The Changing Nature of Computer Networks

David D. Clark

M.I.T. Laboratory for Computer Science

Draft of 6 June 1990

1. Introduction

In the last twenty years, computer networks have progressed from experimental uncertainty to commercial success. Enabled by early development of key technical concepts, and driven by the revolution in personal and distributed computing, networks now span the country and connect most of the computers in operation today.

Indeed, the apparent maturity of computer networking may suggest that there is little remaining in innovation and evolution. The opposite is true. For a number of reasons we can expect the nature of computer networking to change in basic ways over the next decade. The goal of this paper is to review this pressure for change, and to make some predictions about the future of data networking. However, in order to understand the directions of the future, it is necessary to examine some of the past.

2. A short history of computer networks

Computer networks first came into existence at a time when large, time-shared central computers were the normal pattern of computing. The communications problem was thus remote terminal access. The first approach to providing this service was to take advantage of the existing voice-oriented telephone network. By dialing up a telephone line and connecting a modem, access from a terminal to a computer could easily be established. There were, however, several problems with this solution. First, the resulting costs could be very high, especially for long distance. Since the connection was maintained even while the user was not typing, the telephone connection was not used efficiently. Second, the achievable data rates were very low, both as a consequence of the cost and of the available modem technology.

To address these problems, and to provide a better form of communications for computers, a key concept was proposed, which today forms the basis of essentially all computer networking. Since the traffic patterns of remote login were very bursty, with periods of typing and output intermixed with periods of thought, the proposal was to share one telephone connection among several distinct login sessions. A mini-computer, called a *switch*, was connected to the phone line, and the users connected to the switch. When data was generated by a user, it was received and buffered by the switch. The data from the various users was then sent in turn down the line, each fragment of user data preceded by an identifier indicating the identity of the user.

This assembly of data and user identifier was called a *packet*, and the concept was called *packet switching*. Since the packets of several users could share one connection, the costs for the connection were reduced accordingly. If several users needed to transmit at once, the switch buffered the data until the link was free. This might inject delay into the session, but it was considered an acceptable consequence of the overall scheme.

Packet switching is the key concept which has permitted successful and effective computer networking. The interleaving of bursty traffic from multiple sources has controlled the communications costs, and very high-speed transmission can be provided to meet the peak needs of each user. There is no question that packet switching has been very successful. However, we need to understand the limits of the concept to see how networking may evolve in the future. There are limitations to packet switching, limits that may become more obvious as networking evolves.

This very brief introduction to the concept of packet switching will provide enough background for the rest of this paper. The reader desiring a detailed introduction to the current state of networking is referred to the books by Tanenbaum[1] and by Bertsekas and Gallager[2].

3. The fundamental role of networking

As described above, packet switching was developed to meet an economic need. The economic problem was very real; telephone connections were (and are) expensive. However, during the lifetime of packet switching, the economic pressures on the network have changed drastically. Packet switched networking was first proposed when the primary problem was remote terminal access. Since then, remote terminal access has been replaced by the personal computer using remote disk access, which was in turn replaced by distributed disks and remote *file* access, and now with supercomputers we are once again entering the era of remote terminal access. These changes totally alter the traffic patterns, degree of burstiness and peak traffic rates seen on our networks. At the same time, the economics of transmission have changed drastically, with local networks in particular representing a situation in which transmission costs are not a major part of the overall budget.

While the central concept of packet switching has proved very robust in the face of these changes, many products and developments that were based on particular assumptions about economic costs and requirements have faded from the scene in the face of changing patterns of computing and communications. One must be careful of mechanism justified by economics, for the preferred solution dictated by economics can change very rapidly with changes in technology and policy.

To understand the real nature of computer networking, one must look beyond the economics and technology of the moment, and see what fundamental requirements networks fulfill. I believe that there are three central issues around which networking revolves. People are distributed, information is distributed, and we need to build on the work of others. These are fundamental, and not to be sidestepped or revised by innovation or cost-reductions.

