# ATM Internetworking

**Anthony Alles**
**ATM Product Line Manager**
**Cisco Systems, Inc.**
**email: aalles@cisco.com**

**May 1995**

An abridged version of this paper was presented at Engineering InterOp, Las Vegas, March 1995.



## 1.0 INTRODUCTION

It is clear that Asynchronous Transfer Mode (ATM) technology will play a central role in the evolution of current workgroup, campus and enterprise networks. ATM delivers important advantages over existing LAN and WAN technologies, including the promise of scalable bandwidths at unprecedented price and performance points and Quality of Service (QoS) guarantees, which facilitate new classes of applications such as multimedia.

These benefits, however, come at a price. Contrary to common misconceptions, ATM is a very complex technology, perhaps the most complex ever developed by the networking industry. While the structure of ATM cells and cell switching do facilitate the development of hardware intensive, high performance ATM switches, the deployment of ATM *networks* requires the overlay of a highly complex, software intensive, protocol infrastructure. This infrastructure is required to both allow individual ATM switches to be linked into a network, and for such networks to internetwork with the vast installed base of existing local and wide area networks.

This paper is a survey of this protocol infrastructure. It starts by discussing the unique features of ATM networks—such as its connection oriented nature, which contributes to the complexity of ATM protocols. The fact that ATM is connection oriented implies the need for ATM specific signaling protocols and addressing structures, as well as protocols to route ATM connection requests across the ATM network. These ATM protocols, in turn, influence the manner in which existing higher layer protocols can operate over ATM networks. The latter can be done in a number of different ways, each with its own advantages and characteristics, which will be discussed.

The remainder of this paper is organized as follows:

- Section 2.0 presents an overview of the architecture of ATM networks, ATM connection management and ATM connection types.
- Section 3.0 discusses ATM signaling protocols and addressing models.
- Section 4.0 describes ATM routing protocols.
- Section 5.0 then shifts attention to the internetworking of ATM with existing LAN protocols, and, specifically, to the LAN emulation protocol.
- Section 6.0 discusses ATM native mode protocols, an alternate method for carrying higher layer protocols across ATM.
- Section 7.0 discusses some of the latest work of the ATM Forum on multiprotocol transport over ATM.
- Section 8.0 discusses wide area network (WAN) internetworking.
- Section 9.0 concludes the paper.
- Section 10.0 References.
- Appendix A presents a brief overview of ATM traffic management, since some of this material, which impacts ATM internetworking, is fairly recent, and may not be covered elsewhere.
- Appendix B summarizes the status of a number of the key completed and pending ATM specifications from the ATM Forum and the Internet Engineering Task Force (IETF).

This paper assumes familiarity with the fundamentals of ATM technology, including the ATM layer protocols and cell formats, and the operation of ATM switching systems. Many sources are available which describe these aspects of ATM systems—[McDysan], [Minoli], and [Prycker] are good sources for such background information.

Many of the protocols described in this paper were still under development, as of the time of writing, and aspects of their operation may change by the time the protocols are finalized. Consult the latest versions of the referenced specifications for the most current information.

## 2.0 ATM Network Operation

An ATM network consists of a set of ATM switches interconnected by point-to-point ATM links or interfaces. ATM switches support two kinds of interfaces: user-network interfaces (UNI) and network-node interfaces[1] (NNI). UNI connect ATM end-systems (hosts, routers, and so on) to an ATM switch[2], while an NNI may be imprecisely defined as an interface connecting two ATM switches together; slightly different cell formats are defined across UNI and NNI[3]. More precisely, however, an NNI is any physical or logical link across which two ATM switches exchange the NNI protocol[4]. This will be described in greater detail in Section 4.0.

[1] Sometimes also known as network-network interfaces; the difference is subtle and unimportant.

[2] ATM does not have an analog of the redundant physical links provided by FDDI, with its dual attached stations. Hence any end-system requiring a redundant connection to an ATM network will need to support two separate UNIs, and either operate one link in a standby mode, or perform local connection level load sharing between the links.

[3] In NNI cells, unlike UNI cells, there is no Generic Flow Control (GFC) field, and the first four bits of the cell are used by an expanded (12 bit) VPI field. Since the GFC is rarely used, however (its use is not defined, for instance, in the ATM Forum UNI specifications), there is, in practice, no functional difference between UNI and NNI cells, other than in the fact that the latter can support a larger VPI space.

[4] For this reason, the connection between a private ATM switch and a public ATM switch is a UNI—known as a Public UNI—since these switches do not typically exchange NNI information (refer to Section 4.5).

