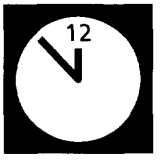

Report and Discussion on the IEEE ComSoc TCGN Gigabit Networking Workshop 1995

The strong message provided by the contributions to this year's conference is that a number of researchers are concerned with the practical implications of running TCP/IP over ATM, and most of those dealing with higher layer issues use TCP/IP over ATM out of expediency.

James P. G. Sterbenz, Henning G. Schulzrinne, and Joseph D. Touch

This is a summary of the second Gigabit Networking Workshop (GBN '95) sponsored by the IEEE Communications Society Technical Committee on Gigabit Networking (TCGN) in association with INFOCOM in April 1995. (Two earlier events were affiliated with ICC and LEOS Summer Topical Meetings, respectively.) We expect this to be an annual event in conjunction with INFOCOM. The workshop was handled entirely electronically: abstracts were submitted and reviewed and authors notified all by e-mail. The presentations were submitted by e-mail for inclusion in the on-line proceedings at `http://info.gte.com/ieee-tcgn/conference/gbn95`. In an attempt to encourage the presentation of current and ongoing work, the duration from the submission of abstracts to on-line publication before the workshop was only four weeks, with this report following three months later.

The focus of the TCGN and GBN workshop is on end-to-end (layer 4) and higher issues. There are of course a number of related network layer issues of interest, particularly in considering the end-to-end implications of network and internetwork architecture, but there is virtually no coverage of the physical and data link layers. Of the 27 submissions, 13 were accepted for presentation and one was used to lead a discussion session, resulting in submissions from five countries. Additionally, six were given poster space on the on-line proceedings. While a majority of the submissions had strong ATM (asynchronous transfer mode) content, quite a few had to do with TCP/IP (transmission control protocol/internet protocol) over ATM and B-ISDN (broadband integrated services digital networks), reflecting the emerging view that the Internet and ATM communities will have to try to coexist in the future. In fact, only two of the accepted contributions were completely free of ATM content. The sessions reflects this, starting out with some pure ATM/B-ISDN topics, continuing with TCP/IP over ATM, and leading into a discussion session critiquing ATM and considering its role in the future Internet-based Global Information Infrastructure (GII). The afternoon was devoted to non-ATM topics, followed by a discussion session on Middleware in which we tried our best to not mention ATM (without always succeeding).

The strong message provided by this year's contributions is that a number of researchers are concerned with the practical implications of running TCP/IP over ATM, and most of those dealing with higher layer issues use TCP/IP over ATM out of expediency.

ATM and B-ISDN

The first session consisted of presentations primarily concerned with ATM and B-ISDN issues.

"Cost Optimization of Bandwidth Usage in ATM Networks" — Aloke Guha and James P. Hughes; Network Systems Corporation, USA

The impact of using ATM and its real cost is expected to be a significant factor when considering networking across long-haul areas, since private ATM networks will not be dependent on carrier tariff rates or by restrictions on the quality of service.

The analysis begins by noting the impact of using ATM on high-end data applications. Unlike high-bandwidth multimedia applications, data applications need better and tighter guarantees on the correctness and the latency of the data delivered. First, error control is an issue because when multiplexing different connections, especially those with bursty traffic, on an ATM link, cell loss due to congestion is not avoidable. Current ATM switch designs indicate that the cell loss in the switch is far more significant than the normal bit error rate (BER) of the media. Rate-based flow control alone, beyond allocating the connection at

JAMES P. G. STERBENZ is with the Broadband Intelligent Networking project at GTE Laboratories, Incorporated

HENNING G. SCHULZRINNE is with GMD FOKUS

JOSEPH D. TOUCH is with the ATOMIC-2 and Cities-OnLine projects at USC/Information Sciences Institute.

0890-8044/95/$04.00 1995 © IEEE

The impact of using ATM and its real cost is expected to be a significant factor when considering networking across long-haul areas.

the peak rate, will not eliminate cell loss that leads to unreliable data transmission.

Second, to address error control during recovery from failed packet transmission, retransmission is required. However, the possibility of repeated retransmissions reduces determinism, not favored by most data applications. Therefore, some mechanism is required to better contain packet delay by minimizing the average number of retransmissions.

While the most obvious approach to avoid cell loss and the concomitant increase in packet delay is by overprovisioning, using dedicated links results in low utilization. However, it provides better performance, albeit at a higher cost. Note that established gigabit networks such as HIPPI use this approach. The key concern in ATM is cost, since the cost of a dedicated line will be higher than that of a shared one. The question is under what conditions does a shared ATM line become cost-effective when considering the delay-bandwidth product as the overall measure of performance. More generally, the optimization problem is to determine at what cell loss level and associated cost of the ATM line does a shared line become preferable over a dedicated one.

A recent proposal [ATM Forum 95-0150], advocates the use of forward error correction (FEC) to address the problem of data loss and packet delay. It shows that if FEC is used to both detect and correct cell erasures to effectively reduce delay, then shared ATM links can be competitive with dedicated links for a small penalty in performance. If the level of FEC is used optimally under different cell loss conditions, then the optimal cost is obtained by minimizing the cost function $C(L)$ with respect to the loss level L, where

$$C(L) = f(L)\,[c' + k(c' + 2TB)\,(nc'L/(1 - nc'L))]$$

where $f(L)$ is the cost per unit bandwidth for a given L, c' the actual number of cells per packet including the FEC overhead, k is half the window size in frames for the ARQ, T the round-trip delay, B the full link bandwidth, and n the number of switching segments in the network.

An interesting case for study, beyond determining the optimal cell loss level, is to compare the case of 1) $L = 0$, (a leased ATM line) and 2) $L > 0$ (a shared ATM line). The particular case studied is an OC-12 (622 Mb/s) line across a 50-mile network comprising ten ATM switching segments on which IP packets of 64 KB are transferred. It is assumed that the TCP window size is four and that each TCP segment contains one packet. The results show that even under modest cell loss (10^{-6}), the use of a shared ATM link with FEC will be cost effective if a dedicated line is priced at more than five percent of the shared line. The expected penalty in the packet delay in using the shared link using FEC is only four percent.

"Network Architectures and Multiple Access for ATM Satellite Networks" — L. Vidaller, J. Aracil, A. Martínez, J. Pérez, and A. Ruiz; Technical University of Madrid, Spain

New-generation satellites can play an important role in the future B-ISDN. In fact, considerable research effort is being done in RACE (Research for Advanced Communications in Europe) and ESA (European Space Agency). Nevertheless, very little technical literature has been written on this topic and the few papers that can be found do not consider important ATM issues such as traffic shaping, QOS (quality of service) guarantees, and statistical multiplexing.

This work focuses on a scenario where a large population of terminals are accessing an ATM cloud via satellite. This architecture is denoted user network interface extension via satellite links. A multiple access technique based on TDMA (time division multiple access) is proposed which can satisfy users QOS requirements with maximum satellite resource efficiency. Bandwidth is assigned to users taking into account delay requirements and transmission overheads of the satellite channel.

Performance evaluation is done by means of Markovian analysis and simulation. CBR, VBR, and ABR (constant, variable, and available bit rate) traffic sources are considered. VBR sources are modeled as on-off sources and expressions are derived to determine an equivalent bandwidth, using the fluid approach. Blocking probability issues for CBR sources and congestion control of ABR sources are considered.

The results show Erlang behavior for blocking probability, yielding closed formulas to dimension the network for a given quality of service. Also, expressions are derived to determine transmission queue length for VBR sources. Statistical multiplexing gain can be achieved by assigning the source an equivalent bandwidth in the range of 70 percent of the peak bandwidth, without degrading source performance. For ABR sources, expressions are derived with which a minimum bandwidth per frame (or utilization factor) can be calculated to avoid source congestion, considering the joint cell arrival process of sources is Poisson.

An experimental ATM gateway is being developed which interconnects ATM networks with the VSAT CODE network, and it will be an experimental testbed for new proposals. This prototype is based on multi-microprocessor boards (transputers) and will be operating within the VSAT CODE network using the Hispasat satellite.