It may seem superficial, even frivolous, to observe that people are not all in the same place. But if one seeks the most significant application of computer networking in the last two decades, one is drawn not to remote login or remote disk access but to electronic mail. Electronic mail, which has little to do with distributed computing but everything to do with distributed people, is a fundamental enhancement to the options for human communication. As used today, it provides the informality and timeliness of a phone call without the pre-emptive nature of the phone and without the endless exchange of phone messages. The informality follows from the rapid delivery, which permits an exchange more like a conversation than a formal correspondence. However, since the message can be held for delivery if the receiver is not immediately available, it permits communication without both parties being available simultaneously.

Most users of electronic mail are convinced of its utility, and easily converted to enthusiastic proponents. This is not a comment about computers, but about people, and the acceptance or rejection of electronic mail relates to matters of human behavior, not computing technology. This application thus illustrates the fundamental issue of computer networks. It is easy and tempting to think of networks as hooking computers together. They are better thought of as hooking people together, with a computer mediating the connection in some effective way. Whether the application is electronic mail, or access to remote information, the motivation for the communication is human need, not internals of computer system design.

4. Some examples from the Internet project

Over the last fifteen years, a distributed group of researchers has defined the protocols for a large network called the Internet, and has used this network as a tool to support their own efforts. The participants come from across the United States as well as Europe, and tools such as electronic mail were crucial to span the distance and different time zones. In addition to mail, there were other network applications that tied the group together and facilitated its efforts. Here are two examples of tools from the Internet project.

On-line publishing -- In support of the group research, a library of working papers was maintained on line, distributed and circulated over the network. Over 1100 documents have been published so far. The style of publication is informal, geared to quick dissemination of information. The documents are not refereed, but there is an editor for the series and they are usually reviewed for content by a relevant working group of the research community. Even so, a document can often be published in as little as a week. This ability for quick circulation of working papers is essential to tie together a distributed working group.

Teleconferencing -- Recently, the Internet research team has experimented with technology for teleconferencing, using the network both for a video and audio connection to view the remote site, and to provide a shared view of a common on-line work space. Although the technology is experimental, and the video quality is low, the acceptance of the facility by the group has been very high. When the facility was decommissioned briefly, there was loud protest, even though it had not been promised as an operational facility.

These two examples illustrate the idea that the components tied together by the network were not computers but information and people. These are examples of the fundamental role of networks, They do not derive from the economics of telephone links or computer components, and they serve as a basis for a durable vision of a network of the future.

5. The layered abstractions of networking.

The packet is the key concept of computer networking as it is normally practiced today. While the packet has proved very useful as a unit of multiplexing, it does not make a good application interface to a network. The requirements of multiplexing cause the packet to be of rather small maximum size, and during transit of the network it can become lost, corrupted, reordered or duplicated. The application builder would prefer to see a more abstract view of the network, with some of the rough edges of the packet smoothed over. An important part of network design is thus the development of abstract models of networking, which simultaneously serve the application builder and fit well with the underlying concepts such as packet switching.

To understand the importance of abstraction, it is helpful to consider another very familiar example of interface abstraction, the succession of proposed abstractions for the disk.

The disk block is not as intractable a building block as the packet; it does not often get lost, reordered or corrupted while on the disk. None the less, it is a small, fixed size unit, which does not fit well with the needs of most applications, so a series of abstractions have been proposed for the disk, with the goal of making it easier to utilize.

The first was to combine a number of fixed size blocks to make a variable size element, the *file*. This idea was older than the disk, since it dates from the era of tape. The next abstraction was to take these files and name them, which eventually led to the hierarchical file system. Another fork in the evolution of abstraction was *virtual memory*, which uses the disk to give the illusion of increased primary memory. One operating system built at L.C.S., the Multics system, combined both of these abstractions, so that the application builder could view the disk as a file system and virtual memory simultaneously.

These abstractions, memory or files, model the disk by relating it to some previous system facility. As the disk became more mature, the next stage of development is abstractions that are new, not a repackaging of previous mechanisms but specific to the particular features of the disk. The database is a good example of an abstraction specific to the nature of the disk. And a current proposal is a multi-dimensional data representation such as the hypercard concept.

