Towards a Framework for High-Speed Communication in a Heterogeneous Networking Environment

Gurudatta M. Parulkar
Jonathan S. Turner

In this article we attempt to formulate a framework for high speed communication in an environment comprising a mix of subnetworks with widely varying characteristics. Recent work on high-speed wide area packet switching systems is expected to lead to the development of large public networks capable of supporting applications ranging from low-speed data to voice, high-speed data and video. If such networks are to realize their full potential, they must be designed to operate in an environment that includes networks with widely varying characteristics. Since the early '70s, much of the work on computer communication has been directed toward the development of protocols that allow interworking among computers, operating systems and communication subnetworks of different types. These efforts have culminated in the Advanced Research Projects Agency (ARPA) Internet Protocol Suite, which has introduced a number of ideas of fundamental importance.

Since the development of the internet protocols, the technological context in which we find ourselves has changed dramatically. The development of high-speed Local Area Networks (LANs) and workstations, and the growing role of supercomputers in scientific computing, have led to new and largely unfulfilled requirements for high-speed computer communication. These needs have been difficult to satisfy for a combination of reasons. First, existing wide-area computer networks have been unable to support the data rates required, and, second, the existing end-to-end protocols and host computers are unable to deliver the data to the application at those rates. These are due to the common problem of lack of control and data separation.

On the other hand, fiber optic transmission systems are being introduced rapidly into the national communications infrastructure offering vast amounts of bandwidth at fairly modest costs. Several research groups at industrial and academic laboratories around the world have demonstrated that new high-speed packet switching techniques can make these resources available in a flexible fashion, but up to now these groups have failed to consider the need to operate in a complex networking environment consisting of autonomous and/or technologically dissimilar subnetworks. We feel that it is important to recognize that this kind of heterogeneous environment is here to stay, and if we are to make the best possible use of new developments in networking, we need to establish a framework that supports such diversity.

In this article we attempt to address these issues. We first provide some background on both the current internet model and high speed packet switching. We then outline the major elements of an extended internet model that allows interworking of new high-speed packet networks with a wide range of other networks, including current data networks and national telephone networks. Finally, we discuss some end-to-end and host interface issues.

Background

The ARPA Internet Model

Internet Components

ARPAnet started out as a single homogeneous packet switching network which was hoped to connect one computer from every major research and academic institute. With the proliferation of LANs, it soon evolved into a backbone network essentially connecting campus LANs from these institutes. ARPAnet has evolved further into what is called the ARPA internet as more and more organizations are discovering the benefits of computer networking. ARPA internet today comprises the following:

- A few backbone networks such as ARPAnet itself, MILnet, NSFnet, BITnet, and others. Typically, link speeds for these backbone networks are 56 kb/s or 1.5 Mb/s.
- A number of regional networks such as MERIT (the Michigan State University network), MIDnet (a network of midwestern universities), NYSERnet (New York) and others. A number of these regional networks are sponsored by NSF as a part of NSFnet activity. Typically, link speeds for these networks are also 56 kb/s or 1.5 Mb/s.
- A large number of campus networks at various participating institutes. Such campus networks typically comprise a few LANs such as ethernet and token ring networks. Typical speeds for campus networks and LANs is from a few Mb/s to 80 Mb/s.

All these diverse networks are interconnected in a complex internet using gateways with varying capabilities.

Internet Protocol Hierarchy

The internet uses a protocol hierarchy which is popularly known as the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite [1] [2]. This hierarchy essentially

0890-8044/90/0003-0019 $01.00 © 1990 IEEE

consists of four levels of protocols: application, transport, internet, and network level. It is important to note that this protocol structure uses the same levels of protocol processing for data and control paths. In other words, there is no separation of data and control path. At the application level in this hierarchy, the three most commonly used applications are TELNET (remote login protocol), File Transfer Protocol (FTP), and Simple Mail Transfer Protocol (SMTP). A variety of other applications including voice and multimedia mail have been developed for the internet but are not widely used. At the transport layer, there are both a datagram, User Datagram Protocol (UDP), and a virtual connection-oriented interface, Transmission Control Protocol (TCP). Most of the applications use the connection oriented interface of TCP that ensures reliable delivery of user packets in sequence with no duplicates.

At the internet level, there is only one protocol, appropriately called the Internet Protocol (IP). IP is a datagram oriented protocol and does not make any guarantees—it can lose, duplicate and resequence packets. IP sits on top of a variety of network protocols but gives an impression to its upper layers that there exists only one homogeneous network.

At the network level, there are all kinds of networks in this internet as mentioned earlier, and they have their own network access protocols. The expectations of the internet from its component subnets are modest: the subnet should try its best to forward datagrams toward their final destinations, but it is acceptable for the network to lose, resequence, and duplicate datagrams.

Internet Strengths and Weaknesses

The success and importance of the internet is indicated by its explosive growth and its membership. The internet has been the center of most of the computer networking research done in the United States and has been the focus of many fundamental contributions. Some of the major strengths of the internet model can be summarized as follows:

- It allows interworking among a truly diverse collection of computer and subnet types. For example, hosts on the internet include low-end personal computers such as the Apple Macintosh and IBM PC to supercomputers such as the Cray X/MP. Similarly, the component subnets are a diverse lot. The fact that these networks and hosts interoperate effectively (most of the time) is a substantial achievement.
- The internet model greatly facilitates the autonomous administration of various subcomponents. For example, if an organization wishes its network to be a component of the internet, the only thing it needs to do is to get a network address and name space and identify a gateway which can provide an interface between itself and the internet. Organization is autonomous in the sense that it can do its own networking and manage its address and name space.
- The datagram-based transport model provides a simple interface, a great deal of flexibility and simple failure recovery.
- The internet model, through its general purpose server-client interface, facilitates the development of a variety of applications. Also, the model supports an open architecture philosophy encouraging its users to develop their own applications.

