Six actual networks--ARPA, CANUNET, MERI MERIT, CYBERNET, OCTOPUS, and DCS--are briefly discussed and compared. Finally, the most current research problems are outlined and some possible approaches taken by the designers of the above networks are contrasted.

17. Key Words and Document Analysis. 17a. Descriptors

Computer networks
Data communications

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 76
	20. Security Class (This Page UNCLASSIFIED	22. Price