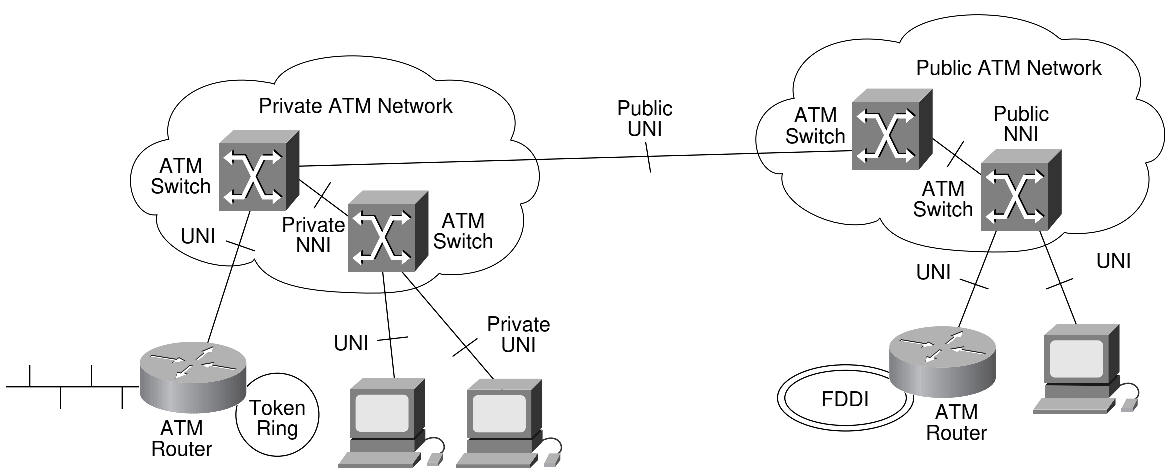



**Figure 1. ATM Network Interfaces**

As noted above, ATM networks are fundamentally connection oriented. This means that a virtual circuit needs to be set up across the ATM network prior to any data transfer. ATM circuits are of two types: virtual paths, identified by virtual path identifiers (VPI); and virtual channels, identified by the combination of a VPI and a virtual channel identifier (VCI). A virtual path is a bundle of virtual channels, all of which are switched transparently across the ATM network on the basis of the common VPI. All VCI and VPI, however, have only local significance across a particular link, and are remapped, as appropriate, at each switch. In normal operation, switches allocate all UNI connections within VPI=0; the use of other virtual paths is discussed later in this paper.

The basic operation of an ATM switch is very simple: to receive a cell across a link on a known VCI or VPI value; to look up the connection value in a local translation table to determine the outgoing port (or ports) of the connection and the new VPI/VCI value of the connection on that link; and to then retransmit the cell on that outgoing link with the appropriate connection identifiers.

The switch operation is so simple because external mechanisms set up the local translation tables prior to the transmittal of any data. The manner in which these tables are set up determine the two fundamental types of ATM connections:

- *Permanent Virtual Connections (PVC):* A PVC is a connection set up by some external mechanism, typically network management, in which a set of switches between an ATM source and destination ATM system are programmed with the appropriate VPI/VCI values. As is discussed later, ATM signaling can facilitate the set up of PVCs, but, by definition, PVCs always require some manual configuration. As such, their use can often be cumbersome.
- *Switched Virtual Connections (SVC):* An SVC is a connection that is set up automatically through a signaling protocol. SVCs do not require the manual interaction needed to set up PVCs and, as such, are likely to be much more widely used. All higher layer protocols operating over ATM primarily use SVCs, and it is these that are primarily considered in this paper.

| Input Port | Input VPI/VCI | Output Port | Output VPI/VCI |
|---|---|---|---|
| 1 | 29 | 2 | 45 |
| 2 | 45 | 1 | 29 |
| 1 | 64 | 3 | 29 |
| 3 | 29 | 1 | 64 |