"Traffic Control Mechanisms Comparison for ABR Traffic Transport" — M. Antico, F. Bernabei, and L. Gratta; Fondazione Ugo Bordoni, Rome, Italy

ABR service is being currently discussed in standardization and research groups, and is intended for the transport of data traffic, which presents a very stringent data integrity constraint while it is insensitive to the transport delay. For this kind of traffic no bandwidth reservation is carried out, consequently, the bandwidth left available by high priority traffic is exploited. In such an environment it is fundamental to avoid cell loss inside the network; in fact, this would cause the waste of bandwidth used to transport traffic that will need to be retransmitted.

This work is carried out in the framework of the RACE Project 2068 (LACE), and deals with the throughput and delay performance of ABR traffic transport mechanisms in ATM networks. A credit mechanism to control traffic inside the network is described, and its performance compared with that of the backpressure mechanism. Up to now, these mechanisms have been studied only at the access stage of the network. Here, the control

mechanisms are applied on a realistic and quite complex network model, constituted by various nodes interconnected on the basis of a two-stage star configuration. Using this network architecture, propagation delays are taken into account as they play a key role in the congestion control mechanism operation; moreover, the amount of signaling overhead is evaluated. In this way, it is possible to follow the congestion effects as they propagate inside the network.

The congestion control algorithm is based on the transmission of absolute occupancy information from the receiving to the transmitting entity. The sending station can evaluate the number of cells it can emit on the basis of this information and knowing the number of cells it has emitted since the last information arrived. The credit number evaluation is obviously conservative, since it must be assumed that the outgoing links of the receiving entity are unavailable. In the case of the credit mechanism, an interesting aspect is the tuning of the emission rate of the credit messages. In fact, a rate too high leads to a significant signaling overhead, while a low signaling rate leads to an overly conservative credit assignment.

The congestion control mechanisms are studied by means of simulation under various traffic and network configurations. The simulation shows that the main element discriminating the two mechanisms is the fact that the backpressure mechanism is dramatically sensitive to the network extent, as it cannot work beyond network spans fixed by the buffers dimension. Moreover, the backpressure thresholds must be properly tuned each time the network latency is changed. The credit mechanism, conversely, does not have limits in its operation, and can work independently of the network latency. However, for increasing network sizes, performance degrades. In the latency region where backpressure can operate, only slight differences can be observed in the throughput and delay performance of the two mechanisms.

Finally, an enhanced switch operation is proposed, aimed at sharing congestion among all the sources feeding an overloaded buffer. This switch operation has turned out to be very effective, mainly if operated in presence of the credit congestion control mechanism. This case shows more than an order of magnitude less cell loss, while in the backpressure case the advantage is less significant.

TCP/IP over ATM

The second session also dealt with ATM and B-ISDN, but explicitly considered running TCP/IP over ATM. Three of these contributions consider performance issues and problems in delivering bandwidth to the applications. The fourth considers how to integrate the two technologies to provide gigabit IP routing using an ATM core switching fabric.

"ATM Performance Measurement: Throughput, Bottlenecks and Technology Barriers" — Yves A. Fouquet, Richard D. Schneeman, David E. Cypher, and Alan Mink; National Institute of Standards and Technology (NIST), USA

ATM is often seen as a solution to networks with bandwidth limitation by many users today. Nevertheless, it is necessary to determine the amount of communication bandwidth actually available to applications, before trying to integrate this technology into legacy environments. Currently, information concerning ATM bandwidth is typically reported as being a function of hardware bandwidth. Indeed, raw ATM cell switching hardware is now extremely efficient, however the bandwidth offered to applications still varies with current internetworking mechanisms, system software, legacy protocols and networks, and existing end system limitations.

In order to determine ATM bandwidth between applications, the performance of a local area network based on the Fore Systems' hardware and software environment is studied. This is achieved by using a variety of data transfer techniques and benchmarks over the Fore Systems' ATM API (application programmer interface) and TCP/IP/ATM, in order to emulate both memory transfer and file transfer applications. Varying parameters such as the type of AAL (ATM adaptation layer), the size of the maximum transfer unit (MTU) and the network configuration, determines their effect on application bandwidth as well as cell loss and bottlenecks.

Throughput increases asymptotically with the amount of data passed to the protocol stack by the application, and follows a saw tooth structure curve. Measurements also show that the asymptotic throughput of TCP/IP/ATM was as much as two-thirds slower than API/ATM, although previous studies indicate this is due largely to the implementation of the protocol rather than the protocol itself. Measurements show that using the ATM API about 83 percent of the architectural bandwidth limit of the processor is asymptotically achieved.

Communications throughput is insensitive to the amount of data to transmit. Although insensitivity to AAL5 vs. AAL3/4 and the use of an ATM switch was also measured, these parameters could not be stressed due to the fact that the processors in this study were not fast enough to load the underlying network. If faster machines were used, one would expect AAL5 to yield somewhat better performance than AAL3/4, but with no measurable effect from the switch. Due to the lack of flow control, cases are encountered where cell loss occurred.

"Demultiplexing on the Adapter, Experiments with Internet Protocols over ATM" — Ernst W. Biersack and Erich Rütsche; Institut Eurecom, France

This work performs a user-level protocol implementation of TCP/IP over ATM to take advantage of the demultiplexing functions that are often implemented in today's high-speed ATM adapters for an optimized implementation.

The Internet protocol stack is implemented in two steps. In the first step the code of IP, UDP (user datagram protocol), and TCP of the 4.3BSD version of Unix is ported to the user space of a Sun 2 running SunOS Version 4.1.3. The goal is to build a library such that any application can chose either the system protocol stack or the library without any change. The memory management of the protocol stack is changed to use the `mbuf` structures on memory blocks gotten from a `malloc` system call. The socket copies the user data to be sent into these memory structures where the subsequent protocol processing takes

Quite a few submissions had to do with TCP/IP over ATM and B-ISDN, reflecting the emerging view that the Internet and ATM communities will have to try to coexist in the future.

In the near future, a large portion of the traffic carried by ATM networks will be generated by applications written to run over a TCP/IP protocol stack.

place. The protocols build packets by chaining the data and the headers in `mbufs`. Data to be sent are written over a standard I/O interface to the AAL, that is implemented on a Fore Systems SBA200 network interface board. A select call signals the reception of data on the same interface. `Pthread` is used to implement a parallel timer thread that watches all outstanding timers.

In the second step, protocol implementation is optimized for the direct mapping of a transport connection on an AAL connection. Therefore a packet filter is developed that looks for all the fields in the PDU (protocol data unit) that can be precomputed once the connection is established. As the addresses are known *a priori*, the filter matches only the addresses that could change, e.g., IP segmentation and options, TCP flags and windows, and UDP flags. For each packet that is successfully received, the filter is updated to match the next packet. The algorithm is similar to the header prediction algorithm of TCP by McCanne and Jacobson, but is enhanced by the filtering of the IP header. After the filter a number of functions can be suppressed, e.g., connection lookup, option, window adaptation, and PCB searching, and the structure of the call can be simplified. If the precomputed filter does not match, this means that at least one of the precomputed assumptions does not hold. The incoming PDU is then processed through the normal path.

The performance of the user-level protocol stack with IP and UDP is 2.5 MBytes/sec compared to 3.3 MBytes/sec of the system stack with an packet size of 2 KBytes. The lower performance of the user-level implementation is due to the higher number of copy operations.

"Throughput Degradation for TCP over ATM in the Presence of Traffic Policing" — Partho Pratim Mishra; AT&T Bell Laboratories, USA

In the near future, a large portion of the traffic carried by ATM networks will be generated by applications written to run over a TCP/IP protocol stack. One of the key factors that will impact user and network performance in this environment is how well the traffic management mechanisms used in the IP and ATM networks mesh together in providing good end-to-end performance.

This work considers some undesirable interactions between the congestion control scheme used in TCP and the policing mechanisms used in ATM network that can significantly degrade the throughput of TCP traffic. The performance degradation effects described here are distinct from those shown by Romanow and Floyd [3], although the effect of TCP packet to ATM cell fragmentation plays a key role in both results.