There are two points to be made about these examples. First, only after we gain familiarity with the new technology do we begin to construct abstractions specific to the technology and its features. The first round of abstractions are almost always based on ideas originally proposed for some other context. Second, the process of exploring and defining disk abstractions has not yet converged, even though the disk has been around for much longer than the network.

Since the network is a relatively young technology, we can expect that the abstractions employed for it will most likely be based on concepts first exploited for some other circumstance. We are just now at the point in the maturation of networks where we are beginning to see proposals based on the particular and novel aspects of networks. Many of the common abstractions used for networks today are perhaps a bit limited, exactly because they were first proposed for some other circumstance.

5.1. The Virtual circuit

The *virtual circuit,* as the name might suggest, attempts to model the network as a wire. This indeed is a comforting abstraction for an application which previously operated over a dedicated communications link and is now being moved onto a network. In the virtual circuit abstraction, the network is modeled as a reliable bi-directional byte stream. Even though the data is actually transported in packets, with all the failures that packets can suffer, at a higher level that data appears to be a reliable sequence of bytes. By numbering the packets, and using these numbers to detect and retransmit lost packets, as well as to sequence misordered packets, the software supporting the virtual circuit abstraction removes from the application any necessity for dealing with these failure modes.

What is the penalty of this abstraction? Its limitation can best be illustrated by exploring an application for which it is ill-suited, real-time digitized speech, in other words telephony based on packet switching.

In this application, the speech waveform is digitized in real time, placed in packets, and transmitted to the receiver (the listener) to be replayed. Since the packets represent a continuous data source (the speaker), they must be delivered rapidly, and without excessive jitter. If a packet is excessively delayed, it will not be available at the time that the receiver requires the information, and the listener will perceive the speech flow as having been disrupted.

What happens if a virtual circuit abstraction is used for speech, and a packet is lost? The abstraction insists on delivering all the data in order. It will hold up the delivery of all packets subsequent to the lost packet until, using requests back to the sender, the lost packet is retransmitted. But this process may make all the subsequent packets unacceptably late, not just the lost packets, totally disrupting the flow.

A much better alternative would be to sacrifice some reliability for better control of delay. In the case of speech, the application can deal with a lost packet, even though it cannot cope well at all with excessive delay. The application can just replace the missing data with noise, or with a replay of the previous packet. This causes a brief audible glitch, but would not usually render the communication unintelligible. If it does, even higher level error recovery can be employed; the listener can ask the speaker to repeat the damaged phrase.

The problem is thus that the virtual circuit abstraction, by insisting on perfect reliability, sacrifices flexibility in other basic service parameters of the network such as delay, parameters that may be critical to specific applications.

5.2. The Remote Procedure Call

Another network abstraction is the remote procedure call. This abstraction is based on a previous idea not related to any form of remote communication, the procedure call. In this abstraction, the distributed computing environment is modeled as a set of procedures or subroutines local to each machine. One machine communicates with another by calling one of the routines residing on the remote machine. By using the paradigm of object oriented programming, one can make remote information available by calling the subroutine that manages that information.

This abstraction is useful because the semantics of a subroutine call is very well understood. Application builders can thus build correct programs by using very mature reasoning processes. The problem with the abstraction is that, precisely because it is based on an idea from another context, it does not match some of the actual features of real networks. For example, when the network or the remote machine fails during a subroutine call, the calling subroutine may not be able to tell if the subroutine has completed. An attempt to repeat the subroutine call later after the failure has been corrected might thus lead to the call being executed twice.

A more serious problem with the remote procedure call as a network abstraction is that it does not deal well with the real delays that are present across networks. Subroutine calls are strictly serial; a second call cannot commence until the first has returned. Further, the caller and the called routine cannot execute in parallel. So the computation executes in a series of strict lock-step exchanges of control between the two machines. The real time delay between the two machines thus limits the rate at which exchanges can occur. For example, a round trip delay of 30 ms. limits a computation to about 30 remote procedure calls per second. This is several orders of magnitude slower than one would normally expect of a local procedure call; if an application based on local procedure calls is mapped onto remote calls, this severe performance degradation may render the application essentially inoperative.