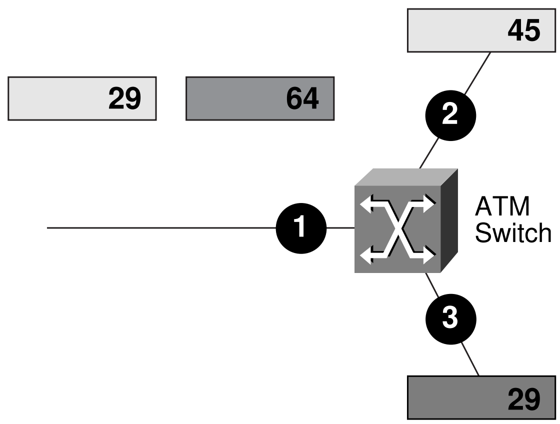



**Figure 2. ATM Switch Operation**

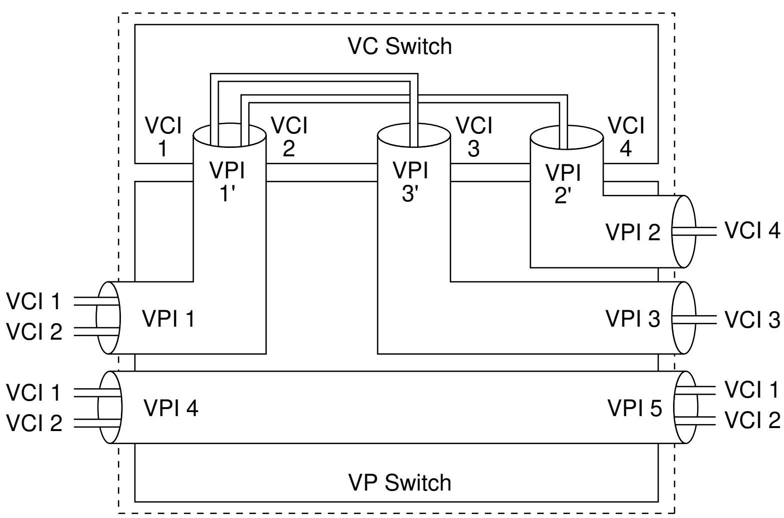



**Figure 3. Virtual Circuit and Virtual Path Switching**

ATM signaling is initiated by an ATM end-system that desires to set up a connection through an ATM network; signaling packets are sent on a well known[5] virtual channel, VPI=0, VCI=5. The signaling is routed through the network, from switch to switch[6], setting up the connection identifiers as it goes[7], until it reaches the destination end system. The latter can either accept and confirm the connection request, or can reject it, clearing the connection. Note that because the connection is set up along the path of the connection request, the data also flows along this same path.

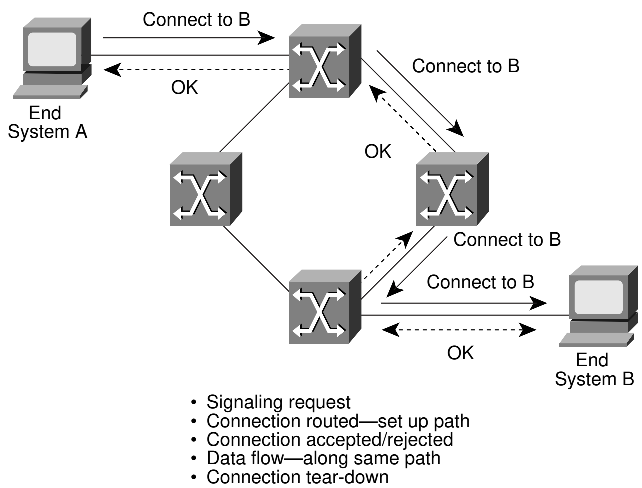



**Figure 4. Connection Setup through ATM Signaling (SVC)**

In the following section we discuss the ATM signaling protocols, while Section 4.0 discusses the ATM routing protocols that actually route ATM connection requests across the ATM network. Before this, the different types of ATM connection that can be set up, either as SVCs or PVCs are discussed.

[5] This means that this virtual channel is reserved for signaling traffic, and no other types of information may be transmitted across the connection. All switches are also preconfigured to receive any signaling packets sent across this connection and pass them to a signaling process associated with the switch. Other well known virtual channels, discussed throughout the paper (for the ILMI protocol, P-NNI protocols etc.) are treated in an equivalent manner. In general, all VCI below 32 are reserved within each VPI for such control purposes; data connections are hence allocated VCI outside this range.

[6] Strictly, the signaling requests are passed between the signaling or call control processes associated with the switches, and it is these that set-up the connection through the switches. In general, however, for the sake of robustness and performance, most vendors will integrate the call control capability into each switch, rather than supporting them on an off-board processor.

[7] The connection identifiers (that is, VPI/VCI values) for a particular connection are typically allocated, across any given link, by the node to which the request is sent, as opposed to the requesting node. Connection identifiers—with typically the same VPI/VCI values—are always allocated in each direction of a connection, but the traffic parameters in each direction can be different; in particular, the bandwidth in one direction could be zero.

There are two fundamental types of ATM connections:

- *Point-to-point connections*, which connect two ATM end-systems. Such connections can be unidirectional or bidirectional.
- *Point-to-multipoint connections*, which connects a single source end-system (known as the root node) to multiple destination end-systems (known as leaves). Cell replication is done within the network by the ATM switches[8] at which the connection splits into two or more branches. Such connections are unidirectional, permitting the root to transmit to the leaves, but not the leaves to transmit to the root, or to each other, on the same connection. The reason why such connections are only unidirectional are described below.