The traffic management mechanisms that will be used in ATM networks are still in the process of being standardized. However, one of the central elements is the use of policing mechanisms to enforce a user behavior envelope on the traffic injected by individual connections. The UNI (user-network interface) 3.0 standard specifies the use of a dual leaky bucket policing mechanism to enforce peak and sustainable cell rate (PCR and SCR), which correspond to the desired peak and average rates for a connection. The values of the PCR, SCR, and the associated leaky bucket burst tolerances (token bucket size) are specified at

rtt	4 ms		40 ms	
size	500	4000	500	4000
Throughput (Mb/s)	1.71	1.60	3.33	3.32
Goodput (Mb/s)	1.48	1.27	3.28	2.86

rtt: round trip time; size: CP packet size in Bytes
SCR = 7.68 Mb/s; burst tolerance = 1000 cells;
bottleneck rate = 15 Mb/s

■ **Table 1.** *TCP over ATM with traffic policing.*

connection setup time. The congestion control scheme used in TCP is the primary traffic management mechanism used in IP networks. This algorithm is designed to modify a TCP connection's window size to match the bottleneck bandwidth using packet loss as an implicit signal of congestion. The evolution of a TCP connection's window size is cyclical: the window size is initialized to one packet following a packet loss and is increased on the receipt of every acknowledgment. The achievable throughput depends on the average window size over a cycle and how quickly packet loss is detected.

In the absence of policing, a TCP connection increases its window size rapidly, following the detection of packet loss, until it crosses a threshold. Subsequently, the window size increases by one packet per epoch duration — an epoch refers to the time taken to insert and receive acknowledgments for an entire window of packets. Initially the duration of an epoch is approximately one round-trip time. Once the window size matches the bottleneck capacity times the round-trip time, packets start accumulating at the bottleneck and the duration of each epoch starts increasing (because of the queuing at the bottleneck). As a result, the rate at which packets are injected by the sender increases quite slowly once the bottleneck capacity is exceeded. Due to the gradual increase in the window size, the connection accumulates packets in the bottleneck buffer until packet loss occurs. Since the window size is increased by one packet per epoch, only a single packet is lost due to buffer overflow and is detected within an epoch at the sender, through the receipt of duplicate acknowledgments.

In the presence of policing, once a TCP connection has increased its window size beyond the SCR times the round-trip time, the virtual leaky bucket buffer starts building up. If the bottleneck capacity exceeds the SCR there is no physical queuing at the bottleneck and hence the duration of the epochs stays about the same. As a result, the same window size increase translates to a higher rate increase causing the (virtual) buffer to fill up very quickly. Once the virtual buffer is full, any cell arriving when there are no tokens available is dropped. Since tokens arrive at the SCR, while cells arrive at a rate R which is greater than the SCR, only a fractional number of the cells SCR/R constituting each TCP packet are allowed into the network, while the rest are dropped. As a result, all of the packets in the TCP window are is dropped and the sender can only detect packet loss via a timeout. Most TCP implementations use relatively coarse grained timers, and as a result timeout intervals are typically 500 ms or 1 s. Thus, the net effect is that a TCP connection sends

a short burst of packets, suffers packet loss, and is then forced to idle for the duration of the timeout interval. This causes the average throughput to be significantly lower than the SCR value, as shown in Table 1. Moreover, a large number of packets may need to be retransmitted causing a significant waste in bandwidth (the difference between throughput and goodput values in Table 1).

Setting the leaky bucket to allow larger bursts through improves the throughput seen by individual connections: with a burst tolerance of 10,000 cells and a packet size of 4000 Bytes, the average goodput increases to 4.20 Mb/s and 5.96 Mb/s for rtt (round-trip time) values of 4 and 40 ms, respectively. However, given the small amount of buffering available for most ATM switches (typically less than 10,000 cells) this approach can lead to massive cell loss if sources become synchronized. Reducing the TCP timeout interval can also ameliorate the problem: using a lower bound of 50 ms rather than 500 ms for the timeout interval causes the goodput to increase to 6.45 Mb/s (rtt = 4 ms, size = 4000 Bytes). However, this solution may be impractical to implement, given the limited software timer resolution of many Unix-based operating systems. A reasonable solution is to use either smarter policing or cell-level traffic shaping schemes; both these techniques improve performance significantly.

"GIPR: A Gigabit IP Router" — Guru Parulkar, Douglas C. Schmidt, and Jonathan S. Turner; Computer and Communications Research Center, Washington University in St. Louis, USA

The Internet protocol suite provides the foundation for the current data communications infrastructure in the United States and much of the rest of the world. The IP protocols have proven to be very flexible and have been deployed widely over the past two decades. As technology makes it possible to communicate at gigabit speeds, it is essential to create scalable, high performance routers that implement IP protocols. In the past ten years, ATM technology has emerged as a key component of next-generation networks. ATM offers unprecedented scalability and cost/performance, as well as the ability to reserve network resources for real-time oriented traffic and support for multipoint communication.

Although IP and ATM often have been viewed as competitors, their complementary strengths and limitations form a natural alliance that combines the best aspects of both technologies. These complementary strengths and limitations make it natural to combine IP with ATM to obtain the best that each has to offer. The Gigabit IP Router (GIPR) effort represents research on new methods and architectures for achieving the synergistic combination of IP and ATM technologies. A highly scalable Gigabit IP Router based on an ATM core has been designed. This router integrates the following core architecture components:

- A gigabit ATM switching fabric that is highly scalable in terms of the number of ports and provides optimal hardware support for multicasting.
- A multi-CPU embedded system that includes a string of ATM Port Interconnect Controllers (APICs) and allows flexible and high performance IP packet processing in software.
- A distributed software system capable of forwarding IP packets at gigabit data rates on the ATM substrate and configuring that substrate dynamically to provide efficient handling of IP packet streams.

This work seeks to go well beyond the conventional approach of implementing IP over ATM in a strictly layered fashion. By allowing the IP processing layer to directly control and manipulate an underlying ATM switch core, IP can directly benefit from the hardware processing efficiencies of ATM switching technology; looking at it from the other perspective, ATM can enjoy the inherent flexibility and adaptability that are among IP's greatest strengths.

The GIPR architecture is designed to allow experimentation with, and fine tuning of, the protocols and algorithms that are expected to form the core of the next-generation IP in the context of a gigabit environment. The underlying multi-CPU embedded system will ensure that there are enough CPU and memory cycles to perform all IP packet processing at gigabit rates. The GIPR architecture will not only lead to a scalable high performance Gigabit IP Router technology, but will also demonstrate that IP and ATM technologies can be mutually supportive.

Discussion Session — ATM vs. IP?

This discussion session was intended to consider not only the relative merits of ATM/B-ISDN vs. IP as a network (layer 3) and internetwork (layer 3.5) infrastructure, but also since there were so many ATM contributions in the workshop, we decided to have Henning Schulzrinne present an IP-centric critique of ATM to incite the discussion.

"ATM: Dangerous At Any Speed?" — Henning Schulzrinne; GMD FOKUS, Germany

Since ATM has been the object of much hyperbole as *the* or even *the one-and-only* future network technology, it is tempting to summarize some of the open issues and, more importantly, principal limitations of the technology. Surprisingly, there has been little published in the technical literature concerning the demerits and problems of ATM [1-3]. Some of the problem areas are shared with its two older siblings: X.25 and ISDN (Q.931) signaling.

In the following, we summarize some of the issues that may interfere with widespread deployment, particularly in heterogeneous nets owned by different service providers. Using ATM permanent virtual circuits as links between routers is proven technology and poses fewer challenges, except in the area of traffic integration. As with any network technology, ATM is evolving, so that some of the reservations may be moot in future revisions and capability sets. Indeed, important changes in ATM perception and technology have already taken place. To name but a few: the reduction from four to two adaptation layers (AAL1 and AAL5), the use of ATM over ever lower bit rates, the lack of use for the generic flow control (GFC) header field, the rare use of SMDS (switched multimegabit data service) over DQDB (distributed queue dual bus) multiple access channels, and the abandonment of the original service classes for more closely defined QOS parameters. But the

ATM has been the object of much hyperbole as "the" or even "the one-and-only" future network technology.

