# Layered Multiplexing Considered Harmful

David L Tennenhouse*
*Laboratory for Computer Science*
*Massachusetts Institute of Technology*

**Abstract**

Traditionally, computer communication networks have been optimized with respect to throughput, robustness and absolute delay, with little or no concern for the variation in delay (*jitter*) induced by the network. It is now desirable that high speed networks support a greater range of telecommunication services by providing a multi-service environment. If the transmission of jitter-sensitive traffic is not to be arbitrarily precluded, then new protocol architectures must take account of, and provide support for, the constraint of jitter.

The ATM approach to broadband networking is presently being pursued within the CCITT (and elsewhere) as the unifying mechanism for the support of service integration, rate adaption, and jitter control within the lower layers of the network architecture. This position paper is specifically concerned with the jitter arising from the design of the *middle* and *upper* layers that operate within the end systems and relays of multi-service networks (MSNs).

## 1 What is Layered Multiplexing?

Multiplexing can be viewed as a layer function that maps some number of upper layer *tributary* associations onto a single lower layer *aggregate* association. The principal effect of multiplexing is the *sharing* of lower layer resources by upper layer clients. Clearly some degree of *inter-system* multiplexing is justified when a common transmission medium is shared by a number of independent systems. This style of multiplexing is normally accommodated within the media-dependent components (MAC) of the Data Link layer. In most environments, there is a further requirement for *intra-system* multiplexing to support the shared use of a system's points of attachment to its transmission media. In traditional layered architectures, *intra-system* multiplexing is not restricted to a single layer. Instead, the functionality is supported in a layered fashion at many points within the *upper* and *middle* layers of the protocol stack.

*Title suggested by Dr David Clark

The principal advantage of layered architectures is that they provide for the *step-by-step enhancement of communication services* [3]. In theory, each service boundary between adjacent layers identifies a stage in the enhancement process. In order to minimize the duplication of functionality across layers, the network architect should *collect similar functions into the same layer.*[1] In the case of the multiplexing function, this principle has been largely ignored. For example, the OSI architecture presently provides for multiplexing within six of the seven layers of the protocol stack.[2]

It is claimed that the extensive duplication of multiplexing functionality across the middle and upper layers is harmful and should be avoided.

# 2 How is all this Multiplexing Harmful?

Each layer of statistical multiplexing increases the exposure of individual upper layer associations to performance *cross-talk* induced by parallel associations. Furthermore, multiplexing and demultiplexing are scheduling processes that substantially increase the complexity of layer protocols and their implementations.

From the perspective of an application layer association, statistical delays are incurred as each data unit is passed down through the layers of an originating system and again as units are demultiplexed within the layers of the recipient system. Within each layer, jitter is induced as a result of the statistical arrival processes of the tributary upper level associations. In the case of recipient systems the demultiplexing process may lead to bulk arrivals that introduce additional jitter components.

These statistical effects lead to Quality of Service (QOS) *cross-talk* in which the network performance experienced by a given application association is unduly affected by the traffic pattern of the parallel tributaries with which it is multiplexed. One approach to this problem is to assign QOS parameters to individual application associations and to prioritize the multiplexing process in accordance with the QOS specifications. However, application-specific QOS parameters cannot be applied to the data units of tributaries that are themselves multiplexed. Consequently, the lower layers of the multiplexing hierarchy must operate on aggregate data units whose QOS requirements are ill-defined. The problem is compounded at recipient systems where demultiplexing takes place: high priority application messages must be delayed until they are fully demultiplexed at each of the intermediate layers.

Cross-talk effects are particularly apparent when a misbehaving application floods a multiplexing or demultiplexing point. Since the traffic generated by the offending application is embedded within multiplexed data units, it cannot be identified and dealt with at the lower layer. This traffic exerts back-pressure within and below the multiplexing

[1] Principle P4 guiding OSI layer determination [3].

[2] The presentation layer is the sole exception. However, presentation address selectors have been incorporated into the naming and addressing scheme [4] on the grounds of *architectural consistency.*

layer and may inhibit or delay the traffic of other tributaries.

# 3 So What's the Alternative?

From the perspective of a given layer, the statistical multiplexing of messages corresponds to the *scheduling* of *events* generated within the vertically adjacent layers. With layered multiplexing, the individual scheduling decisions are related to multiplexed aggregates rather than individual application associations. If the QOS cross-talk observed at the application layer is to be constrained, then application level scheduling information must be visible at each multiplexing point. In the absence of this information, sensible scheduling decisions are precluded.

The obvious alternative is to limit intra-system multiplexing to a single lower layer immediately adjacent to the network point of attachment. Multiplexing within the intermediate layers is eliminated by enforcing a one-to-one binding between the associations of vertically adjacent layers. Logically, each application association has its own vertical stack of association-specific layer elements. Transmitted messages are only multiplexed (within the data link layer) at the bottom of each stack where upper layer associations are bound to shared link layer resources. Similarly, incoming messages are directly demultiplexed on an association-specific basis and are immediately assigned to the appropriate protocol stack.

Since the protocol stacks of independent associations operate in parallel, scheduling decisions are made on an association-specific basis and the overall exposure to QOS cross-talk is significantly reduced. The vertical integration of each application's protocol stack is particularly convenient when layer processing functions are performed within a thread-based software environment. The processing of each application data unit is performed by a single thread which traverses only two contexts: the multiplexed link context; and the relevant application context. With layered multiplexing, additional threads and contexts may be introduced at each layer.