At the same time, the current internet model is not without its weaknesses. These are summarized below:

- One fundamental problem with the current internet is that it simply does not have enough raw bandwidth to carry the increasing traffic and support new applications such as remote interactive graphics and multimedia communications, because these applications need an order of magnitude more bandwidth than current applications.
- Clearly, putting new links with higher bandwidth (100 Mb/s and more) will not be a solution by itself, because the packet switching technology in use cannot effectively use such high bandwidths. A packet switch is typically a low-end minicomputer, and it does most of the per-packet processing by its slow software controlled processors. As a result, obtainable throughput from a packet switch is much lower than the data rate of a high speed link.
- The internet model is unable to support a predictable level of performance for its applications. This is mostly true because of its datagram philosophy and because it expects so little of its component subnets. As mentioned earlier, the internet requires that a component subnet try its best to forward a datagram toward its destination but allows the subnet to lose, resequence, and duplicate datagrams. Moreover, there is not even an attempt to characterize the throughput that the subnets can deliver to an application. This makes it difficult to provide predictable performance.
- Finally, the lack of central management makes the engineering of facilities (that is determining when, where and how new transmission facilities and switching systems should be installed) difficult. While this lack of central management is intentional and in some ways desirable, it does make it difficult to maintain a network configuration that can keep up with growing demand.

It should be noted that some or all of these weaknesses are well-known within the research community, and the federal agencies which support the internet have already started working on a plan for the next generation of internet [3]. Through this article, we want to present our initial thoughts on a model of a high-speed internet appropriate for the future. We feel as many do that such a high-speed internet should be based on some of the recent developments in high-speed packet switching, which are reviewed in the following section.

High-Speed Packet Switching

Since the late seventies there has been a growing interest in the possibility of large scale, public communication systems that are flexible enough to support a wide range of applications from low speed data, to voice, to still image transmission and full rate video. The Integrated Services Digital Network (ISDN) is in large part an outgrowth of this interest, but at least in its current incarnation offers only limited capabilities. In the early eighties researchers at AT&T Bell Labs proposed Fast Packet Switching (FPS) as a possible solution to the problem of supporting diverse applications in an integrated fashion [4] [5]. At about the same time, a team of researchers at the Centre National d'Études des Télécommunications (CNET) in France were proposing a similar approach which they referred to as Asynchronous Time Division Switching (ATD)[6]. These efforts have in turn spurred considerable additional activity at a number of other research centers [4] [5] [7–11].

In general, the various research teams working on this problem have recognized that some form of packet switching is needed to provide the flexible use of bandwidth required to support the range of applications of interest. At the same time, they have recognized that in order to provide such communication services in the context of large public networks, higher performance and more cost-effective implementations of packet switching are required. The various efforts rely on the combination of high performance digital transmission facilities and hardware implementations of the basic switching and protocol functions. They also use a connection-oriented model of packet switching, in part to simplify the hardware implementation but more importantly to allow explicit resource allocation, which in turn allows the network to provide predictable performance during periods of overload. A number of differences exist at more detailed levels but, philosophically, the approaches taken are fairly similar.

To allow a more concrete discussion in later sections we now briefly describe certain aspects of the work being carried out by the authors and others at Washington University on

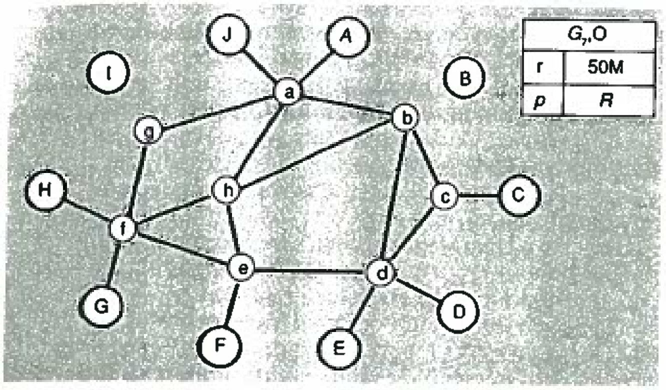


Fig. 1. One-to-many connection.

high-speed packet switching. This project is fairly representative of the work being carried out at other places; perhaps its main distinguishing feature is its strong focus on problems associated with multipoint communication.

Multipoint Connection Model

We start by describing the basic model of communication used in our work. It is a connection-oriented model in which a user must open a connection before transferring data, but, unlike conventional virtual-circuit models, the network does not guarantee that all packets will be received or that they will be delivered in sequence. The basic mechanisms are designed so that the probability of both packet loss and misordered packets is small, but the network provides no internal error recovery mechanisms in order to reduce that probability to zero. The philosophy is that such mechanisms are best provided on an end-to-end basis, as they are needed, rather than on a universal basis. Perhaps the key reason we favor a connection-oriented model is that it permits the network to keep track of the available capacity and refuse new connections when it cannot support them with acceptable performance. This in turn allows the network to offer predictable performance to meet application-specific requirements.

When opening a point-to-point connection, a user is required to give a "rate specification," giving his required peak data rate, average data rate and a measure of burstiness that we refer to as the burst factor. These data are used by the network in making resource allocation decisions, and, once a connection is established, the actual bandwidth use is monitored to prevent users from exceeding their allocation and possibly usurping resources allocated to others. The basic model can be extended to allow negotiation of the rate specification for applications that do not really have hard requirements. In addition, we can permit the rate specification to vary during a connection, perhaps in response to changes in requirements by other connections.

In our work, the basic connection model supports multipoint communication; that is, a single connection may include a large number of endpoints. Several modes of multipoint communication appear to be useful, including "one-to-many" (see Figure 1) which can be used to distribute a signal such as a video program to many receivers, "many-to-one" which can be used for data collection, "many-to-many" which can be used for conferencing and LAN interconnection and, finally, hybrids which combine different modes. To support this sort of flexible connection model, we need a simple method to describe a general multipoint connection. We illustrate our approach here with a few examples; additional details can be found in Haserodt and Turner's research [12].

Our connection abstraction allows a multipoint connection to have more than one channel and each channel to have different bandwidth parameters and access permissions. The concept of multiple channels within a connection can be useful in more complex applications, such as video lecture, and multimedia conferencing.