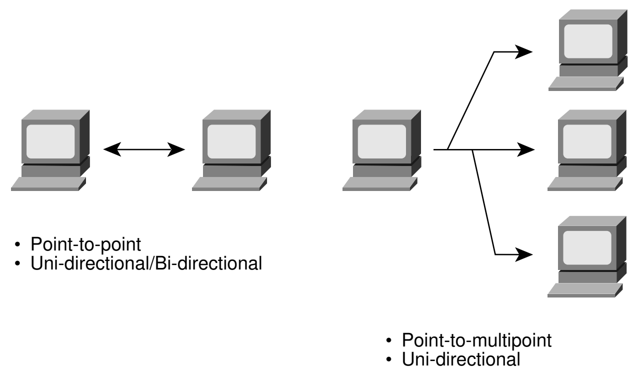



**Figure 5. Types of ATM Connections**

What is notably missing from these types of ATM connections is an analog to the multicasting or broadcasting[9] capability common in many shared medium LAN technologies such as Ethernet or Token Ring. In such technologies, multicasting allows multiple end systems to both receive data from other multiple systems, and to transmit data to these multiple systems. Such capabilities are easy to implement in shared media technologies such as LANs, where all nodes on a single LAN segment must necessarily process all packets sent on that segment. The obvious analog in ATM to a multicast LAN group would be a (bidirectional) multipoint-to-multipoint connection. Unfortunately, this obvious solution cannot be implemented when using AAL5, the most common ATM Adaptation Layer (AAL) used to transmit data across ATM networks.

[8] End systems could also replicate cells and send them to multiple end systems across multiple point-to-point links, but generally, ATM switches can perform replication much more efficiently than end systems.

[9] Broadcasting, where a single system transmits to all other systems, can be viewed as a special case of multicasting, and is so treated in this paper.

Unlike AAL 3/4[10], with its Message Identifier (MID) field (see [Forum1]), AAL 5 does not have any provision within its cell format for the interleaving of cells from different AAL5 packets on a single connection. This means that all AAL5 packets sent to a particular destination across a particular connection must be received in sequence, with no interleaving between the cells of different packets on the same connection, or the destination reassembly process would not be able to reconstruct the packets.

This is why ATM AAL 5 point-to-multipoint connections can only be unidirectional, for if a leaf node was to transmit an AAL 5 packet onto the connection, it would be received by both the root node and all other leaf nodes. However, at these nodes, the packet sent by the leaf could well be interleaved with packets sent by the root, and possibly other leaf nodes; this would preclude the reassembly of any of the interleaved packets. Clearly, this is not acceptable.

Notwithstanding this problem, ATM does require some form of multicast capability, since most existing protocols, being developed initially for LAN technologies, rely upon the existence of a low-level multicast/broadcast facility. Three methods have been proposed for solving this problem:

- **VP-Multicasting:** In this mechanism, a multipoint-to-multipoint VP links all nodes in the multicast group, and each node is given a unique VCI value within the VP. Interleaved packets can hence be identified by the unique VCI value of the source. Unfortunately, this mechanism requires a protocol to uniquely allocate VCI values to nodes; such a mechanism does not currently exist. It is also not clear whether current segmentation and reassembly (SAR) devices could easily support such a mode of operation[11].
- **Multicast Server:** In this mechanism, all nodes wishing to transmit onto a multicast group set up a point-to-point connection with an external device known as a multicast server (perhaps better described as a resequencer or serializer). The multicast server, in turn, is connected to all nodes wishing to receive the multicast packets through a point-to-multipoint connection[12]. The multicast server receives packets across the point-to-point connections, then retransmits them across the point-to-multipoint connection—but only after ensuring that the packets are serialized (that is, one packet is fully transmitted prior to the next being sent). In this way, cell interleaving is precluded.

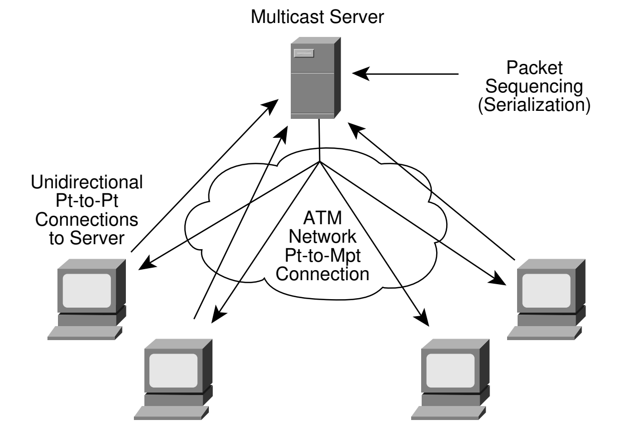



**Figure 6. Multicast Server Operation**

- **Overlaid Point-to-Multipoint Connections:** In this mechanism, all nodes in the multicast group establish a point-to-multipoint connection with each other node in the group, and, in turn, becomes a leaf in the equivalent connections of all other nodes. Hence, all nodes can both transmit to and receive from all other nodes.