The feature of networks that causes problems here, and with the virtual circuit as well, is that networks have substantial and highly variable delays. Unless a network is restricted to a small geographic scope, such as a room, the delays due to the propagation of the data along the communication links can be a major consideration. In addition, since the network is a shared facility, as discussed above, additional delay may arise while waiting for an occupied link to become free. These delays are fundamental, and cannot easily be masked, most especially by an abstraction such as a procedure call which is totally synchronous by its nature. In this respect, the procedure call seems a rather poor model of a network.

In an attempt to overcome this problem, and provide a network abstraction that deals well with delay of various sorts, the Mercury project at L.C.S. has been exploring the idea of *stream calls* [cite Barbara's paper], in which one call can begin before the previous has completed. The goal of this rather more complex abstraction is to capture at least some of the semantics of the procedure call while dealing effectively with the real delays found in real networks.

6. Networking Today

We are now at a point to summarize the present state of networking, so that we can begin to reason about the changes that are to come. First, while networking is very successful, it is still very young. It is still using abstractions based on previous technology, abstractions which even today can be seen to have substantial limitations. Second, it has been driven to a considerable extent by reasons of economics, which, while they are real, are transient. The most exciting applications of networks, such as electronic mail, do indeed derive from fundamental concerns, the distribution of people and information.

Third, the packet, that unit of multiplexing so critical to network design, has taken on a very large role in the network architecture of today. In constructing abstractions such as the virtual circuit, there are several control problems that must be solved: *flow control*, insuring that the sender does not transmit faster than the receiver can process the information, *error recovery*, in which delivered information is acknowledged and lost information is retransmitted, and *congestion control*, in which the sender is prevented from overloading the network. In the designs of today, the packet has become the element around which all of these controls are achieved. The importance of this will become more obvious as I consider the future role of the packet.

My conclusion, which I will attempt to defend in the remander of this paper, is that the designs of today, heavily based on the packet as a unit of control, and the abstractions of today, still borrowed from other contexts, will not carry us into the next generation of networks. That next generation is about to burst upon us, and we must rethink some of our basic assumptions if we are to succeed in building the networks of tomorrow.

7. The Shape of Tomorrow

Our view of networking is based on the assumption that we are about to experience a great change in capability and function. So we must begin by describing and defending our vision of networks. The central change is that networks will get faster in speed and larger in size. They will be built using much faster trunks, in the multi-gigabit range. More significantly, they will support much faster flows at the application level, with speeds up to a gigabit or more. In size, they will expand from the largest data networks of today, which have perhaps 100,000 end points, to match the phone system in size, with several hundred million end points. In other words, they will grow, both in size and in speed by perhaps three orders of magnitude. This prediction, if true, suggests a general reason for concern. Other computer system artifacts, such as operating systems, have seldom survived the attempt to scale them by one order of magnitude. Experience gives us little hope that we can scale a network by three orders of magnitude both in speed and size without a complete re-examination of the fundamental architectural assumptions.

Why do I predict this growth? There must both be the possibility and the need. The possibility is technological, based on two advances, the fiber optic link and the VLSI chip.

The role of fiber optics is obvious. Not only does it permit greater transmission speeds, but it does this at greatly reduced cost, which is perhaps more important. Fiber optic links permit a change in the economic operating point of communication links. As a consequence of the fiber installations currently underway or planned, we can expect the impact of fiber transmission to be realized over the next decade. Indeed, over the next ten years fiber links may become abundant.

VLSI is the key that permits us to manage these links and to use them to build networks. Networks require that we provide a switching function to operate at the speeds of the links. As links get faster, the switch must similarly increase in speed. Today, packet switches are built using general purpose processor chips and memory architectures. Switches thus assembled are already proving a bottleneck with today's link speeds. To deal with the much higher speeds that fiber will permit, it will be necessary to develop special purpose switch elements. Without VLSI, this would not be practical. But today we can construct highly specialized switch components at reasonable cost.