Multipoint Switching

We now briefly describe one of several switching system architectures that has been proposed for high speed multipoint packet switching. A more detailed description can be found in two studies by Turner [5] [10]. The overall structure is shown in Figure 2. It is topologically simple and well-suited to Very Large Scale Intergration (VLSI) implementation. The system consists of a set of Packet Processors (PP) which interface to the external links and provide all per packet protocol processing, a Connection Processor (CP) which sets up and maintains multipoint connections, and a switch fabric consisting of a Copy Network (CN), a set of Broadcast and Group Translators (BGT), a Distribution Network (DN) and a Routing Network (RN).

Packets enter one of the packet processors at left, where an address translation is performed. For point-to-point packets this yields an outgoing link number and an outgoing channel number. These are placed in the header of the packet, which then passes through the CN, one of the BGTs and the DN, following some arbitrary path. When the packet reaches the RN, it is routed using the outgoing link number. The RN is a conventional binary routing network with sufficient storage at each node to store a small number of complete packets. When the packet reaches the outgoing PP, the extra header information added at the incoming PP is stripped off and the packet is transmitted on the outgoing link. The role of the DN is to randomly distribute packets it receives across its outputs. This prevents congestion that can otherwise occur in the RN when subjected to traffic patterns with strong "communities-of-interest." The CN on the other hand is responsible for making appropriate number of copies of an incoming packet belonging to a multipoint connection.

The system is controlled by the Connection Processor (CP) which can change the contents of the control memories in the PPs and BGTs, in order to set up, modify or remove connections. This design is well-suited to implementation in a medium speed, high-density technology like Complementary Metal Oxide Semiconductor (CMOS). While the per node buffering makes the individual switch elements moderately complex, the topological complexity is very low. The only large memories are in the PPs and BGTs, and these need be accessed only a few times per packet cycle, permitting the use of high-density memories with relatively long cycle times.

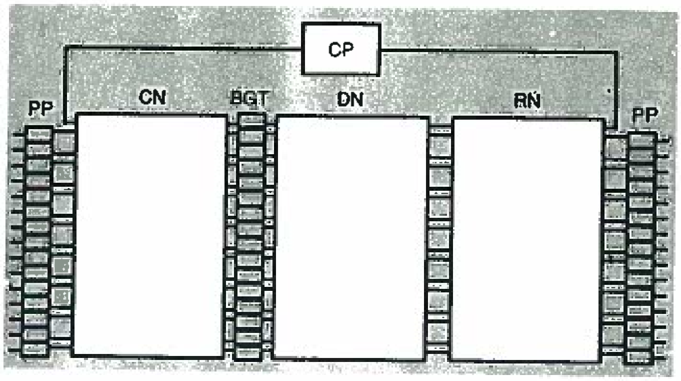
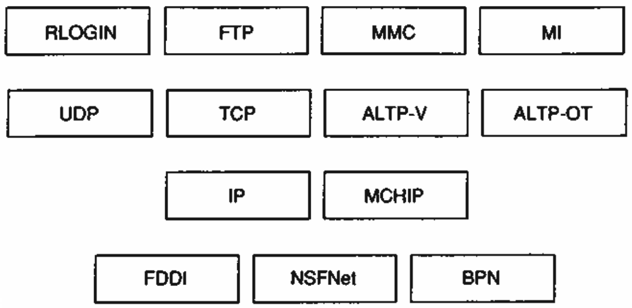


Fig. 2. Broadcast packet switch fabric.



RLOGIN	Remote Login
FTP	File Transfer Protocol
MMC	Multi-Media Conferencing
MI	Medical Imaging
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
ALTP	Application-oriented Lightweight Transport Protocol
ALTP-V	ALTP for Voice
ALTP-OT	ALTP for Object Transfer
IP	Internet Protocol
MCHIP	Multipoint Connection-oriented High Performance Internet Protocol
FDDI	Fiber Distributed Data Interface
NSFNet	National Science Foundation Network
BPN	Broadcast Packet Network

Fig. 3. Extended Internet protocol structure.

Elements of an Extended Internet Model

The ideas developed in the ARPA Internet protocols are important ones, because they demonstrate that it is possible to build systems that support interworking across independently administered and technologically dissimilar subnetworks. We believe that the existence of such subnetworks is a fact of life and that continuing technological change and organizational imperatives will ensure their future proliferation.

The current communications environment offers a multitude of examples of the heterogeneity of communication systems. In the U.S. telephone network, there are several major long distance carriers and a fairly large number of small carriers; there are about ten large companies offering local telephone service and over a thousand small companies. Each of these entities, in some sense, operates its own network, although detailed standards and regulations enforce a high degree of uniformity. Private businesses also operate independent systems, including Private Branch Exchanges (PBX) and corporate wide-area telephone networks, which currently cannot be integrated into the larger public network in a completely satisfactory way. In the computer networking community there is even greater diversity. The current U.S. research internet includes over a dozen wide-area networks using several different communication technologies and supporting communication among thousands of sites, most of which connect through campus or local area networks.

The future high speed networks now under development will have to operate in an environment that at the least includes the networks that exist today. They will have to support interoperation with telephone networks and data networks including those based on current internet protocols and commercial X.25-based networks. They must also provide support for interworking with LANs, PBXs and private corporate networks of various sorts. In addition, since there can be no final solution in a time of rapid technological progress, new network technologies will arise which we will want to incorporate into the developing communication infrastructure. We can reduce the effort required to accommodate such developments by establishing a suitable framework into which they can fit.

Given then that future networks will operate in a heterogeneous environment, how must we extend current internet concepts to take advantage of the high speed communication technologies now under development? We feel that at the internet level, the model must be extended in three major ways. First, it must support a connection-oriented transport service at the internet level, that can support applications with demanding performance requirements. Second, it must support a more general addressing scheme, to allow interworking among diverse subnets. And, third, it should provide a framework for parametric description of subnet capabilities and connection requirements, allowing the routing of connections through subnets with appropriate capabilities in an application-independent fashion.