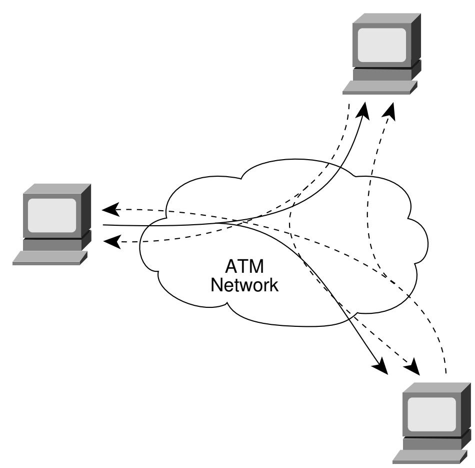



**Figure 7. Multicast Through Overlaid Point-to-Multipoint Connections**

[10] Despite the problems that AAL 5 has with multicast support, it is not really feasible to use AAL 3/4 for data transport instead. This is because AAL 3/4 is a much more complex protocol than AAL 5 and would lead to much more complex and expensive implementations; indeed, AAL 5 was developed specifically to replace AAL 3/4. In any case, while the MID field of AAL 3/4 could preclude cell interleaving problems, allowing for bidirectional, multipoint-to-multipoint connections, this would also require some mechanism for ensuring that all nodes in the connection use a unique MID value. There is no such mechanism currently in existence or development; the number of possible nodes within a given multicast group would also be severely limited due to the small size of the MID space.

[11] Furthermore, there is no support for switched virtual paths in the existing (UNI 3.0/3.1) signaling specifications. This capability will be added to the signaling protocols (UNI 4.0) currently under development.

[12] The multicast server could also connect to each of the destinations using point-to-point connections, and replicate the packets before transmission. In general, however, ATM networks can perform replication, through point-to-multipoint connections, much more efficiently.

The last mechanism requires each node to maintain *N* connections for each group, where *N* is the total number of transmitting nodes within the group, while the multicast server mechanism requires only two connections. This mechanism also requires a registration process for telling nodes that join a group what the other nodes in the group are, so that it can form its own point-to-multipoint connection. The other nodes (see below) also need to know about the new node so they can add the new node to their own point-to-multipoint connections. The multicast server mechanism is more scalable in terms of connection resources, but has the problem of requiring a centralized resequencer, which is both a potential bottleneck and a single point of failure.

In short, there is, as yet, no ideal solution within ATM for multicast. Higher layer protocols within ATM networks use both the latter two solutions for multicast, as will be discussed later in this paper. This is one example of why internetworking existing protocols with ATM is so complex. Most current protocols, particularly those developed for LANs, implicitly assume a network infrastructure very similar to existing LAN technologies—that is, a shared medium, connectionless technology with implicit broadcast mechanisms. As noted above, ATM violates all of these assumptions. In later sections the mechanisms used to work around these problems will be discussed.

Before proceeding, this brief survey of ATM networking will conclude with a mention of the Interim Local Management Interface (ILMI) protocol. The ILMI protocol uses SNMP format packets across the UNI (and also across NNI links, as discussed later) to access an ILMI Management Information Base (MIB) associated with the link, within each node. The ILMI protocol is run across a well known virtual channel, VPI=0, VCI=16. The ILMI protocol allows adjacent nodes to determine various characteristics of the other node—for example, the size of each other's connection space, the type of signaling used, hooks for network management autodiscovery, and so on. One of its most useful features, address registration, greatly facilitates the administration of ATM addresses and is discussed in the next section. The ILMI will likely be extended in the future to support other autoconfiguration capabilities, such as for group addressing, as discussed later.

## 3.0 ATM Signaling and Addressing

The current and planned ATM signaling protocols and their associated ATM addressing models are discussed in this section. ATM signaling protocols vary by the type of ATM link—ATM UNI signaling is used between an ATM end-system and an ATM switch across an ATM UNI; ATM NNI signaling is used across NNI links. As of the time of this writing, standards exist only for ATM UNI signaling, although work is continuing on NNI signaling. The current standard for ATM UNI signaling is described in the ATM Forum UNI 3.1 specification [Forum1], which is a slight modification to the earlier UNI 3.0 specification[13] [Forum2]. UNI signaling requests are carried across the UNI in a well known default connection: VPI=0, VCI=5.