Can ATM or IP rationally coexist in the future Global Information Infrastructure as the successor to the current Internet?

evolvability of ATM certainly also applies to missing features of IP and its implementation in high-speed networks.

Architectural Issues for ATM Internetworks — ATM contributes to a continuing confusion over the network model. Together with the AAL, it is not simply a data link layer, and offers important network layer functionality like routing and global addressing, but it is far from clear whether it can be carried efficiently over as wide a range of media as traditional internetwork layer protocols like IP, without sacrificing important data link layer functionality such as anonymous multicast. Running ATM cells over important local area networks like Ethernet is feasible, but the architectural and efficiency implications must be carefully examined.

The division of public networks and customer premise equipment (CPE) reflects traditional telecom views and is often artificial. For example, what what may appear as a UNI to the carrier is actually a network-to-network interface to the local ATM cloud. Is the distinction really necessary (the Internet lives without it)?

The idea of connectionless service as espoused by the ITU-T (International Telecommunication Union — Telecommunication Standardization Sector) seems to merely duplicate the functionality of a classical IP network. Since applications are unlikely to use the ATM connectionless service directly, adding it on top of ATM merely adds it and an encapsulation layer between the network and the application, reducing efficiency and making management more difficult. Thus, carriers might be better off offering private IP networks to their customers. In the local area, the benefits of LAN emulation over, say, switched 10 or 100 Mb/s Ethernet seem minor, but the complexity in specification and implementation need to be carefully weighed.

Cell Format — The earliest objections to ATM by Cidon *et al.* [1] centered around the small cell size. Overhead is a particular concern for low-speed links and small IP packets; for example, packet voice over IP over ATM wastes about 40 percent on packet headers, not counting SONET overhead. Furthermore, Sterbenz [2] points out that the small cell size not only limits speed from below, but also from above, as cell switching times approach nanoseconds. Chopping packets into cells also causes loss of a whole packet for each cell loss, potentially leading to a form of congestion collapse due to retransmissions.

The partial packet discard and early packet dropping mechanisms suggested by Romanow and Floyd [3] cure this problem, but imply that cell switching has to be aware of the AAL type. Also, packet-aware mechanisms are feasible only if the number of different AALs remains very small. This, and the desirability to implement packet reassembly in network adapter hardware, is likely to limit ATM AALs to AAL5, with niche uses for AAL3/4 and AAL1.

The original intention for ATM was to have extremely low packet loss due to both the use of fiber and careful traffic control. Both sources of loss would appear to be higher for ATM as a universal packet layer, running over less reliable media and handling more bursty data. Is is unclear whether the cell size issue can be reconsidered, since existing switches are hard wired for the current size (but if new switches were to also support larger cells connections using only these switches could use larger cells).

Connections and Signaling — Due to the small cell size, ATM can only operate in a connection-oriented mode, where signaling establishes the path for data cells switched by short VC (virtual channel) identifiers. The performance and features of the network are largely determined by those of the signaling protocol.

The currently standardized signaling protocol (Q.2931) is extremely heavyweight, with numerous timers, protocol states, and a difficult-to-parse packet structure featuring BCD numbers and 7-bit bytes. The several round-trip times and the processing costs impose a heavy delay burden on short-lived associations like RPC (remote procedure call) and query-response applications such as HTTP (hypertext transfer protocol). The overhead gets worse if one or more name resolutions (URN to URL) are needed in the context of the World-Wide Web and other applications that we expect to inherent this naming and location model.

Currently, there is no security on call establishment, unless you trust all network operators along the path. There is also no provision for per-AAL-packet authentication, thus "stealing" VCs is possible even with authenticated call setup.

Just like AAL5, as a "simple and efficient adaptation layer" (SEAL), more or less replaced the cumbersome AAL3/4, a lightweight signaling protocol, unifying UNI and NNI protocols and operating within a single round-trip time is called for. A processing efficient signaling protocol is also important if end systems dumber than workstations are to be directly connected to ATM networks, e.g., as suggested by the concept of desk area networks.

The limited number of VCs available (and thus the likely resulting charge for holding a VC even without generating traffic), both due to cell header and ATM switch implementation restrictions, force relatively frequent signaling within a single upper layer (e.g., TCP) connection. Some such connections can last days, and can be inactive for most of the time. Even with a timeout of 64 seconds, a current IP router may see 2000 flows. Typical current ATM switches have a capacity of 4096 VCs, which seems inadequate. With IPng (IP next-generation) flow labels, similar flows can be aggregated; there is no mechanism yet to aggregate VCs into VPs (virtual paths). If VCs are charged for setup and simply for being there, applications will have to learn subtle optimization strategies, depending on the current tariff structure, such as trying to guess when it is better to drop a VC and reconnect later. In contrast, with TCP, transient network failures can be hidden as long as the end system stays up. With ATM, applications may have to reconnect on VC failure. After a backhoe fade, this would yield massive signaling load with all applications trying to reestablish connectivity ("redialing"). Probing packets (à la TCP) is a lot simpler than trying to decide whether to abandon a connection. (To avoid ATM layer

signaling for local failures, lower layers like SONET or fiber switching are likely to handle fiber cuts.)

It is also not clear what happens if an end system or application crashes without cleaning up connections. Will this have the same undesirable consequences as leaving the phone off the hook when calling New Zealand from Europe? What happens to resource reservations (and charges) if the access link is down, so that the end system cannot tear down a connection?

Unlike other technologies such as Ethernet, Token ring, or FDDI, ATM is point-to-point only and requires additional signaling support for multicast. Currently, only root initiated joins are possible; even with leaf-initiated joins in the ATM Forum UNI 4.0 specification, explicit knowledge and management of source and sinks is still required, with the attendant management and rendezvous problems. In ATM, joining a 1000-member multipoint-to-multipoint group takes 1000 `ADD PARTY` messages and responses. For example, for a distance learning application, the teacher needs to find all student receivers, where the number of students is dynamic. Having a student ask a question could require all other students to set up connections to the interlocutor. Contrast that with IP multicast, where senders do not need to know the identity of potential receivers and receivers specify a single group address to receive packets from any number of senders. Also, for IP multicast, routers only need to maintain state per group, while ATM requires per sender state.

ATM as Part of the Internet — Pure ATM solutions come in two degrees of purity: applications directly on top of the adaptation layer, or end-to-end ATM with a traditional stack (e.g., TCP/IP) on top. Using ATM within an Internet (today's and particularly a more ATM dominated one), poses additional problems, beyond the current packet loss delaying deployment of the ATM-based NAPs (network access points) in the Internet. For example, there now is a separate address space, that of E.164 numbers embedded in NSAPs (network service access point). Previous experience with NSAP assignment has shown severe scaling difficulties with the purely organizational rather than topological assignment of NSAPs. It would be helpful if the management of a separate namespace, above IP numbers, could be avoided by appropriate management of the NSAP space. Otherwise, another layer of global address lookup becomes necessary, yielding a host name — IP address — ATM NSAP chain. If large numbers of local ATM systems are assigned numbers that are not globally and topologically meaningful, the problems faced by the interconnection of IPX and AppleTalk networks will be repeated.

There are a number of operational and network management considerations, for example, there are no ATM tools like `ping` or `traceroute`, and the the necessary protocol and switch support for such tools is missing.

Formerly, X.25 showed that interconnecting connection-oriented networks is more difficult and more fragile than connectionless networks. It will remain to be seen whether ATM improves upon this record.

Overall, from a purely technical standpoint, it appears we could get the (relatively) cheap switching of ATM by simply using application-sized frames at the host and on the wire and cells of whatever size within switches. The queuing delays due to long packets are not significant at optical speeds; at lower speeds, ATM overhead becomes noncompetitive. Generally, if bandwidth is cheap, ATM multiplexing is not needed since we can simply use fixed-size pipes; if bandwidth is expensive, we may not be able to afford the ATM overhead.

Discussion

The reason this session was titled "ATM vs. IP?" was to not only discuss whether ATM or IP was a better solution, but to consider whether ATM and IP could rationally coexist in the future Global Information Infrastructure as the successor to the current Internet.