One disadvantage of the scheme is that it increases the *degree* of multiplexing at the lower layer and the number of logical elements (protocol state machines) within each of the intermediate layers. In practise, the overhead associated with the multiplexing function is only slightly dependent on the number of upper layer tributaries. Similarly, the overhead associated with additional state machines is of little consequence as it represents an increase in the volume of stored state rather than the volume of traffic processed within each layer. The increased memory requirement is more than offset by the reduction in processing complexity that is achieved through the elimination of multiplexing within the intermediate layers.

# 4 If Multiplexing is Harmful, Why do we do it?

The *preservation of layer independence* is one oft-cited argument for the incorporation of multiplexing within every layer. Although layer independence is a somewhat emotional issue, the view taken here is that the principal architectural objective should be the layered enhancement of communication services. Accordingly, functions such as multiplexing should not be replicated at every layer. This approach does not otherwise preclude the independent operation and management of individual layers.

At the middle and upper layers, intra-system multiplexing is often used as a convenient mechanism for the realization of a variety of functions. Many of these functions constitute unnecessary or inappropriate applications of multiplexing. One reason for the pervasive spread of layered multiplexing is that its principal symptoms, the introduction of in-band complexity and jitter, have little impact on traditional computer communications traffic. With the integration of jitter-sensitive MSN traffic these symptoms have become noticeable.

Layered multiplexing is sometimes used to conserve supposedly valuable logical resources such as virtual circuits. The assertion that the overhead associated with such resources is sufficient to warrant sharing seems somewhat tenuous. Furthermore, assumptions of this type can prove to be self-fulfilling. In the OSI environment, the Common Application Service Elements (CASE) support the sharing of presentation/session connections. It can be argued that the multiplexing of session activities is a prime contributant to the implementation overhead associated with session and CASE protocols.

Another application of multiplexing is the support of protocol discrimination at recipient systems. For example, SAP addresses in the IEEE 802 data link layer are used to demultiplex incoming messages on the basis of their network layer protocol. In the ARPA protocol suite, protocol discrimination is again the demultiplexing axis of the IP Network layer. Only at the transport layer (UDP and TCP) is demultiplexing performed on an association-specific basis. The proposed alternative is that protocol selection be performed on an association-specific basis rather than on a message by message basis.

Finally, multiplexing is used to support the synchronization and/or coordination of different upper layer associations. For example, it is important to preserve the relative timing of the constituent voice, image and text components of an integrated presentation. This synchronization function may be realized through the multiplexing of the individual components onto a common session layer association. In practise, this technique provides only a limited subset of the desired functionality. The individual application layer components have different traffic characteristics and the multiplexing solution precludes the use of different lower layer technologies for the transmission of each component. There is clearly a need for further research in this area.

# 5 Practical Experience

The approach described in this paper has been adopted in the design of the ATM-based protocol suite used within Project Unison [6]. The multi-service data link protocol (MSDL) supports the direct transfer of ATM cells between peer data link clients. Cells are exchanged on an association-specific basis and each ATM cell contains a link level association identifier. Since there is a simple vertical binding between application and data link level associations, multiplexing is only performed at the media access (MAC) interface to the transmission service.

The vertical compression of multiplexing functionality does not preclude the relaying of network traffic. MAC-level relaying supports the transparent interconnection of transmission systems. At the network layer, end-to-end associations are supported through the concatenation of link level associations [5]. Although concatenation involves the translation of the association identifiers carried within individual cells, the binding between application and data link layer associations is maintained: each data link association carries messages arising from a single application association. Management functions, including association establishment, are performed on an out-of-band basis using application level management associations.

The Unison architecture is designed for use in MSN environments, where the in-band components of applications such as real-time voice and video services do not require complex transport and session functions. Since each application association is supported by its own protocol stack, it is easy to customize the middle and upper layer services to suit individual applications. For example, a telephony application can uses separate associations to support the transfer of in-band samples and out-of-band signalling information. The sample stream operates directly over the network layer service. In contrast, the RPC-based signalling scheme requires transport service data unit delineation, RPC session, and data structure presentation functions.

The initial Unison facility supported local transmission over the ATM-based Cambridge Fast Ring (CFR [2]) and long haul interconnection over 2 Mbps ISDN links [1]. MAC layer relays provide CFR/ISDN interconnection whilst local CFR bridging is supported by network or MAC layer relaying, depending on whether or not addressing domain boundaries are traversed. Work is now underway to extend the architecture to support enhanced network layer relaying and the use of non-ATM transmission substrates.

# References


[1] J.W. Burren. Flexible Aggregation of Bandwidth for Primary Rate ISDN. In *Sigcomm*, ACM, September 1989.

[2] A. Hopper and R.M. Needham. The Cambridge Fast Ring Networking System. *IEEE Transactions on Computers*, 37(10):1214–1223, October 1988.

[3] *Open Systems Interconnection - Basic Reference Model.* International Standards Organization - Information Processing. International Standard 7498-1.

[4] *OSI Reference Model - Part 3: Naming and Addressing.* International Standards Organization - Information Processing. International Standard 7498-3.

[5] D.R. McAuley. *Internetworking with ATM.* Working Paper UC039, Project Unison, April 1989.

[6] D.L. Tennenhouse and I.M. Leslie. A Testbed for Wide Area ATM Research. In *Sigcomm*, ACM, September 1989.