The UNI 3.1 specification is based upon Q.2931, a public network signaling protocol developed by the International Telecommunications Union-Telecommunications Sector[14] (ITU-T), which, in turn, was based upon the Q.931 signaling protocol used with Narrowband ISDN (N-ISDN). The ATM signaling protocols run on top of a Service Specific Convergence Protocol (SSCOP), defined by the ITU-T Recommendations Q.2100, Q.2110, and Q.2130. This is a data link protocol that guarantees delivery through the use of windows and retransmissions[15].

ATM signaling uses the 'one-pass' method of connection set-up, which is the model used in all common telecommunications networks (e.g. the telephone network). That is, a connection request from the source end-system is propagated through the network, setting up the connection as it goes, until it reaches the final destination end-system. The routing of the connection request - and hence of any subsequent data flow - is governed by the ATM routing protocols (e.g. the P-NNI protocols discussed in the following section). Such protocols route the connection request based upon both the destination address, and the traffic and QoS parameters requested by the source end-system. The destination end-system may choose to accept or reject the connection request, but since the call routing is based purely on the parameters in the initial connection request message, the scope for negotiation of connection parameters between source and destination - which may, in turn, affect the connection routing - is limited.

[13] Apart from some minor "bug-fixes," the only substantive difference between UNI 3.0 and UNI 3.1 is in the data link protocol, SSCOP, used for the reliable transport of the ATM signaling packets. UNI 3.1 brought the ATM Forum signaling specification into alignment with the ITU-T's Q.2931 signaling protocol stack; UNI 3.0 had referenced an earlier draft, Q.93b. There are no functional differences between UNI 3.0 and UNI 3.1, but unfortunately, the two are not interoperable due to the differences in the data link protocol—UNI 3.0 referenced an earlier, non-interoperable draft of Q.2100, known as Q.SAAL.

[14] Known formerly as the CCITT.

[15] Note that in general, ATM does not offer an assured service—cells are not retransmitted by ATM devices upon loss, for instance, since it is assumed that higher layers (such as TCP) will handle reliable delivery, if this is what the application requires. This also makes ATM devices much simpler, faster, and cheaper. Refer to [Partridge3] for a discussion of reliable delivery in ATM networks. ATM signaling requires the assured delivery guarantees of SSCOP since it does not run on any standard higher layer protocol like TCP, and the signaling state machines can be made much simpler if assured delivery can be assumed.

A number of message types are defined in the UNI 3.0/3.1 specification, together with a number of state machines defining the operation of the protocol, cause error codes defining reasons for connection failure, and so forth. Data elements used in the signaling protocol - addresses, for instance - are carried within Information Elements (IE) within the signaling packets.

In overview, a source end-system wishing to set up a connection will formulate and send into the network, across its UNI, a *Setup* message, containing the destination end-system address, desired traffic and QoS parameters, various IEs defining particular desired higher layer protocol bindings (see Section 6.2.1) and so forth. This Setup message is sent to the first, ingress switch, across the UNI, which responds with a local *Call Proceeding* acknowledgment. The ingress switch will then invoke an ATM routing protocol, as discussed in the following section, to propagate the signaling request across the network, to the egress switch to which is attached the destination end-system.

This egress switch will then forward the Setup message to the end-system, across its UNI. The latter may choose to either accept or reject the connection request; in the former case, it returns a *Connect* message, back through the network, along the same path, to the requesting source end-system. Once the source end-system receives and acknowledges the Connect message, either node can then start transmitting data on the connection. If the destination end-system rejects the connection request, it returns a *Release* message, which is also sent back to the source end-system, clearing the connection (e.g. any allocated connection identifiers) as it proceeds. Release message are also used by either of the end-systems, or by the network, to clear an established connection.

The ATM Forum greatly simplified the Q.2931 protocol, but also extended it to add support for point-to-multipoint connection set up. In particular, UNI 3.1 allows for a root node to set up a point-to-multipoint connection, and to subsequently add a leaf node. While a leaf node can autonomously leave such a connection, it cannot add itself.

The ATM Forum is currently working on new signaling capabilities, which will be released in the second half of 1995 as part of its UNI 4.0 specification [Forum3]. UNI 4.0 will add support for, amongst other things, leaf-initiated joins to a multipoint connection. While some would like to use this to allow for true multipoint-to-multipoint connections, it should be noted that signaling support for such connections does not imply the existence of a suitable mechanism for such connections. At the time of this writing, it is not clear that UNI 4.0 will have any better solution for multicast within ATM than what exists today.

The most important contribution of UNI 3.0/3.1 in terms of internetworking across ATM was its addressing structure. Any signaling protocol, of course, requires an addressing scheme to allow the signaling protocol to identify the sources and destination of connections. The ITU-T has long settled upon the use of telephone number-like E.164 addresses as the addressing structure for public ATM (B-ISDN) networks. Since E.164 addresses are a public (and expensive) resource, and cannot typically be used within private networks, the ATM Forum extended ATM addressing to include private networks. In developing such a private network addressing scheme for UNI 3.0/3.1, the ATM Forum evaluated two fundamentally different models for addressing.