The IP-centric view was presented by Henning. We did n0t specifically have an ATM-centric presentation and had no defenders of "everything over ATM solves the world's problems" (hopefully it was not that they were too intimidated by the IP-centric presentation). The discussion ranged from the IP-centric view to the idea that the IP and ATM communities would merge and provide a common solution (but not necessarily *current* IP or IPng over *current* ATM). The contribution by Parulkar and [2] also present this latter view.

There was general agreement that the problems that are being solved are the same, regardless whether it is the IP or ATM communities providing the solution. Furthermore, there was significant opinion that ATM and IP can't coexist well as currently implemented.

One problem is that both ATM and IP implement network layer functionality, and in a manner that is not terribly compatible. Note that we are including the AAL, VC routing, and signaling along with ATM cell relay when considering ATM network layer functionality. If multiplexing is done in ATM, it can't be accessed in IP, and vice-versa. The same is true for signaling. The control mechanisms of ATM have not been designed for interaction with TCP (and of course TCP and most of its optimizations predates ATM). Examples of this problem are given in the contribution by Mishra and in [3, 4].

The issue of connectionless (IP) vs. connection-oriented (ATM) is generally regarded as one of the most fundamental differences. We generally agreed that both types of services need to be available to the end user, and is it possible to offer connections over a connectionless infrastructure (e.g., TCP over IP) and datagrams over a connection-oriented infrastructure (as ABR traffic will do in ATM).

The real question has to do with performance: How well can datagrams operate over ATM, particularly for transaction type services such at HTTP in the World-Wide Web (where response to clicking on a link should be subsecond), and of control messages requiring low latency with reliable delivery such as for session control (higher level than ATM signaling)? How well can connection-oriented multimedia traffic operate over IP, particularly in the presence of congestion? How well will these two traffic types interact over either infrastructure?

There was general agreement that the problems that are being solved are the same, regardless of whether it is the IP or ATM communities providing the solution.

In the near term, the evolution of the Internet appears to be in the direction of IPng over an increasing number of ATM subnets; there was far less agreement on how this will evolve in the long term.

We also generally agreed that state is needed in the network either way. In the case of a connectionless IP network, state can be used to to allow reuse between datagrams. In the case of a connection-oriented ATM network, state can be used to cache between short-lived connections to provide a datagram-like service. We note that this soft state caching only helps ATM for long streams (connection-like or connection-oriented) and reuse of host pairs, and will be of limited value for World-Wide Web browsing, particularly where the ratio of traversed links per host is small.

It is clear that IP provides a useful operational infrastructure for higher layer protocols and applications, whatever its shortcomings may be for emerging connection-oriented applications. Similarly, ATM provides a high data rate switched infrastructure that is being widely deployed, even in the face of technical flaws and details that have not yet been resolved in signaling, traffic management, and network management. Thus in the near term, the evolution of the Internet appears to be in the direction of IPng over an increasing number of ATM subnets. There was far less agreement on how this will evolve in the long term, and how the telephone, CATV, and entertainment broadcast networks will be integrated into a single infrastructure.

Topics in Gigabit Networking

The last session consists of contributions that are concerned with gigabit networking issues independent of the network layer below (even though all but one do use ATM for at least part of the network infrastructure).

"Packet-Switched Service Over A Dynamically Reconfigurable All-Optical Network" — D. Marquis, S. A. Parikh, S. G. Finn, R. A. Barry, D. M. Castagnozzi, B. R. Hemenway, M. L. Stevens, E. A. Swanson, R. Thomas, C. Ozveren, and I. P. Kaminow; MIT Lincoln Laboratory, Digital Equipment Corporation, and AT&T Bell Laboratories, USA

This project is developing a means of dynamically reconfiguring a high-speed all-optical circuit switched network infrastructure supporting a packet switched network. The work is being developed in a wavelength division multiplexed (WDM) all-optical network (AON) testbed being developed by AT&T Bell Laboratories, Digital Equipment Corporation, and the Massachusetts Institute of Technology under a grant from ARPA. This capability allows efficient allocation of network resources by reconfiguring variable bandwidth optical trunks to adaptively adjust to the offered load presented by the electronic packet switches.

Typical packet switch networks operating over traditional telecommunication networks use routers to provide packet service over fixed bandwidth-leased carrier services. This approach suffers where either the offered load is small compared to the available bandwidth (wasted carrier resources), or where it is large (users denied service). It would be better to dynamically reconfigure the carrier services supporting the operating packet switched network, provisioning bandwidth as needed based on offered load at the packet switches. Ideally, the reconfiguration would happen so as to have minimal impact on the higher layers of the packet network.

The system considered uses a resource allocation application running on a workstation connected to the control and management network of the AON, and would be invisible to the packet switched users. The application uses a network management interface (SNMP — simple network management protocol) to connect to both the commercial high-speed packet switches (in this testbed, a DEC Gigaswitch) and the bandwidth provisioning scheduler in the AON. At periodic intervals the network management application polls the packet switch to estimate changes in offered load. The application then uses SNMP requests to reallocate bandwidth to the packet switches as necessary from the resources of the all-optical network. If additional bandwidth is indicated, circuit switched path bandwidth is added as some combination of additional wavelengths and additional time slots within the optical network.

One of the key enabling technologies for this project is the AON *B-service*, which can provide wavelength-division and time-division sharing of resources among up to 128 users. The B-service can simultaneously provide bandwidth for independent users of differing data rates that are optically routed throughout a metropolitan area while sharing network resources such as terminal equipment and wavelengths. Because B-service switching times are small, reconfiguration of the network infrastructure can be handled so as to minimize negative effects on the higher layers of the network.

This is the first demonstration of autonomous dynamic bandwidth provisioning of a high-speed all-optical network based on the demands of a packet switched network. This capability may prove important in future networks that employ a combination of packet switched user services and circuit switched physical transport. This technique can make more efficient use of available carrier bandwidth and may decrease carrier service costs during times of reduced load.

"The Failure of Conservative Congestion Control in High-Speed Networks" — Hyogon Kim and David J. Farber; Distributed Systems Laboratory, University of Pennsylvania, USA

This work shows the failure of conventional conservative congestion control (e.g., using additive-increase/multiplicative-decrease) to exploit network bandwidth efficiently in high-speed networks. A new approach that scales better with increasing bandwidth and yields an order of magnitude higher performance is proposed. Existing performance analysis on the effect of a large bandwidth-delay product shows that the conservativeness of rate adaptation mechanisms adopted by most frequently used protocols such as TCP and ATM flow control algorithms becomes increasingly expensive for the transport of the relatively small packets of today's networks. Only window based control (e.g., TCP) is considered in this study, but we conjecture that this argument also applies to rate based control using conservative rate adaptation.

Non-adaptive non-conservative control, which would have caused more severe congestion in small bandwidth-delay product networks in the past, can solve the problem in large bandwidth-delay product networks. In this approach, the traffic sources

transmit data in large bursts without being regulated by conservative rate adaptation algorithms and without depending on network feedback. The packets in the burst are ordered with decreasing priority so that network switches can discriminate against longer bursts in favor of new, short bursts and interactive traffic. This approach contrasts with the conventional approach in that the sources are equally aggressive, rather than equally conservative. The intuition is that as propagation latency increases, conservative traffic sources cannot exploit the bandwidth quickly enough, especially when the transported data size is relatively small compared to the bandwidth-delay product. Efficient use of network bandwidth can only be achieved with the aggressiveness of the traffic sources, which is moderated by the network switches rather than by the sources.

The performance of a new congestion control method *Blitz* based on this new approach is compared to that of TCP Reno by simulation. The data size distribution conforms to Caceres's empirical statistics, subgigabit (150 Mb/s) and gigabit network rates, propagation delays sized to the NSFNET and AURORA testbed, and various switch buffer sizes. With a rate of up to a few thousand arrivals per second to load the network, Blitz outperforms TCP Reno by a factor of 10 or more in almost all configurations, where the performance is measured in network power [Byte/sec^2]. For instance, in a gigabit NSFNET-sized network, the network power of Blitz was 17 times that of TCP Reno, while in an AURORA-sized network with 150 Mb/s (where the bandwidth-delay product is more than order of magnitude smaller), the power of Blitz was four to five times larger. More importantly, the performance gap widens with increasing bandwidth-delay product, i.e., Blitz shows much better bandwidth scalability than TCP. It appears that the performance gap will widen further in multi-gigabit networks.

