Alternative Approaches to ATM/Internet Interoperation

Donald F. Box, Douglas C. Schmidt, and Tatsuya Suda
dbox@ics.uci.edu, schmidt@ics.uci.edu, and suda@ics.uci.edu
Department of Information and Computer Science,
University of California, Irvine,
Irvine, CA 92717, U.S.A.
(714) 856-4105 (phone)
(714) 856-4056 (fax) *

*This material is based upon work supported by the National Science Foundation under Grant No. NCR-8907909. This research is also in part supported by University of California MICRO program.

1 Introduction

The advent of long haul, high-speed Asynchronous Transfer Mode (ATM) networks introduces a "backward compatibility" dilemma for existing higher-layer communication protocols, *e.g.*, the TCP/IP suite operating in the Internet environment. This paper compares and contrasts four possible migration paths to ATM-Internet interoperation: (1) integrate ATM transparently below existing internetworking and transport protocol layers, (2) create new special purpose protocols designed specifically for ATM, (3) extend existing protocol requirements to add support for ATM, (4) design optional special-case enhancements to existing protocols.

It is difficult to architect a clean solution that is both efficient and backwardly compatible with existing systems. Protocols designed for the current Internet require general case assumptions, since it is difficult to control possible end-to-end path conditions. In the best case this adds redundancy, in the worst case it degrades performance.

Our research group at University of California, Irvine is currently developing an environment for analysis and experimentation with various alternatives for incorporating higher-layer internetworked communication protocols with ATM. In order to support development of protocol designs that precisely meet application requirements and underlying network characteristics, we are developing a system called ADAPTIVE, which stands for "A Dynamically Assembled Protocol Transformation, Integration, and Validation Environment." Using ADAPTIVE, we are able to match diverse multimedia applications to a wide range of network characteristics. This paper details one problem in high-speed networking our work seeks to address.

2 ATM Characteristics

The CCITT has chosen Asynchronous Transfer Mode (ATM) as the transfer mode for implementing Broadband ISDN (B-ISDN). ATM is a major departure from conventional data networks, such as the Internet and IEEE 802 LANs. It promises extremely low packet loss and bit error rates, sequenced data delivery, and a channel that experiences only transient congestion. In an ATM network, user frames are divided into small-sized cells (53 bytes), and cells are switched through an ATM network at high-speeds. Lower layer protocols within a network are simplified and their functionality minimized to only provide error checking of cell headers.

One distinguishing ATM network characteristic is its large bandwidth-delay product. Assuming 53 byte cells, a 155.52 Mbps ATM standard channel, and a propagation delay of 5 μsec per Km of a cable, a coast-to-coast optical fiber transmission line (around 5,000 Kms) can hold approximately 9,170 cells (or equivalently, 486 Kbytes). With the soon-to-be-standard channel speed of 622.08 Mbps, the ATM link can hold almost 37,000 cells (around 2 Mbytes) under the same conditions. In contrast, a coast-to-coast standard DS-1 transmission line (1.544 Mbps) holds only 4.825 Kbytes.

Another ATM network characteristic is its dynamism, resulting from the highly bursty multimedia traffic that ATM supports. Bursty network traffic can cause temporal network congestion, and during congestion periods many cells may be lost. To avoid this undesirable situation, cell-level congestion control is performed at the edge of an ATM network (*i.e.*, at user-network interface). Research shows that ATM networks employing proper congestion control achieve 10^{-9} cell loss rate, which is comparable to the underlying optical fiber's bit error rate.

3 Internet Characteristics

The TCP/IP protocol suite is designed to operate over the widely diverse networks that comprise the Internet, *e.g.*, Ethernet, FDDI, T1 links, etc. TCP/IP assumes a best-effort, unreliable datagram delivery service as its basic data transport mechanism. Its higher-layer protocols perform various functions that compensate for underlying network differences and limitations. For example, TCP performs error correction and detection based on sender time-out, positive acknowledgment, and retransmission. Likewise, to avoid flooding intermediate gateways, TCP handles end-to-end flow and congestion control based on a smoothed feedback-based estimate of the round trip delay [JB88].