At the transport level, we argue that the transport protocols should be simpler, designed to be mostly implemented in VLSI, well-integrated with the host architecture and operating system, and should provide reliability and performance guarantees as requested by specific class of applications. We call such a transport protocol an Application-oriented Lightweight Transport Protocol (ALTP).

The extended internet protocol hierarchy is shown in Figure 3. It is important to note that the protocol hierarchy is for "control path," and that the implementation model aims at allowing "data path" to be implemented in VLSI with no layers of protocol processing. Extensions to the internet level are discussed more fully in the following subsections.

A Connection-Oriented Internet Protocol

One common element in most work on high speed networks is the use of connection-oriented packet switching. There are several reasons for this. Perhaps the most obvious is performance. Connection-oriented systems separate the more complex control operations from data transfer, allowing simple and fast hardware implementations of the data transfer. This is an important issue both within switching systems and within the various devices that communicate across the network. We feel that this separation of control and data will also be important in delivering high performance to computer applications, since it will facilitate the development of hardware-based mechanisms for speeding data transfers into and out of computer systems.

A second reason that connection-oriented networks are attractive is that they allow the network to make explicit resource allocation decisions when connections are established, and this in turn makes it possible to offer far more predictable performance than is possible in connectionless networks. Such predictable performance, while not always essential, is necessary for applications like interactive voice and video. As mentioned above, such resource allocation decisions must be made using some form of rate specification that describes the amount of bandwidth the user requires. For applications that do not have hard requirements, one can have "soft specifications" that give a range of acceptable possibilities and allow the network to make its resource allocation decisions relative to other traffic. This can be extended further; we can include connections with "degradable specifications" and allow the network to take resources away from such connections in order to accommodate new traffic.

Finally, connection-oriented networks offer more generally useful methods of multipoint communication than is possible in truly connectionless networks. Connectionless networks can support multipoint communication in one of two ways: either transmission of packets with a list of destination addresses or by transmission to a multicast address. The first mechanism is useful only for multicasts with a small number of endpoints and the second is essentially a form of connection, since it must be possible to associate a given set of endpoints with a given multicast address and these associations can be expected to change over time.

The arguments given above imply that an extended internet model should include a connection-oriented service, as indicated in Figure 3. We refer to the connection-oriented service as Multipoint Connection-oriented High Performance Internet Protocol (MCHIP). MCHIP is a plesio-reliable connection-oriented protocol which means there is no hop-to-hop flow and error control as in X.25 virtual circuit networks. A connection in this context only implies a predetermined path for packets and some resources "statistically" bound to the connection. Resource allocation accounts for the characteristics and performance needs of the connection or the application. The multipoint communication support is an integral part of this protocol because multipoint communication is a very useful facility for a number of applications and for control and management of networks. Finally, the next generation protocol has to be high performance which means it allows fast hardware implementation of per packet processing to ensure high throughput and low latency. A detailed design and specification of MCHIP is in progress [17].

As shown in Figure 3, MCHIP would not replace the existing connectionless service but would likely be the method of choice for applications that are connection-oriented at a higher level.

A Flexible Internet Addressing Scheme

One of the key requirements for an internet that can support communication across diverse subnetworks is a flexible addressing scheme that allows one to specify terminal devices that may be located in any of the subnetworks. Some of the specific objectives we have in mind are listed below:

- We want a scheme that allows us to address devices on existing subnetworks as diverse as current telephone networks (both public and private), the ARPAnet, X.25 data networks, and local area networks. Note that these networks currently use completely different forms of addressing and it is unrealistic to expect them to change.
- The addressing scheme should support a hierarchical organization of the address space, allowing the separate entities responsible for various subnetworks to manage their portions of the address space independently of the others.
- In order to support very large networks the addressing method must support routing methods that are not dependent on detailed global knowledge of network topology or traffic.
- The addressing method should allow for devices that have an identity independent of their current network location, facilitating the relocation of terminal devices and communication among mobile devices that may not have any fixed location.

Current networks offer a variety of possible models for a general addressing scheme. For example, the telephone networks support a large hierarchically organized address space with portions of the address designating country and areas within a country. These allow separate administration of the address space at least at the national level, with more limited autonomy for individual organizations within a country. The hierarchical organization also supports routing methods that use the hierarchy to limit the knowledge required of individual switching systems. This scheme also has some limitations. First, it does not support autonomous address space administration for private organizations; in particular those organizations that operate their own wide area subnetworks have no means of identifying devices with their subnetwork, making it difficult, for example, to route connections that go between the private and public networks in the most efficient way. Second, because devices have no identity separate from their location, relocation of devices requires manual updating of routing information. The "domain addressing" method used in the ARPA internet protocols provides a more flexible structure, supporting independent address space management by private organizations both large and small. It also supports identification of devices independent of their location, allowing straightforward device relocation.

Neither of these techniques provides the level of flexibility that we feel is needed in a general internet environment. In particular, neither allows for operation in an environment that includes subnetworks with their own addressing schemes that may be very different from one another. We next outline a framework for addressing that would allow co-existence of such diverse subnetworks. Our framework is driven be our desire to support interoperation among existing and future subnetworks, without imposing the unrealistic requirement that their native addressing methods be abandoned. Our framework supports a hierarchical organization that allows routing to be carried out without the need for detailed global information. We envision this hierarchy to be based on routing knowledge rather than on physical connectivity. The proposed form of an address in our framework is as follows:

$$\mathrm{OBJ} :: \mathrm{DOM}_1 :: \mathrm{DOM}_2 :: \mathrm{DOM}_3 ::...$$

OBJ denotes an object, which can designate a physical terminal device or some other entity within a terminal. Object identities must be unique across the entire network, a requirement that can be fulfilled by having the identity incorporated into a device at the time of its manufacture. DOM denotes a domain, which is either a subnetwork or a subnetwork together with the address of some entity within the subnetwork, where such an internal entity is identified using the subnetwork's native addressing scheme. Subnetwork identities are globally unique, a requirement which implies the existence of an organization or organizations to assign them.