A pure, aggressive version of Blitz (henceforth *Blitz-x*) has been tried which lacks the burst prioritizing and the complicated queuing discipline for fair distribution of bandwidth. One notable result obtained is that Blitz-x yields almost the same network power and fairness as the original Blitz when the network bandwidth is large enough so that there is no physical bottleneck that causes chronic congestion. This result is very encouraging because it opens an opportunity of smooth interoperability between conventional TCP and Blitz. When network load is high, conventional TCP can be used, with a return to Blitz-x when the overload disappears. This issue is under further examination.

The results of this investigation strongly imply that the conservative principle should be discarded when the network bandwidth increase changes the delay model in future high-speed networks. The performance gap between the new approach and the old approach is a function of bandwidth-delay product and transported data size. As the bandwidth-delay product grows the performance gap increases as well, while the larger data size decreases the performance gap. While the data size distribution is expected to change only slowly, the bandwidth continues to increase by orders of magnitude factors; this is a promising approach in this context.

"A Distributed and Cooperative Broadband Service Management Architecture: North Carolina Information Highway Experience" — Hao-Jen Fu and Renu Chipalkatti; GTE Laboratories, USA

With the proliferation of ATM networks and the offering of a multitude of services such as video conferencing, frame relay, SMDS, and customer network management (CNM), there is a critical need for effective operations support. The management of these ATM services offers a variety of challenges, particularly since ATM deployment is proceeding even while standards are still evolving. The fact that these ATM services are often provided across LEC (local exchange carrier) boundaries further complicates the approach that service providers have to take to manage broadband services.

A distributed approach is proposed for managing broadband services from end to end, transparent of the network domains of the service providers. It is based on the experience gained from developing a management system for the Distance Learning service in the North Carolina Information Highway (NCIH).

The North Carolina Information Highway represents a major step in the deployment of broadband networks, bringing many concepts from research to the field. One of the initial major service offerings of the NCIH network is the Distance Learning service. This is a real-time, interactive, multimedia conferencing service among multiple, remote sites. The goal is to provide a remotely located teacher and student classroom sites with the full set of capabilities necessary to conduct a "normal" classroom session. For the Distance Learning service to be useful, it must be provided ubiquitously across the multiple LECs participating in the NCIH network, namely, GTE, BellSouth, and Carolina Telephone. Consequently, the operations and management of the service must be provided cooperatively by the three LECs.

The operations architecture currently adopted for the NCIH Distance Learning service is based on a centralized control through a scheduler that interacts with individual LEC's operations system to set up and tear down connections. This approach lacks generality, scalability, extendibility, and reliability needed for effective management by either the carriers or customers.

In the distributed architecture, the operations systems of the individual carriers are responsible for the scheduling, resource reservation, and connection management of the Distance Learning classes. The operations systems cooperate using well defined interface to manage the Distance Learning service. This is also imperative for the CNM service.

"Network Integrated Processing" — Joseph B. Evans, Douglas Niehaus, Victor S. Frost, and David W. Petr; Telecommunications and Information Sciences Laboratory, University of Kansas, USA

This work develops methods of fully integrating general purpose processing and memory elements with gigabit local and wide area networks. A distinguishing aspect is the innovative use of the virtual circuit capabilities of B-ISDN services provided by ATM networks to provide connectivity, in combination with new approaches to protocols, buffering, and caching. This approach is called Network Integrated Processing (NIP), since it eliminates the dichotomy between the processor and the network.

The class of distributed computing applications can consume a high percentage of the available network bandwidth.

Distributed information services are a major area of gigabit applications.

The NIP architectural approach creates a distributed computation environment within which:

- Components of a computation can exchange data across thousands of miles as easily as they can across a room.
- Distributed computational architectures using tens or hundreds of nodes from a pool potentially containing millions are dynamically composed and used.
- Additional resources can be attached at any network location while still providing uniform access and thus provide support for scaling.

The NIP architecture supports several novel capabilities, including:

- Reduction of latency associated with distributed computations by reducing the overhead of the network–processor interface and by developing protocols for buffering and caching data which take advantage of the unique properties of the ATM interconnection facilities.
- Creation of a distributed computing environment supporting the description and compilation of a computation as a set of communicating components that can be mapped onto the processing architecture composed from the pool of NIP processing elements.
- Methods for rapid and reliable estimation of computation component properties, and binding of components and data to NIP resources.

The initial NIP prototype architecture consists of 10 to 15 commodity processing elements, such as DEC Alpha or Pentium motherboards, and large (at least 1 GB) shared memory resources, connected via a common interface to the DEC AN2 ATM switch fabric operating at 800 Mb/s. Specialized processing resources such as FPGA (field programmable gate array) coprocessor boards are also envisioned. The use of the DEC AN2 switch provides a guarantee of no congestion induced cell losses in its environment, which allows aggressive use of lightweight protocols between NIP nodes. The NIP architecture includes a network virtual address (NVA), with memory consistency models which vary according to the type of resource being accessed. Support for existing protocols is provided by mappings within the NIP node operating system.

This research is being driven and validated by its application to selected problems from the MAGIC gigabit testbed and DREN (Defense Research and Engineering Network) testbed community. This environment provides the opportunity to evaluate the latency reduction capabilities of NIP within a wide area network, as well as to explore the concept of offering computational and network services as part of a unified offering.

"Traffic Characterization of Gigabit Applications" — Peter Steenkiste; Carnegie Mellon University, USA

The Gigabit Nectar testbed consists of two local area networks, one on the Carnegie Mellon University (CMU) campus and one at the Pittsburgh Supercomputer Center (PSC), connected by a 26 km MAN link. The LANs are based on HIPPI, while the link between CMU and PSC consists of both an experimental ATM over SONET and a commercial HIPPI link. A variety of computing resources are connected to the network, including the C90-T3D and CM2 at PSC and the iWarp and workstations at CMU. The testbed is a joint project with CMU, Bellcore, PSC, and Bell Atlantic.

The testbed is now operational and a variety of applications have made use of the network. They range from traditional network applications (FTP, Mosaic, etc.) to distributed computing applications that use the network more aggressively. The class of distributed computing applications is of particular interest, since they can consume a high percentage of the available network bandwidth. Since there is relatively little experience with this class of applications it is valuable to look at their traffic characteristics and to assess their impact on the network.

The applications executed over the testbed fall roughly in three categories: heterogeneous supercomputing applications that are distributed over a small number of supercomputers, workstation cluster applications that have fairly predictable traffic patterns, and workstation cluster applications that have a lot of data dependent communication.

Heterogeneous supercomputing applications combine a small number of supercomputers. They exploit very coarse-grain parallelism, i.e., large blocks of computation separated by the exchange of very large data sets. A chemical flow sheeting application that is distributed over the Intel iWarp on the CMU campus, and the C90 and CM2 at the Pittsburgh Supercomputer Center (PSC) has been characterized.

Workstation cluster applications typically have a finer granularity. They often have a very regular traffic pattern, which follows directly from the way they exploit data parallelism: the application is organized as a sequence of alternating compute and communicate steps. The communication phase involves collective communication (all nodes exchange data). An environmental modeling application running over an Alpha cluster has been measured.

Finally, the data-dependent cluster applications typically has data dependent communication superimposed on the regular communication pattern. Traffic measurements for the Dome environment executing a molecular dynamics computation have been done. The data dependent computation is a result of dynamic load balancing.

"Progress on the GBN '94 'Five Challenges That Define High-Speed Protocols'" — Joe Touch; HPCC Division, USC/Information Sciences Institute, USA

At GBN '94, five techniques were presented that challenge the definition of gigabit protocols research, as distinguished from existing high performance systems issues. An application that meets any of these challenges can be solved by known methods:

- Increase the clock rate.
- Multiplex.
- Use large payloads.
- Increase the window size.
- Relocate everything.