TCP was originally designed for an internet that that possessed relatively low channel speeds, relatively high error rates, and a small bandwidth-delay product, due to the low latency of local area networks and the low bandwidth of existing long haul links. This small bandwidth-delay product permits feedback-based flow and congestion control schemes, such as the adaptive retransmission scheme mandated in the TCP requirements specification [JB88].

4 Interoperating ATM with TCP/IP

Feedback-based schemes can not react quickly enough to correct congestion in a high bandwidth-delay ATM network environment [BS91]. Moreover, highly transient congestion caused by bursty traffic sources further reduces the effectiveness of feedback-based schemes. In short, ATM networks possess several characteristics that are either redundant, or at odds, with TCP/IP mechanisms, *e.g.* ATM congestion control or proposed ATM-CS-layer frame retransmission.

If end-to-end communication service is provided only between the "edges" of an ATM network (or similar high-speed network), it is advantageous to replace or modify existing certain higher-layer protocol mechanisms with more appropriate (perhaps NULL) mechanisms that are more suitable for the underlying ATM network characteristics. To complicate matters, however, a potentially different set of functions must be performed if end-to-end service is provided over paths that also include lower speed LANs or MANs.

As listed in Section 1 above, there are several approaches for integrating ATM into the existing Internet environment, each with a varying degree of interoperability. This section outlines the advantages and drawbacks of each approach.

Solution 1: Integrate ATM Transparently Below Existing Protocols

One obvious way to incorporate ATM within TCP/IP is to simply provide an IP interface at gateways (*e.g.*, use the ATM adaptation layer facilities to fragment and reassemble IP frames into cells and vice versa). This approach is backwardly compatible with existing solutions and transparent to applications, since higher-layer existing protocols require no changes. Its primary disadvantage, however, is that end-to-end connections cannot fully utilize available ATM bandwidth due to limitations with existing feedback-based TCP flow/congestion control schemes. In addition, this solution pushes responsibility for managing ATM connections down to IP-ATM gateways, where end-to-end connections cannot be known without further blurring the TCP and IP layering boundary.

Solution 2: Create New Special-Purpose Protocols

A less transparent, but potentially more effective approach is to develop special-purpose higher-layer protocols that are tuned for particular ATM-characteristic/multimedia-application-requirement pairings [Top90, Pro90]. This approach supports performance tailoring by designing higher-layer protocols to take advantage of particular ATM services, such as reliability, high-speed, congestion control, etc. Furthermore, special-purpose protocols may minimize higher-layer duplication of ATM functionality, *e.g.*, transport- and session-layer connection management.

As discussed by Watson, however, there are hidden costs associated with special-purpose protocol designs [WM87]. For instance, achieving backwards compatibility becomes more problematic, since special-purpose protocols might not directly interoperate with existing protocols if changes are not transparent (*e.g.*, header formats, incompatible protocol behavior).

Solution 3: Add ATM Support to Existing Protocol Requirements

Another way to interoperate ATM and TCP/IP is to formally extend the basic assumptions that TCP makes about the underlying ATM network behavior. All conforming TCP implementations would then require modifications to explicitly support ATM characteristics. There is precedent for this approach, since enhancements to TCP have traditionally been mandated via modifications to requirements specifications, *e.g.*, TCP congestion control and adaptive retransmission schemes [Bra89].

In addition to improving existing performance, this approach ensures backwards compatibility, provided that the modifications/enhancements obey existing conventions. Compatibility is also achieved if mechanisms are provided to modify a protocol's local behavior, *e.g.*, disabling delayed transmission for interactive connections.

The primary disadvantage with solution 3 is that compromises made to encompass existing networks *and* ATM networks may reduce performance in both domains. Also, as with solution 1, specific ATM characteristics may not be available to end-to-end connections. For example, existing Internet gateways do not adequately support ATM-style services *e.g.*, constrained delay, resource reservation, packet prioritization and low packetization delay, etc. Finally, it is not clear that a compromise solution could deliver satisfactory performance and still be 100% backwardly compatible.