Just knowing that such speeds are possible does not mean that they will come to be. Unless there is a need for these new capabilities, there will not be a sufficient pressure to drive the redesign necessary to achieve these speeds. This pressure must come from new applications, which will generate the demand. It is thus worthwhile to speculate on how applications and their communications requirements might evolve over the next decade.

There is a current trend in applications which drives the future of networking as much as does fiber optics. This trend is the increasing use of graphics, scanned images, video and other visual forms as part of the application interface. Computers have been dealing with images for many years now, in medical systems, cartography, and publishing, to give only few examples. In the past, these applications have been viewed as outside the mainstream of computers, requiring special hardware and systems. Now, however, the image is about to enter the more general context of computing. The term *multi-media workstation* is often used to describe the next generation of user interface, a workstation that combines traditional computing with visual and audio capabilities. Many prototypes of this concept exist now, and commercial products are beginning to enter the marketplace.

What is the value of images in applications? The following examples, based to some extent on experience with the Internet project, show a range of possibilities.

7.1. Multi-media documents

The current popularity of facsimile machines derives to some extent from the fact that any form of document can be transmitted: text, chart, picture etc. Computer-based document preparation tools are only now beginning to permit the construction of documents that incorporate all of these modes. The marriage of computers to video will permit even more exciting forms of documents, in which fragments of video are included as illustrations in the document. Imagine a scientific paper in which video is used to record the experiment performed.

The existence of these multi-media documents will provide the basis for a number of document management tools. For example, it will be necessary to support document browsing, and searches based on content. Browsing of multi-media documents is not complex, it just requires bandwidth. But content searches of a multi-media document is a very challenging problem, since we have very few tools for expressing a search criterion for media other than text.

7.2. Desktop teleconferencing

The experience of the Internet community gives first-hand evidence of the appeal of teleconferencing technology. One obvious extension is to attempt to permit a teleconference from the office. While earlier technology such as the Picturephone may suggest problems with desktop video interaction, I suspect that the concept can be successful if it permits the video equivalent of a conference call, because experience with the Internet facilities suggests that only when there are several participants does the value of the video link become obvious. If desktop teleconferencing were to prove useful, it would generate a tremendous demand for switched multi-site video technology, which should be integrated into the workstation to permit the conference to display and manage both the view of the distant participants and some shared distributed workspace.

7.3. Multi-player interactive games

A consumer-oriented application is the multi-player game. An informal observation is that games against other human players are more challenging and interesting than a game against a computer. Imagine a hybrid of today's video games which permit several players, each at a separate screen, to participate in some joint contest, either of dexterity or strategy. This concept, if successful, could define a new market with a potential equal to the home video games today together with a network capability to hook them all together.

A related example is distributed gaming as a training tool. The U.S. Army is using such a tool today as a training aid for tank operators. The different tank drivers see a view on their display corresponding to the view out of their tank, and can interact with each other, joining up with friendly forces and shooting at the enemy. All such actions have a realistic consequence on the displays.

7.4. Visual user interfaces

In making a computer easy to use, a graphic or visual interface often seems effective. Current computer systems portray and organize information using visual cues such as file cards, date books and similar indicators. These are intended to make it easy for the user to understand and classify the information being presented. Control information appears in windows that pop up, out, or otherwise attempt to capture their organization in their imagery. This trend seems successful, and its use should expand in the future. As more powerful tools for picture management become available, more powerful user interfaces should quickly follow.

One potential example would be an interactive "help" facility that can provide fragments of prerecorded video instruction, or failing that a video connection to a human consultant, all of which could be archived for later review.

7.5. Computer aided image creation

Perhaps the most important aspect of visual interaction is the computer construction and display of images. A very important example of this arises today in the use of supercomputers. Supercomputers are typically used today for simulation, the results of which need to be presented to the user. The supercomputer community has adopted the word "visualization" for the concept that such data should be displayed in moving image form, rather than as lists of numbers. There seems to be a consensus that animated output of simulation results is significantly more effective than other forms of output.