The "World-Wide Web as a Distributed Application" was presented at GBN '94 as passing all five of these challenges.

The ARPA/NSF Workshop on Research in Gigabit Networking (WRGN) [6] helped confirm the assertion at GBN '94, that distributed information services are a major area of gigabit applications. It presented a taxonomy of gigabit applications:

gigabit-enabled	exist only at Gb/s
gigabit-challenged	radical revision for Gb/s
gigabit-enhanced	require Gb/s to be practical

The World-Wide Web as a networked browsing tool (point, click, and wait) is an example of a gigabit-enhanced application. The Web as a distributed application (point and click) is gigabit-challenged. By "gigabit," WRGN intended a combination of high bandwidth and high bandwidth-delay product.

In the past year, the development of low latency distributed information access via the Web has proceeded. Measurements of Web server logs indicate that a 1/3 reduction in request/response latency is possible for a seven times increase in bandwidth. The request/response latency was reduced to less than one round-trip time. These measurements indicate that a 56 Kb/s ISDN line responds interactively (within 200 ms, including propagation delay) with a probability of 14 percent, but that this can be increased to 83 percent with source preloading. A source preloading mechanism that substitutes for a proxy cache, and that supports server-side preloading of a receiver-side cache in background between requests has been developed. This mechanism is being implemented now, and measurements should be available by GBN '96.

In addition, this work is investigating how to scale source preloading to serve a large, city-wide client population. Methods for Intelligent Bandwidth (IB) to support information distribution aggregation via multicast media and asymmetric channels are being developed. IB is *middleware* that uses network feedback on latency, bandwidth, power, and topology to govern the efficient distribution of shared responses. It also aggregates responses over a request collection interval, to amortize channel load and use multicast and broadcast capabilities.

Discussion Session — Middleware

A number of recent and upcoming conferences have considered middleware issues, so we decided to have Joe Touch lead a discussion on the issue of *middleware*, and how it is related to gigabit networking.

"Middleware Issues" — Joseph D. Touch and James P. G. Sterbenz; USC/Information Sciences Institute and GTE Laboratories, USA

GBN '94 was dominated by interactive video applications; GBN '95 was dominated by ATM and TCP/IP over ATM issues. A number of recent and upcoming conferences have considered middleware issues, so we decided to hold a discussion on the issue of middleware, and how it is related to gigabit networking.

Although most of us seemed to know what middleware was (or at least would know it if we ran into it), we had a difficult time coming up with a concrete definition. Research after the workshop indicated that the term middleware is of dubious origin; we found references back to 1992, from a variety of disciplines, but it is believed to have originated at least as early as 1990.

The relationship of middleware to gigabit networking has been raised in various forums in addition to GBN '95, including The National Research Council's NRENAISSANCE Committee study of the future issues in Internet research [5], the ARPA/NSF Workshop on Research in Gigabit Networking [6], and the SIGCOMM'95 workshop on middleware issues.

The ARPA/NSF workshop indicated that middleware is a major gap in current research. It defined middleware as a set of shared resources and services that enabled coordinated application use of network and OS resources. In this report, middleware is considered to conventionally refer to a set of toolkits used by application designers. A different sort of intermediate infrastructure is required, that which manages network capabilities and feedback in relation to application capabilities and feedback.

The ARPA/NSF report argues that applications need to be environment-sensitive, that the network should provide "knobs and dials" for the application, and that the application should do the same for the network. Middleware can then be considered an intermediate layer between the two. The network and OS have their notion of the space of tunable parameters and feedback, as does the application. Middleware may be a layer that translates between these parameter spaces. But some contend that it is more, that it also includes persistent state and process to assist with this translation, i.e., to perform resource management.

The SIGCOMM Middleware Workshop has yet to occur, but the call for participation defines middleware as infrastructure that: integrates, raises the level of common services, and introduces new types of higher level services. It considers middleware as an interface between the network and the application, a network version of an operating system.

An attempted definitive description of middleware describes it as the layer between an application and the network and OS [7]. This layer is characterized further by a set of standard programming interfaces and protocols. In this case, middleware can be viewed as a standard interface to the OS, and as a layer 7 (application) interface to the network.

They also describe middleware as implemented by a client/server system that is a client to the OS and a server to the application. This is the first case where middleware is described as having both a protocol and persistent state and process. The persistent state is shared among applications, and the process manages the shared resources within the middleware layer.

They furthermore claim that the following could be implemented as middleware: graphics managers, data converters, time services, file managers, databases, message managers, thread/process/job managers, and event managers. As a result, middleware has grown to encompass all OS resource management as well as network services. In essence, middleware is a "distributed, portable, and standard" implementation of these OS services.

Middleware and Distributed Systems — We would argue, however, that merely distributing an operating system does not make it middleware, nor does organizing a distributed system as a client/server architecture. This still allows middleware to be used to effectively coordinate the resources of the various nodes of the network, the network resources,

The ARPA/NSF workshop indicated that middleware is a major gap in current research.

One thing we can state clearly about middleware: it is something the OS and network designers think is part of the application, and something the application designers think is part of the OS or network.

possibly over heterogeneous platforms and subnetworks.

Note that we are using the term middleware to describe application — application coordination outside the OS. Most opinions do indicate that middleware is more than just part of an OS. The distributed nature of middleware, the inextensibility of existing OSs, and the desire to include and use middleware independent of OS upgrades, versions, and architectures, all likely result in this opinion.

It is not clear that middleware should not be part of an OS or network. It has aspects of being an interface to a virtual machine, and of including resource management. Thus, there are architecture dependent pieces of middleware ("lower middleware") that must understand and manipulate the resources of particular operating systems and networks.

Furthermore, it is clear that making middleware independent of the OS is useful ("upper middleware"). It permits use where OS sources are not available, or where users don't have sufficient control over their OS environment (i.e., root access in Unix). It also permits development of the middleware to evolve independently of a particular OS, and for implementations to be more easily ported across OSs. Note that the middleware API itself is not middleware.

One thing we can state clearly about middleware: it is something the OS and network designers think is part of the application, and something the application designers think is part of the OS or network. It is a layer created of services orphaned by two existing layers. To confuse matters, middleware is sometimes used as user-level extensions to the OS (i.e., user-level shared resource management and services), or as any set of tools built on other tools, or tools that support other tools.

There seems to be some agreement that middleware provides integration of applications. It is a "glue" that helps coordinate applications horizontally, within an OS as well as across multiple OSs.

What is Not Middleware? — It is worthwhile to note that although a strong definition for middleware may not yet exist, there are some candidates that we can argue strongly against. The basic argument is that anything that can be understood with an existing definition need not be renamed. Protocols are not in themselves middleware, they are protocols. They achieve many of the goals of middleware, providing a platform independent interface, and encouraging interoperability. Protocols built on protocols, such as reliable multicast, multicast itself, and hypertext protocols are themselves well understood as protocols. Additional services may be required to enable the effective use of these protocols, but that's another issue. Standard data formats are not middleware either; they too enable interoperation, but do not themselves integrate applications or systems. Applications that manage other applications are not middleware per se.

Languages are not middleware. They enable interoperation as much as standard data formats (they are standard data formats for the control flow). Alone, however, they do not coordinate or control, without the assistance of an OS-like layer. That layer is more appropriately called middleware, and languages are just languages.

Multimedia conference control systems (e.g., MMCC, SD) are control programs that have some of the right attributes, but are not yet widely deployed infrastructure that span and coordinate among various platforms.

A Proposed Definition and Applicability Test — We believe that middleware can be most completely and most generically described as: a service/API/protocol for OS-independent distributed shared resource management and integration.

We say *service/API/protocol* because to a programmer it is an interface, or an interface to a server, whereas to a networker it appears more like a protocol. In all cases, however, there is also a notion of persistent process and state associated with the resource management that spans peer relationships. This latter spanning is more than most protocols or API's would provide, and is more of an OS-like service.

One criteria for for middleware is a decomposition test: consider a set of applications A, a piece of middleware M, and an OS. If we can provide A with the services of M equally well by multiple instances of M that do not interact, as with a single (central or distributed) M, M is not middleware. In this case it's just a service, either of the OS, network or another application. It does not assist in a new way the coordination of applications.