To be useful, the particular sequence of domains in an address must tell us something about routing. We require only that DOM_i have a mechanism for determining how to reach DOM_{i-1}. This allows (but does not require) that the "path" implicit in the address to be used for routing. In general, when a route is required for a given address, local routing information would be used at various switching systems and gateways to find a short path. So, for example, if a path to the device specified by OBJ were known by a particular switching system, that path could be used, ignoring the remainder of the address. Similarly, if a given gateway knew how to reach DOM_2, it would not be obliged to go through DOM_3. Note that there is no requirement that DOM_i be physically connected to DOM_{i-1}, only that it be able, using whatever internal mechanisms it has at its disposal, to find DOM_{i-1}.

The OBJ portion of the address is optional, allowing for anonymous objects identified only by their location. This is needed for interworking with networks that support only anonymous devices (such as telephone networks), and relieves simple devices of the requirement for explicit identities. As indicated above, DOM_i may specify either a subnetwork or an entity within a subnetwork. The appropriate form would be determined by the subnetwork. For example, in a subnetwork in which every switch/gateway knows how to reach every other, DOM_i could specify just the subnet rather than a particular entity on it. On the other hand, in a large wide area network where individual switches/gateways have only partial knowledge, DOM_i might specify both the subnetwork and the identity of a gateway that joins the subnetworks of DOM_i and DOM_{i-1}. Alternatively, DOM_i might give the identity of a routing processor within the subnetwork. In the case of the first domain, DOM_1 is a little special; in particular, when no explicit object identity is provided, DOM_1 must include the address within the subnet of the object.

This kind of a framework allows for the interoperation of existing networks, including some very dissimilar ones. For example, it allows a telephone call to be placed from an appropriately equipped personal computer located on a local area network to an ordinary telephone on the existing telephone network. It also allows connections between LANs to be routed using switched connections in the telephone network, if appropriate. It supports flexible routing for private wide area networks, for example, by allowing routes that enter the private network at the most convenient gateway rather than by requiring (as with current private wide area telephone networks) that they enter at a particular place. This in turn allows transparent movement of devices within such subnetworks.

Parametric Description of Subnet Capabilities

Given the variety of capabilities of the subnetworks included in an extended internet, it is essential that the internet protocol include mechanisms for describing the capabilities of subnetworks, so that routing decisions can be guided by this information. For example, when selecting a route for a connection requiring a bandwidth of 1 Mb/s it is essential that the route not traverse subnetworks incapable of supporting that bandwidth. Similarly, connections requiring low packet loss rates should not be routed through subnets that lose packets frequently.

The following list gives a few of the parameters that might be included as part of a subnet description; a few of these parameters are given relative to a "standard reference path," which for example might be a path carrying heavy traffic between devices that are, say, 1,000 km apart:

- Bandwidth options. This specifies the various connection bandwidths that the subnet can support. It may be specified as a few discrete values or ranges of values.
- Bandwidth allocation option. This specifies the type of bandwidth allocation that the subnet can support. Options include peak bandwidth allocation, in which bandwidth is dedicated to a connection and cannot be statistically shared with other connections; statistical allocation, in which connections with varying instantaneous data rates can statistically share bandwidth, but with explicit allocation provided to ensure predictable performance; and no bandwidth allocation, in which no performance guarantees are provided.
- Packet loss rates. This specifies the frequency of packet loss on a standard reference path.
- Packet misordering separation. This specifies the time between transmission of packets on a standard reference path at which the likelihood of packet misordering exceeds some threshold (say, 10^{-3}).
- Packet delay. This specifies delay on a standard reference path (perhaps average and ninety-ninth percentile).
- Multipoint capability. This enables one to support multipoint connections.
- Transit traffic. This specifies a subnet's willingness to carry transit traffic, that is, traffic that crosses the subnet but does not terminate at some device on the subnet.

The parameters listed above are envisioned as static. It may also be useful to allow more dynamic traffic information to be included and updated periodically. Using these parameters and possibly others together with knowledge of individual connection requirements, it is possible for the MCHIP protocol entities to make informed decisions when routing connections.

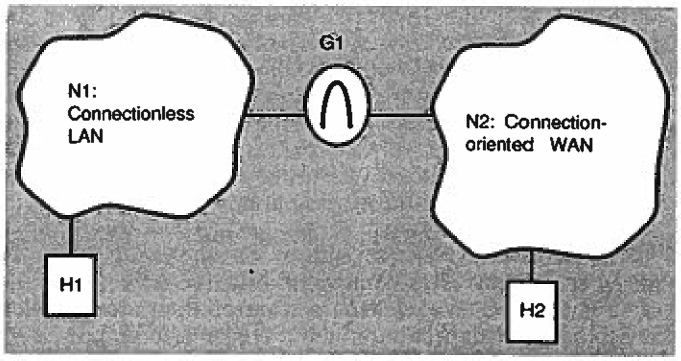


Fig. 4. Connection across a LAN.

Design and Implementation Issues

We have argued that the current connectionless Internet Protocol should be supplemented by a connection-oriented service for use by applications that need predictable and high performance connection-oriented service. In a heterogeneous networking environment, however, there will be subnetworks that are either purely connection-oriented or purely connectionless, and, consequently, we need to address the question how to provide a connection-oriented (connectionless) internet service on and across connectionless (connection-oriented) subnets. In this subsection we present our initial thoughts on some of the implementation considerations relating to the interconnection of connection-oriented and connectionless subnetworks using a few specific cases.