Indeed, if this were a multi-media document, it would be possible to include fragments of video that would demonstrate instantly the value of this sort of output. Sadly, I can only suggest how compelling it would be to see such a video.

Once the idea of visualization is recognized, it is obvious that there are interesting images that do not require a supercomputer to create. For example, a tool for constructing images could be of great use to professionals such as architects who today use images to convey their concepts. These images are traditionally created "by hand", sometimes with great difficulty, and once created are static. A computer created image could be created, placed in a context, or modified, all in support of discussion among humans.

Fields such as medicine are already making heavy use of computer-generated images, with some of the more complex and sophisticated scanning tools for diagnosis. Indeed, medical images are among the highest resolution images being processed today.

The use of images in medicine provides a good example of the value of integrating several of the above examples. Once an image, perhaps a moving image, has been created, it might be desirable to have several doctors, perhaps at distant locations, confer about that image. That is, what is needed is a teleconference in which the shared workspace itself is a computer-generated moving image.

The purpose of this extended discussion of applications is to generate some sense of need for high bandwidth digital communications. The reader may find one or another of these examples particularly compelling, or indeed may believe that the need for networks lies in some other direction altogether. If our experience with past technical innovations is any guide, all predictions of its utility will be wrong. Some new application will arise unexpected, such as electronic mail on the last generation of networks, or the spreadsheet programs on the personal computer, and will shape and justify the development of the technology. We must have some vision of the future to motivate us to build the enabling technology, but it is almost certainly off the mark.

8. How to Get Fast

To this point I have argued the following:

- Technology exists to permit the development of networks much faster than those today.
- There are potentially new and exciting applications that could be built using this technology.

So what does it take to get there? Can we just turn up the clock on the network, and be done? Sadly, it is not as easy as that. There are some very hard technical problems to be solved if we are to build this next generation network, problems that arise from fundamental issues of speed and scale. In this last part of this paper, I will review the source of these problems, and give a few specific examples of problems to be solved.

The essence of the problem is that not all the components of the network are getting faster at the same rate. New fiber optic transmission technology has caused the raw bandwidth to get quite a bit faster. VLSI has permitted the construction of faster switches, but switching has not taken the same step in speed as transmission. The attached hosts, which support the applications the network is to serve, are getting somewhat faster, for example by means of RISC processors, but still have some of the same performance bottlenecks of bus and memory. And the speed of light, which relates to network performance in a fundamental way, has not changed at all.

A simple example will show the sort of problem that arises. A cross-country network has a round-trip latency of about 30 ms., because of the speed of light. If the network is increased from 1 Mbps to 1 Gbps, the amount of data in the channel at any one time increases from 4 KBytes to 4 MBytes. The control algorithms that are devised to regulate the flow of information into the network must thus be designed to manage three orders more data.

8.1. New control algorithms

The term *congestion control* describes the mechanisms used to regulate network traffic in the case that the users are collectively exceeding the capacity of the network. Today, users send data into the network and receive feedback if they are overloading it. As networks get faster, feedback becomes less effective. The problem is that for even large data elements, a transmission at these speeds may well be complete before feedback can come into play. Feedback cannot be effective unless the total amount of data being transmitted is substantially larger than the amount in transit across the network at any instant (or at least larger than the share of that which the user can expect to claim). Unless we believe that applications will scale up the size of their transmissions to match the increased data rates, we must presume the need for new models of congestion control, which are based more on precomputed expectations and open-loop controls.

8.2. Packet processing overhead

Simple arithmetic will show the potential for a performance bottleneck in the switch. Measurements of typical local area networks today show packet sizes around 200 bytes. To make things better, I will assume that packets will get larger over time, perhaps reaching 1000 bytes.

Assuming this size, a 1 Gbps link carries 250,000 packets per second, and a 10 Gbps link carries 2.5 million packets per second. Current packet switches based on mini-computers can forward between 1000 and 10,000 packets per second. That is, we are off by several orders in speed. While there are steps that can be taken to make the processing faster, there are also forces that are making the problem worse. The result is a real concern that switching will represent a bottleneck for the networks of tomorrow.