Examples — At GBN '95, we created a list of candidate middleware components, which are annotated by comments generated from the discussion. Some of these were mentioned in other reports, notably the NRC report. They include:

- File systems — a standard version could be part of the OS; if this is middleware, it's to implement a standardization layer only.
- Security — some aspects are part of the OS or network, but others such as authentication services may apply.
- Addressing/routing assistance — probably part of the network layer.
- Network/user management — conventionally part of the link, network, and transport layer.
- Session management — sometimes referred to as call control in telephony, but used here as more than just a bundle of connections; application coordination across heterogenous OS architectures over the network.
- Information caching, e.g., Web pages or parts of video on demand streams — this combines OS and network stack functions, and so cannot be effectively implemented exclusively in either; this hints at what we believe may be an interesting feature of middleware.
- Billing/feedback — network resource charging is a network stack issue, but the price functions and management in conjunction with OS resources may be middleware; a general cost trade off mechanism cannot exist in any one exclusive domain.

Two additional examples of interesting potential middleware components were presented in GBN '95 sessions:

- MIT's user-transparent load balancer (load balancing requires network and OS cost function coordination).
- ISI's intelligent bandwidth (an explicit trade-off between network and OS resources, both within

a single host and among the members of a network).

Middleware itself is likely to remain an ill-defined and popular topic of debate at upcoming networking events. We hope this discussion will foster dialogue on how middleware can be something new, rather than a new name for old tricks.

Posters

There were six submissions accepted as posters: "Study of Interoperability between Various Rate-Based Flow Control: Mechanisms for Available Bit Rate Traffic in ATM Networks" by Yoon Chang, Nada Golmie, and David Su; "In-Service Monitoring of QOS in ATM Networks" by Thomas Chen, Steve Liu, and Vijay Samalam; "Credit-Based Flow Control for ATM Networks Revisited" by Joseph B. Evans, Luiz Dasilva, Hongbo Zhu, and Victor S. Frost; "Project xbind" by Aurel A. Lazar; "ATOMIC-2: Production Use of a Gigabit LAN" by Joe Touch, Hong Xu, Annette DeSchon, and Ted Faber; and "Models for Multipoint Connections in Gigabit Networks" by Bernard M. Waxman. These are all on display on the GBN '95 Web page.

Final Remarks

The individual contributions in this summary have been edited for length and consistency from the accepted abstracts, available in their original form on the GBN '95 Web page. Any errors in this process are solely that of the primary author of this paper. Some of the abstracts contained references which have been omitted here.

The discussion sections are based on contributions prepared for the workshop by Henning Schulzrinne and Joe Touch, and substantially reworked jointly by the authors of this summary, based in part on the discussion during the workshop.

The success of the workshop can be attributed to the contributers, participants, and program committee members: Nim Cheung (Bellcore), David Feldmeier (Bellcore), Bryan Lyles (Xerox PARC), Ira Richer (MITRE), Richard Skillen (Northern Telecom), James Sterbenz (GTE Laboratories), Richard Thompson (University of Pittsburgh), and Shukri Wakid (NIST). We would also like to thank David Feldmeier and Elizabeth Wilber for their support in the generation of this report and its quick publication in *IEEE Network*.

More information on the activities of the TCGN and future GBN workshops, as well as the on-line Proceedings of GBN '94 and GBN '95, are available on the TCGN Web page: `http://info.gte.com/ieee-tcgn`.

References

[1] I. Cidon *et al.*, "A Critique of ATM from a Data Communications Perspective," *Journal of High Speed Networks*, vol. 1, no. 4, 1992, pp. 315-336.

[2] J. P. G. Sterbenz, "Protocols for High Speed Networks: Life After ATM?" in Protocols for High Speed Networks IV, G. Neufeld and M. Ito, eds., (Chapman & Hall, London, 1994), pp. 3-18; also available as http://info.gte.com/jpgs/paper/pfhsnIV.html

[3] A. Romanow and S. Floyd, "The Dynamics of TCP Traffic over ATM Networks," Proc. of SIGCOMM '94, London, pp. 79-88.

[4] K. Moldeklev and P. Gunningberg, "Deadlock Situations in TCP over ATM," in Protocols for High Speed Networks IV, G. Neufeld and M. Ito, eds., (Chapman & Hall, London, 1994), pp. 243-259; revised version to appear in *IEEE/ACM Trans. on Networking*.

[5] CSTBUs NRENAISSANCE Committee, Realizing the Information Future, (National Academy Press, Wash. D.C., 1994); also available as http://xerxes.nas.edu/70/1/nap/online/rtif.

[6] C. Partidge, ed., Report of the ARPA/NSF Workshop on Research in Gigabit Networking, Wash. D.C., July 1994; available from ftp://ftp.std.com/pub/craigp/report.ps.

[7] P. Bernstein, Middleware: An Architecture for Distributed System Services, DEC CRL 9/36, Digital Equipment Corp., Cambridge, MA, 1993.

Biographies

JAMES P.G. STERBENZ [M '91] received bachelors degrees in electrical engineering, computer science, and economics from Washington University in St. Louis in 1980. He then worked for NCR corporation and IBM corporation until 1984. He received masters and doctorate degrees in computer science from Washington University, the latter in 1991 with dissertation work on a zero copy gigabit host-network interface supporting distributed virtual shared memory. He returned to IBM HPCC in Milford, Connecticut and Hawthorne, New York to work on network interface architecture and ATM support for fast packet switches until 1994. He is now a senior member of technical staff in the Broadband Intelligent Networks Project at GTE Laboratories, Waltham, Massachusetts. His research interests are broadband networking and the operating systems, computer architecture and VLSI design, and communication protocols required to support high performance applications and network services. He is a member of ACM, EFF, IEEE, IFIP, ISoc, Usenix, and vice-chair of the IEEE Communications Society Technical Committee on Gigabit Networking. He is program chair for the Gigabit Networking Workshops (GBN) and on the technical program committees for IEEE INFOCOM and HPCS, ACM SIGCOMM, and IFIP PfHSN. His e-mail address is: jpgs@ieee.org and URL is http://info.gte.com/jpgs.

HENNING G. SCHULZRINNE [M '92] received his undergraduate degree in economics and electrical engineering from the Technische Hochschule in Darmstadt, Germany, in 1984, his M.S.E.E. as a Fulbright scholar from the University of Cincinnati, Ohio and his Ph.D. from the University of Massachusetts at Amherst in 1987 and 1992, respectively. From 1992 to 1994, he was a member of technical staff at AT&T Bell Laboratories, Murray Hill. In 1994, he joined GMD FOKUS, Berlin, Germany, as a postdoctoral researcher. His research interests encompass real-time network services, the Internet and modeling and performance evaluation. His e-mail address is: hgs@fokus.gmd.de and URL is http://www.fokus.gmd.de/step/hgs.

JOSEPH D. TOUCH [M'92] received a B.S. (Hons.) degree in biophysics and computer science from the University of Scranton in 1985, an M.S. from Cornell University in 1988 and a Ph.D. from the University of Pennsylvania in 1992, both in computer science. He joined USC/Information Sciences Institute, Marina del Rey, California, in 1992, and is currently a project leader in the High Performance Computing and Communications Division there, directing the ATOMIC-2 and PC-ATOMIC tasks. He is also a research assistant professor in the Department of Computer Science, University of Southern California, where he teaches Advanced Operating Systems. Since 1988 he has been addressing issues of latency and source-anticipative protocols. In 1994, he received a U.S. patent for a device for latency reducing processor-memory interface. He is also interested in issues of telecommuting and on-line city services, and response-time reducing extensions to the World-Wide Web. He is a member of the program committees of IEEE INFOCOM '94 and '95, Protocols for High Speed Networks '94, and Physcomp '94. He is a member of Sigma Xi (S'84, M'93). His e-mail address is: touch@isi.edu and URL is http://www.isi.edu/~touch.

We believe middleware can be most completely and most generically described as: a service/API/protocol for OS-independent distributed shared resource management and integration.