Connection Across a Connectionless LAN

Consider Figure 4. How should we implement a connection across the LAN which has no concept of connection? There are two major issues involved. First, how is a connection established and, second, once it is established, how are packets transferred from the gateway to the appropriate host and vice-versa? One possibility is to include the connection establishment functions within the gateway; that is, hosts requiring connections would exchange control messages with the gateway at the time of connection establishment, and the gateway would in turn exchange control messages with the connection-oriented network (N2) to complete the connection. Assuming the connection can be established, the gateway and H1 must agree on a way to identify packets belonging to the new connection. Then, during the data transfer phase, the main gateway function would be to perform a routing translation; that is, when a packet is received from N2 on the logical channel selected during connection establishment, it would be reformatted to include the address of H1 and whatever local identifier agreed upon by H1 and the gateway; in the other direction a similar translation would be performed. One of the gateway's functions would be to monitor the usage of resources on the access link to N2, rejecting new connections that would overload it. In addition, the gateway should ideally monitor the LAN traffic and take it into account when accepting new connections. The monitoring function is not difficult to implement if one takes advantage of the broadcast nature of most LANs. On the other hand, since much of the traffic on the LAN is not under the control of the gateway, resources available at connection establishment may not continue to be available later. How serious this issue is clearly depends on the proportion of the traffic that is subject to explicit resource allocation. We note that the connection establishment functions need not be built into the gateway device. Any host on the LAN could implement these functions, allowing the gateway to be implemented as a relatively simple and fast hardware device. Its functions would in this case be reduced to packet relaying, using information in control tables that can be updated by the controlling host.

Connection Across a Connectionless WAN

Consider Figure 5. Two hosts H1 and H2 have set up a connection through networks N1, N2, and N3 where N1 and N3 allow a connection-oriented access, whereas N2 is a Wide Area Network (WAN) and provides only a connectionless service. The MCHIP protocol entities at G1 and G2 should attempt to make the fact that the connection is routed through a connectionless subnet transparent to the applications on H1 and H2. This may require that the gateways attempt to improve the connectionless subnet's performance in terms of packet loss or resequencing. A few possible strategies for doing this are listed below:

- They can send packets across network N2 using network N2's standard datagram facility. As a result, the probability of out of sequence, lost, and duplicated packets received at the other end will increase. However, one may argue that it is the cost of routing a connection through a datagram network. The penalty may not be prohibitive if the network is not heavily loaded and does not generally resequence the packets which can be expected of future datagram networks.
- Gateways G1 and G2 can manage a virtual connection between them for every connection they route via network N2. In other words, G1 and G2 could use a standard protocol with flow control and error recovery so that that the packets in the connection are delivered to the other side of network N2 in sequence and without being duplicated or lost. Clearly, this would require more complexity at gateways and would introduce additional delay.
- If network N2 supports source routing, G1 and G2 can choose a path and use source routing to send packets of a connection to each other. Because all packets will use the same path, the probability of their being resequenced can be low. Clearly, this only addresses the resequencing issue and not the packet loss issue.
- Packets could be resequenced at the receiving gateway using a simple protocol that simply attempts to reduce the frequency of out-of-sequence packets by adding a small resequencing delay and reordering misordered packets within this time window. This avoids the complexity and performance penalties associated with a completely reliable protocol.

In terms of resource allocation and management, it is not clear how MCHIP entities can do resource allocation and management across a wide-area datagram network. One simple approach to achieving this functionality is to designate every gateway to serve as a resource manager or resource server—similar in spirit to a name or a route server. The gateways keep track of active connections and available resources in the network. Every time a new internet level connection is established within or across this network, one of the gateways is consulted to check if the appropriate resources are available to support the connection. Gateways talk to each other to ensure that their picture of the resource availability in the network is consistent.

In case of datagram networks, packets or datagrams of a connection may travel on different routes, and, therefore, a gateway needs to consider alternate paths and allocate suitable resources on these paths. If a datagram network allows source routing, the gateway can avoid allocating resources on alternate paths by specifying a source route and allocating resources only on this path. Feasibility and effectiveness of such resource allocation methods on datagram networks is a subject of further research.

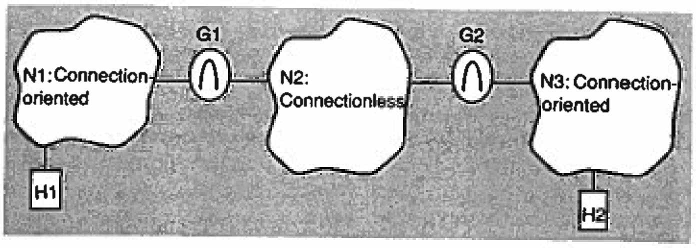


Fig. 5. Connection across a WAN.

It is also possible that connections with demanding resource requirements should be routed around connectionless networks whenever possible and in the case where no alternative is available, a resource negotiation would take place, making it possible for the application to either accept the available performance level or decide not to complete the connection.

Datagrams Across a Connection-Oriented Network

The third possibility that we have to deal with is that of routing datagrams in a connection-oriented network. For example, consider Figure 4, and suppose we want to route datagrams across the connection-oriented subnet, N2. The datagrams entering network N2 via various gateways may be either destined to hosts in network N2 or may be forwarded by N2 to other networks. That is, in the second case, N2 is only a transit network. One of the following three strategies can be used:

- A simple but perhaps inefficient solution is to set up a connection within N2 whenever a gateway has a datagram to route. Once the datagram is successfully routed, the connection is torn down. Clearly, if the overhead of setting and breaking a connection is high, this solution will lead to degraded performance.
- A slightly modified scheme is to open a connection for a datagram and keep the connection open with the expectation that there will be more subsequent datagrams to be routed along the same path. As the overhead of setting and breaking a connection is distributed over a number of datagrams, this approach can do better than the first one. However, in this case, gateways have to be more intelligent in terms of deciding when to set and break a connection. This is a scheme similar to the one currently being used in ARPAnet where IP datagrams are routed on X.25 connections.
- In the case of networks which allow multipoint and variable bandwidth connections, a more elaborate scheme can be designed for routing datagrams on a connection-oriented network. For example, assume network N2 is such a network. Gateways like G1 can have a number of multipoint connections open at all times to route datagrams. When a gateway has a datagram to send to another gateway, it will try to use an existing connection or a concatenation of parts of existing connections to route the datagram. The bandwidth and connectivity of these connections can be dynamically adjusted depending on the datagram traffic. For example, if there is not much datagram traffic on a certain connection, its bandwidth can be slowly decreased. On the other hand, if there is some new datagram traffic to a gateway or host which is not already on an open multipoint connection, the host or gateway can be added to the connection. In short, hosts and gateways on multipoint connections can observe and learn from the traffic pattern and adapt the connections such that datagrams are routed without much per-packet overhead. Of course, as these connections will learn and adapt on a continuous basis, there will be times when some datagrams will see more processing and experience more delay, but on an average, this scheme should give reduced per-packet processing, higher throughput and lower delays.