There are several obvious ways to make the switch faster. A larger packet size would help, since it would reduce the packet rate for a given bit rate. But the packet size is related to application requirements. While some applications might be able to use very large packets, in many cases the packet size is application determined.

In fact, rather than getting larger, the packets are getting smaller. A current proposal from the telephone switching community is for a single fixed size multiplexing unit suited for both voice and data. This concept, called Asynchronous Transfer Mode, or ATM, is based on a suggested packet size of 48 data bytes. So while an order or two of speedup is needed, this proposal takes away perhaps an order.

One could abandon the mini-computer as a base for a switch, and use specialized chips and architectures. However, even the fastest specialized processing elements proposed for switches have trouble keeping up with the above rates. One could propose a highly parallel switch, and indeed parallel designs proposed in the context of ATM seem quite promising. But the designs proposed today require a great simplification in the processing complexity to achieve the needed rates.

As the speed requirements force a simplification of the processing task to control overhead, at the same time there is pressure for more processing in the switch. If we are to build networks that control and account for their usage, it will be necessary to log network traffic, provide access controls, and so on. These policy checks could easily cost another order of magnitude in speed if designed in a naive way.

In order to deal simultaneously with the need for decreased overhead and increased processing complexity, a change is needed in the way processing is structured. In many networks today, often called "connectionless" networks, each packet is processed in isolation, without regard to previous and subsequent packets. This approach must be changed so that when a packet is processed, the results are cached and applied to the similar packets that may follow. This caching provides a way to link together a sequence of packets for management purposes. Our research group has used the term *flow* to describe such a sequence.

8.3. Host interfacing

The overhead of processing is even more limiting in the host, where the protocols necessary to implement the higher level network abstraction, e.g. the virtual circuit, are implemented. There is significant complexity in implementing such protocols. At the same time, the programs executing these protocols must run in the context of a general purpose operating system, rather than in the specialized programming environment of a packet switch.

Even with the networks of today, host software is perceived as a bottleneck to utilizing the raw network bandwidth. If networks speed up by several orders, what can the host do to keep up?

There seem to be two possible answers. One is to rethink the relevant parts of the host environment: operating system, I/O architecture and memory architecture, so that the host can execute protocols at the needed speeds. There is some evidence that this sort of redesign can be successful. However, depending on the exact nature of the starting environment, the magnitude of the effort may be considerable. An alternative is to move the processing out of the host execution environment and onto some outboard network controller, which embodies a special memory architecture and program execution environment. The goal of this outboard processing would be to permit the host to deal in units of data larger than the multiplexing elements, in particular data units related to application needs. Thus, for example, a file transfer application might deal in units of a disk block, while the outboard network controller would fragment this into packets, or into 48 byte ATM multiplexing units.

8.4. Network dynamics

Large networks are complex dynamic systems, which unless controlled can display undesirable behavior. These effects are visible in networks today, and they can only get worse with increased size and speed. An example of a current problem may help to illustrate the tendency.

Both observation of real networks and simulation in our group suggest that queue length and (in consequence) perceived delay across the network are varying periodically. In other words, the network is oscillating. The cause of this is conjectured to be a synchronization among the users of the network, leading to large bursts of offered load. The network becomes overloaded and looses some packets. This causes all of the affected hosts to time out and retransmit. Since they use the same timeout algorithm, they retransmit at about the same time. This causes another network overload, with more lost packets, and the pattern repeats.

If this analysis of the problem is valid, then to correct the problem it will be necessary to change the control algorithms of the network to smooth out and regulate offered load. This is particularly true as the network become faster. Since the speed of light does not change, for a network of a certain physical scope, the amount of data outstanding in the network goes up proportionally to the speed of the links.

The term *flow control* is used to describe the tools used to regulate the sending of traffic. Today flow control is achieved using the concept of *windows*, which control flow by regulating the number of packets that may be outstanding in the network at any time. That is, in order to fully utilize a link, one must "open the window" enough to fill the path from sender to receiver. As the speed goes up, so does the window size. This need to increase the window size reduces the opportunity for control which the window mechanism affords.