These two models differed in the way in which the ATM protocol layer was viewed in relation to existing protocol layers, in particular, existing network layer protocols such as IP, IPX, and so on. These existing protocols all have their own addressing schemes and associated routing protocols. One proposal was to also use these same addressing schemes within ATM networks. Hence ATM endpoints would be identified by existing network layer addresses (such as IP addresses), and ATM signaling requests would carry such addresses. Existing network layer routing protocols (such as IGRP and OSPF [Dickie]) would also be used within the ATM network to route the ATM signaling requests, since these requests, using existing network layer addresses, would look essentially look like connectionless packets.

This model was known as the *peer* model, since it essentially treats the ATM layer as a peer of existing network layers.

An alternate model sought to decouple the ATM layer from any existing protocol, defining for it an entirely new addressing structure. By implication, all existing protocols would operate over the ATM network. For this reason, the model is known as the *subnetwork* or *overlay* model. This mode of operation is, in fact, the manner in which such protocols as IP operate over such protocols like X.25 or over dial-up lines. The overlay model requires the definition of both a new addressing structure, and an associated routing protocol. All ATM systems would need to be assigned an ATM address in addition to any higher layer protocol addresses it would also support. The ATM addressing space would be logically disjoint from the addressing space of whatever protocol would run over the ATM layer, and typically would not bear any relationship with it. Hence, all protocols operating over an ATM subnet would also require some form of ATM address resolution protocol to map higher layer addresses (such as IP addresses) to their corresponding ATM addresses.

Note that the peer model does not require such address resolution protocols. By using existing routing protocols, the peer model also may have precluded the need for the development of a new ATM routing protocol.

Nonetheless, it was the overlay model that was finally chosen by the ATM Forum for use with UNI 3.0/3.1 signaling. Among other reasons, the peer model, while simplifying end-system address administration, greatly increases the complexity of ATM switches, since they must essentially act like multiprotocol routers and support address tables for all current protocols, as well as all of their existing routing protocols. Current routing protocols, being originally developed for current LAN and WAN networks, also do not map well into ATM or allow use of ATM's unique QoS properties.

Perhaps most importantly, the overlay model, by decoupling ATM from other higher protocol layers, allows each to be developed independently of the other. This is very important from a practical engineering viewpoint—as will be seen, both ATM and evolving higher layer protocols are extremely complex and coupling their development would likely have slowed the deployment of ATM quite considerably. Though there is a price to pay for such layering, in the need for disjoint address spaces and routing protocols, and in possibly suboptimal end-to-end routing[16], the practical benefits arguably greatly exceed the theoretical costs.

Given the choice of the overlay model, the ATM Forum then defined an address format for private networks based on the *syntax* of an OSI Network Service Access Point (NSAP) address. Note, however, that an ATM address is *not* an NSAP, despite the similar structure; while in common usage such addresses are often referred to as "NSAP addresses," they are better described as ATM private network addresses, or ATM end-point identifiers, and identify not NSAPs, but subnetwork points of attachment.

The 20-byte NSAP format ATM addresses are designed for use within private ATM networks, while public networks typically use E.164 addresses that are formatted as defined by ITU-T. The Forum did specify, however, an NSAP encoding for E.164 addresses. This will be used for encoding E.164 addresses within private networks but may also be used by some private networks. Such networks may base their own (NSAP format) addressing on the E.164 address of the public UNI to which they are connected and take the address prefix from the E.164 number, identifying local nodes by the lower order bits.

[16] This may happen in large, meshed networks consisting of both packet routers and ATM switches because the higher layer packet routing protocols operate independently of the ATM level routing protocol [Cole]. Hence once a path is chosen, crossing the ATM network, a change in the topology or characteristics of the ATM layer would not become known to the higher layer routing protocol, even if that change would result in a different, more optimal end-to-end path, bypassing the ATM network, being chosen. While this is indeed a potential drawback of the overlay model, in practice it is unlikely to be a major problem since it is likely that in any practical network the ATM network would always remain the preferred path.