End-to-End and Host Interface Issues

We are entering a period where computer networks, for both local and wide area applications, with raw data rates of 100 Mb/s to over 1 Gb/s are becoming a reality. Unfortunately, current computers and workstations have yet to deliver the performance potential of even the much slower networks (such as Ethernet) that are now commonly available. Current communication protocols and applications software were designed around a set of assumptions that are growing more outdated every day and the time has come to rethink the way we handle these applications at a very fundamental level.

There is growing recognition of this problem and several researchers have begun to address it [13–15]. These proposals can be characterized by a couple of common threads. First, they argue for end-to-end mechanisms, such as end-to-end flow and error control which work well for the high speed environment with large bandwidth-delay product. Second, they propose that applications be allowed to specify transfers of very large amounts of data at one time, requiring possibly hundreds or thousands of individual packet transfers. Finally, that special-purpose hardware be brought to bear on the problem, to permit higher speed transfers than possible with software alone. We propose taking this a couple steps further. The important aspects of our approach can be summarized as follows:

ALTP Approach

We argue that there should be a few Application-oriented Lightweight Transport Protocols (ALTPs) which can provide variable grade end-to-end flow and error control to different classes of applications. In general, the transport protocols should be simpler, designed to be mostly implemented in VLSI, well-integrated with the host architecture and operating system, and should provide reliability and performance guarantees as requested by specific class of applications. For example, a transport protocol designed to support voice communication can be designed such that it guarantees less than 30 m/s delay, guarantees no out-of-sequence packets, and allows only a few dropped packets. A transport protocol designed for distributed IPC should be able to guarantee minimal delay, reliable transaction delivery, use a shared memory interface to processes on different machines, etc.

Flow Control in ALTPs

When an ALTP or an application opens a connection, it specifies attributes of the connection in terms of average and peak bandwidth and a factor reflecting the burstiness of the transmission. Since the connection set up is end-to-end, all the intermediate systems (including packet switches, gateways, and the end hosts) can make appropriate resource reservations based on the rate specifications. As a result, as long as both ends transmit subject to the rate specification, the probability of packet loss due to buffer overruns is very low, and, thus, there is no need for end-to-end flow control. During the life of a connection, the transmitter and receiver may want to change the specifications of the connection which requires end-to-end negotiations similar to what is done at the beginning of the connection. In ALTPs, error and flow control will be separated, and mechanisms will be provided to ensure that resources and buffers are available for the life of the connection.

End-to-end flow control cannot be eliminated for some applications, because they require frequent changes to the connection characteristics or because the underlying network does not do static resource allocation for the connection. In the latter case, the network may once in a while need to throttle some sources to relieve local short term congestion. In such a situation, the best way to implement flow control may be similar to what is suggested in the current transport protocol proposals except we would like to make it more application specific.

Error Control in ALTPs

For end-to-end error control, application specific methods which are independent of end-to-end latency are used. For example, consider a transport protocol designed for a distributed object-oriented inter-process control (ALTP-IPC), which allows processes to communicate by a shared memory paradigm. Assume that one process sends a memory segment with N pages, and, because of errors, n pages are corrupted or lost. In this case, the transport protocol should store the pages received correctly in memory, allow processes to access those pages, and should request retransmission of just the n pages received with error. Thus, the processes can start to execute, unless they need access to pages which are being retransmitted. Note that this is possible because ALTP-IPC has some knowledge of its application, and understands the application specific data objects.

In the case of a voice transmission (ALTP-V), when a packet is lost, there is no need for a retransmission. Similarly, if a packet arrives out of sequence, ALTP-V drops this packet rather than sends it to the application.

In the case of a file transfer (ALTP-FTP), the whole file is received, and only packets received in error or lost are retransmitted (which do not include out-of-sequence or duplicated packets). It should be noted that the errors and corresponding retransmissions do not affect the error-free transmission of other packets. In other words, two ends do not need frequent synchronizations, and the selective retransmission strategy is application-dependent.

Note that in these examples the various ALTPs use different error control mechanisms and that the mechanisms are more effective because they use the knowledge of the application and its data units that are being communicated.

Protocol Implementation Model

We want to emphasize that the layered or hierarchical model is good for the design and understanding purposes, however, it is inefficient for the actual implementation. Thus, we want to propose a general purpose powerful model for high performance protocol implementation. This model can be used for the implementation of a stack of protocols or for the implementation of a single protocol within a host.

The layered implementation tends to be slow because of data formatting and other processing to be done at each layer to interface with the lower/higher layer. Most of this processing for a variety of applications is simple enough to be implemented in hardware, but we lack mechanisms for making this happen. We argue that if the application is really to see the performance that emerging networks are capable of delivering, we must provide hardware support for user applications, and, since there is a wide range of differing applications to deal with, we also must establish a framework that is sufficiently flexible and general so that as new needs arise, we can satisfy them by the addition of new hardware modules.

We propose that the user applications or protocols of concern describe their needs to the communication subsystem in a more abstract fashion, specifying where the data to be transferred can be found and what processing must be performed on the data in carrying out that transfer. One possible approach is for the application software to describe its requirements to the communication subsystem by specifying pipelines of "abstract filter modules" (or simply filters) that collect the data to be transferred, perform one or more computational operations upon that data, and transfer it to one of possibly several network interfaces. The filters are abstractions that can be implemented either in hardware or software, depending on performance requirements.