For this reason, our group is exploring an alternative in which flows are regulated by bounding the *rate* at which packets may enter the network, rather than the total number. This may permit the stable regulation of the large amounts of data that must be permitted to enter the high speed nets of tomorrow. One example of this work is reported in the thesis by Zhang[3].

8.5. Policy routing

Policy routing is an example of a problem that does not directly relate to increases in speed, but rather to increases in scale. As networks grow in size, they will be decomposed into parts that are managed by separate organizations. Like the telephone network after divestiture, there will be campus networks, regional networks, long haul (cross country) networks, private interconnect facilities, and so on.

Each of these parts will have its own rules as to what community it serves, how its costs are recovered, and so on. In this complex of interconnected facilities, a user will need to route a packet in such a way that it only attempts to use permitted facilities, that the costs are minimized, and the best service is obtained for the cost.

In order to achieve these goals, new tools to control routing will be needed. In most computer networks today, routing algorithms are designed to meet a simple and well defined goal, the minimization of some cost such as delay. A single-variable minimization will not provide the richness of function needed to meet the above needs. A new sort of routing function, often called *policy routing*, must be architected, to control traffic flow at the level at which resource control policies are established. This function must route traffic so that local policy concerns are satisfied, and so that accounting and billing can be performed.

9. The future of the packet

As was discussed earlier in the paper, the packet as a multiplexing element was a key idea, perhaps *the* key idea, which enabled the successful development of computer networks. It was initially the unit of multiplexing, but also came to be the unit of flow control (e.g., in window flow control systems), and the unit of error recovery (e.g., after buffer overflows caused by congestion).

In the future, this complex role for the packet may change. Because of processing overhead, the unit of control can no longer equal the unit of multiplexing. Especially in the ATM case, where the unit of multiplexing is 48 bytes, processing, e.g. flow control, error recovery, policy routing or cost accounting, will have to be applied to a higher level unit of data. At the host interface, this will be some unit of data that matches application requirements. In general, there will be any number of ways of achieving the needed bandwidth multiplexing. In addition to the ATM multiplexing, there may be traditional packets, or more advanced concepts such as wavelength division of a fiber link. Network designs of the future must be able to deal with any intermixing of these, if they are to have the necessary generality.

What is needed is a new layering and modularization of network structure. Some new abstraction will be needed, one that is above the low level function of bandwidth multiplexing, but that lies below the traditional application abstraction such as the virtual circuit. This new abstraction will provide the necessary grouping to permit network management and control. In fact, very different groupings may be needed for distinct functions such as error recovery and flow control. For a specific proposal for the re-organization of network protocols, see the paper by Clark and Tennenhouse[4].

10. Conclusions

The advent of fiber optics is a one-time phenomenon. The telecommunications infrastructure of the United States is being reconstructed at the present time, and once installed will serve as the communications medium for the next several decades. There is an opportunity now, first to demonstrate the need for high-bandwidth, large scale networking of computers, and second to demonstrate the key concepts by which this goal can be achieved. New abstractions and new control concepts will be needed.

By seizing this opportunity, we can insure that the fiber now being installed is designed in such a way as to facilitate this sort of transmission. If we succeed, the change in computer communication will be so significant as to seem, not an evolution of the present facilities, but a revolutionary change in capability. Our goal should be nothing less than to expand the options for human interaction at a distance, for the fundamental role of networks is to provide new opportunities for building the global community.

References

1. Tanenbaum, A. S., *Computer Networks*, Prentice Hall, 1988.
2. Bertsekas, D, and R. Gallager, *Data Networks*, Prentice Hall, 1987.
3. Zhang, L., ''A New Architecture for Packet Switching Network Protocols'', TR 455, M.I.T. Laboratory for Computer Science, August 1989.
4. Clark, D. D. and Tennenhouse, D. L., ''Architectural Considerations for a New Generation of Protocols'', *SigComm 90 Symposium*, ACM, Sept. 1990.

Table of Contents