Solution 4: Special-Case Existing Protocols

More recent enhancements to TCP, such as big-window and timestamping [JB88, JBZ90], are special-case approaches. These enhancements enable two end-points to negotiate optional feature selection during connection establishment. The additional negotiated parameters are carried in the TCP header's options field. Implementations that support a particular option inform the source by setting the appropriate option field in a return header, whereas those that do not simply ignore the additional header fields.

Using this type of negotiation mechanism enables IP-ATM gateways to continue interoperating with existing IP gateways, while also adding services, such as resource reservation and admission control (with respect to a given ATM Virtual Path/Virtual Channel). Higher performance, end-to-end congestion and flow control schemes that would not work in the general Internet become possible if TCP can make certain assumptions about intermediate link characteristics. For instance, these assumptions could be made safely if IP exported an interface to TCP for services like resource reservation and fixed routing of datagrams.

There are several major advantages with solution 4. First, it interoperates with existing networks and hosts. Second, it is backwardly compatibility with existing protocol implementations. Third, it can achieve tailored performance when the environment is known and options are supported. Fourth, performance is no worse for existing environments when options are not supported.

However, there are also a number of disadvantages. For example, handling multiple options increases implementation complexity; designs and implementations which are sufficiently modular will require significantly less effort to adapt and will be able to support a greater variety of options than tightly coupled, monolithic designs. Moreover, certain functions may require hop-by-hop enhancements to intermediate IP gateways, which also limits applicability. Finally, options that are enabled/disabled only during connection establishment may not adapt to the highly dynamic fluctuations of underlying ATM networks. Likewise, options that are enabled/disabled at any time, based on feedback, may not adapt to underlying delivery system fluctuations quickly enough due to propagation delay.

5 Conclusions

Effectively interoperating ATM networks into the Internet requires potential modifications to existing protocol requirement specifications, protocol implementations and/or protocol architectures.

Solution 1, integrating ATM transparently below existing protocols, only requires new IP-ATM gateways, which are necessary regardless of the approach taken. Solution 2, adding new special-case protocols, requires developing protocols from scratch; to achieve backwards compatibility, additional network or transport layer gateways or convergence layers may also be necessary. Solution 3, adding ATM support to existing protocols, may permit selectively modifying existing implementations to reflect protocol changes. Solution 4, adding special-case options to existing protocols, only requires modifying existing implementations that wish use the protocol enhancements.

Our group at UC Irvine is developing ADAPTIVE, an environment to support development of protocol designs that precisely meet application requirements and underlying network characteristics. ADAPTIVE provides us with a highly modularized protocol development environment that we use to empirically explore the architectural and performance trade-offs of several solution alternatives described above.

References

[Bra89] R. Braden. Requirements for Internet Hosts. *Network Information Center RFC 1122*, pages 1–116, October 1989.

[BS91] Jaime Jungok Bae and Tatsuya Suda. Survey of Traffic Control Schemes and Protocols in ATM Networks. *Proceedings of the IEEE*, 79(2):170–189, February 1991.

[JB88] Van Jacobson and Robert Braden. TCP Extensions for Long-Delay Paths. *Network Information Center RFC 1072*, pages 1–16, October 1988.

[JBZ90] Van Jacobson, Robert Braden, and Lixia Zhang. TCP Extensions for High-Speed Paths. *Network Information Center RFC 1185*, pages 1–21, October 1990.

[Pro90] Protocol Engines Incorporated, Santa Barbara. *XTP Protocol Definition*, September 1990.

[Top90] C. Topolcic. Experiemntal Internet Stream Protocol, Version 2 (ST-II). *Network Information Center RFC 1190*, pages 1–148, October 1990.

[WM87] Richard W. Watson and Sandy A. Mamrak. Gaining Efficiency in Transport Services by Appropriate Design and Implementation Choices. *ACM Transactions on Computer Systems*, 5(2):97–120, May 1987.