We consider an example of multimedia communication where we can specify the data transfer as follows:

```
GetWindow p n siz inc flag1|
FlowControl flag2|
ErrorControl flag3|RunCode|
PutEther header
```

We can imagine this model being implemented by a collection of processing modules connected to the host's main bus system. Each module would possess a substantial amount of memory in the host's address space. This memory would be used for buffering and for a set of control blocks that the communication software would use to manage the various pipelines implemented by the modules. Modules could perform block transfers across the bus to move data to the next module in the pipeline. Several modules could be combined into a single hardware unit for more economic implementation or to reduce the performance impact on the host's bus.

Conclusion

In this article we have briefly presented an overview of the ARPA internet model and of the state of high-speed packet switching. We have argued that the internet model needs to be extended in order to support the growing need for high-speed applications. Similarly, the effort in high-speed packet switching needs to be broadened to account for the fact that such networks have to operate in a complex internet of heterogeneous subnets.

In an effort to work out a model of an extended internet, we have argued for a connection-oriented internet protocol, a general addressing scheme, an ALTP approach for the transport protocols design, and a general purpose VLSI-based protocol implementation model for host interfaces. More details on the extended internet model, specification of MCHIP, and the issues concerning the host-network interface can be found in [16–20].

References

[1] J. Postel, "DOD Standard Internet Protocol," ARPA RFC 791.

[2] J. Postel, "DOD Standard Transmission Control Protocol," ARPA RFC 793.

[3] National Research Council, "Toward A National Research Network," National Research Network Rev. Comm. Computer Science and Technology Board, Commission on Physical Sciences, Mathematics and Resources.

[4] J. J. Kulzer and W. A. Montgomery, "Statistical Switching Architectures for Future Services," *Proc. of the Int'l. Switching Symposium*, May 1984.

[5] J. Turner, "Design of an Integrated Services Packet Network," *IEEE J. on Selected Areas in Commun.*, pp. 1,373–1,380, Nov. 1986.

[6] J.-P. Coudreuse and M. Servel, "Prelude: An Asynchronous Time-Division Based Network," *Int'l. Commun. Conf.*, 1987.

[7] M. De Prycker and J. Bauwens, "A Switching Exchange for an Asynchronous Time Division Based Network," *Int'l. Commun. Conf.*, 1987.

[8] A. Huang and S. Knauer, "Starlite: A Wideband Digital Switch," *Proc. of Globecom '84*, pp. 121–125, Dec. 1984.

[9] T. Takeuchi *et al.*, "An Experimental Synchronous Composite Packet Switching," *Proc. of the Int'l. Zurich Seminar on Digital Commun.*, pp. 149–153, Mar. 1986.

[10] J.S. Turner, "Design of a Broadcast Packet Switching Network," *IEEE Trans. on Commun.*, vol. 36, no. 6, June 1988.

[11] Y. S. Yeh, M. G. Hluchyj, and A. S. Acampora, "The Knockout Switch: A Simple Modular Architecture for High Performance Packet Switching," *Int'l. Switching Symposium*, Mar. 1987.

[12] K. Haserodt and J. S. Turner, "An Architecture for Connection Management in a Broadcast Packet Network," Washington Univ. Computer Science Dept., WUCS-87-3.

[13] D. Cheriton, "VMTP: A Transport Protocol for the Next Generation of Computer Systems," *SIGCOMM '86 Symposium: Commun. Architectures and Protocols (ACM Computer Commun. Rev.)*, vol. 16, no. 3, 1986.

[14] G. Chesson, "Protocol Engine," *Proc. of the USENIX Conf.*, 1986.

[15] D. D. Clark, M. Lambert, and L. Zhang, "NETBLT: A High Throughput Transport Protocol," *SIGCOMM '87 Symposium: Frontiers in Comp. Commun. Tech. (ACM Comp. Commun. Rev.)*, vol. 17, no. 5, 1987.

[16] G.M. Parulkar, "The Next Generation of Internetworking," Washington Univ., Computer Science Dept., WUCS-89-19, 1989. Also *ACM SIGCOM Computer Commun. Rev.*, vol. 20, no. 1, Jan. 1990.

[17] J. P. G. Sterbenz and G. M. Parulkar, "Axon: An Architecture for High-Speed Distributed Applications," Washington Univ., Computer Science Dept., WUCS-89-36. Also to appear in the *Proc. of IEEE INFOCOM '90*.

[18] T. Mazraani and G. M. Parulkar, "Specifications of a Multipoint Congram-oriented High Performance Internet Protocol (MCHIP)," Washington Univ., Computer Science Dept., WUCS-89-20, 1989. Also to appear in the *Proc. of IEEE INFOCOM '90*.

[19] J. P. G. Sterbenz and G. M. Parulkar, "Axon: Network Virtual Storage Design for High Performance Distributed Applications," Washington Univ. Computer Science Dept., WUCS-89-13, 1989. To appear in the *Proc. of Tenth IEEE Distributed Computing Systems.*

[20] J. P. G. Sterbenz and G. M. Parulkar, "Axon: Application-oriented Lightweight Transport Protocol Design," Washington Univ., Computer Science Dept., WUCS-89-14, 1989.

Biography

Gurudatta Parulkar is an Assistant Professor of Computer Science at Washington University in St. Louis. He has been working on very high-speed internetworking and on a novel communication architecture for distributed applications requiring remote visualization. He has contributed to the IETF and INARC task forces.

Dr. Parulkar received the Ph.D. in computer science from the University of Delaware in 1987. His thesis was on the design of a high-speed reliable flood local area network.

Jonathan S. Turner received the M.S. and Ph.D. degrees in computer science from Northwestern University in 1979 and 1981. He is an Associate Professor of Computer Science at Washington University, where he has been since 1983. His primary research interest is the design and analysis of switching systems, with special interest in systems supporting multicast communication. He has been awarded a dozen patents for his work on switching systems and has a number of widely cited publications in this area. His research interests also include the study of algorithms and computational complexity, with particular interest in the probable performance of heuristic algorithms for NP-complete problems.