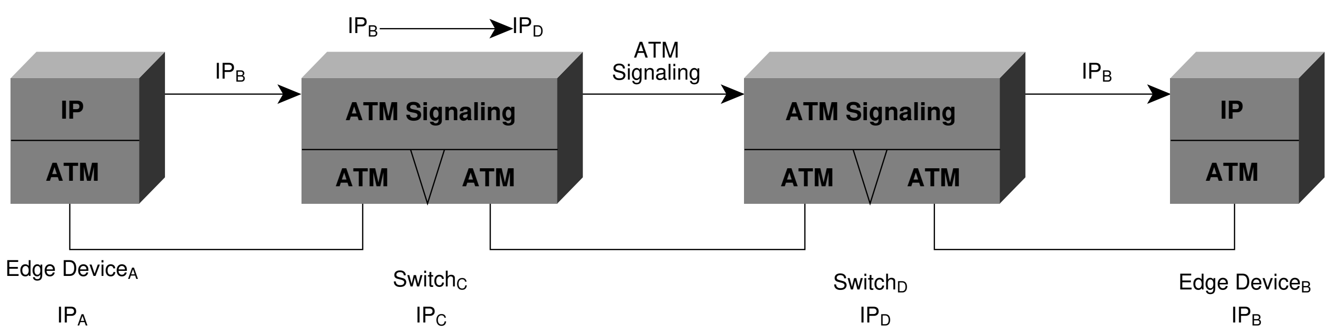



**Figure 8. Peer Model of ATM Addressing**

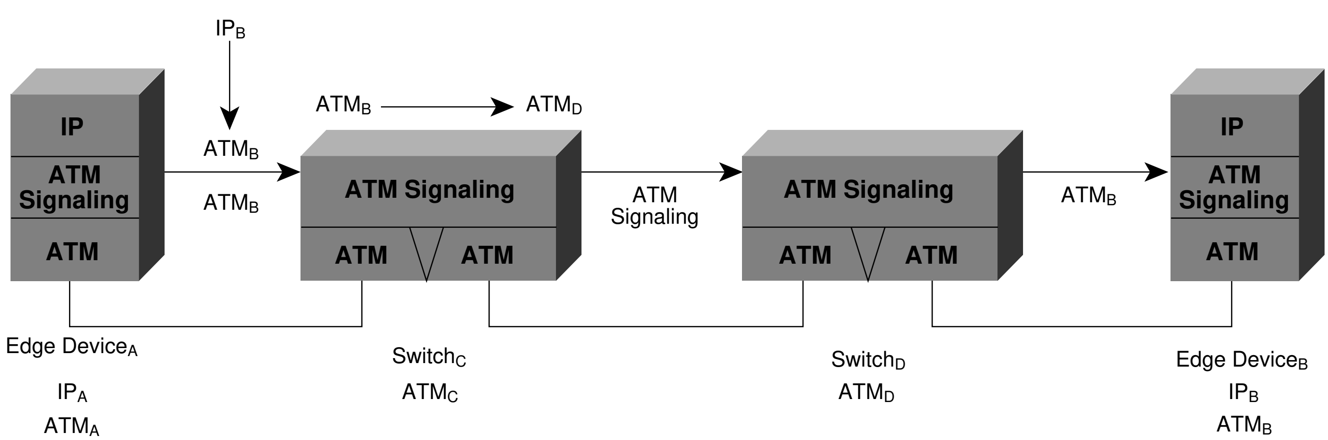



**Figure 9. Overlay Model of ATM Addressing**

All NSAP format ATM addresses consist of three components: an Authority and Format Identifier (AFI), which identifies the type and format of the Initial Domain Identifier (IDI); the IDI, which identifies the address allocation and administration authority; and the Domain Specific Part (DSP), which contains actual routing information. The Q.2931 protocol defines source and destination address fields for signaling requests, and also defines subaddress fields for each; the use of the latter are explored later in this paper.

There are three formats of private ATM addressing that differ by the nature of the AFI and IDI:

- *NSAP Encoded E.164 format:* In this case, the IDI is an E.164 number.
- *DCC Format:* In this case, the IDI is a Data Country Code (DCC); these identify particular countries, as specified in ISO 3166. Such addresses are administered by the ISO National Member Body in each country.
- *ICD Format:* In this case, the IDI is an International Code Designator (ICD); these are allocated by the ISO 6523 registration authority (the British Standards Institute). ICD codes identify particular international organizations.

The ATM Forum recommends that organizations or private network service providers use either the DCC or ICD formats to form their own numbering plan. Organizations that want to obtain ATM addresses would do so through the same mechanism used to obtain NSAP addresses (for example, through a local address administration body—in the US, this is ANSI). Once obtained, such addresses can be used for both ATM addresses and also, if desired, for NSAP addressing[17].

In real NSAPs, the DSP is typically subdivided into a fixed hierarchy that consists of a Routing Domain (RD), an Area identifier (AREA), and an End System Identifier (ESI). The ATM Forum, however, has combined the RD and AREA fields into a single High-Order DSP (HO-DSP) field, which is then used to support flexible, multi-level addressing hierarchies for prefix-based routing protocols. No rigid boundary exists within the HO-DSP; instead, a range of addressing hierarchies will be supported, using prefix masks, as with IP subnets. This is described in more detail in Section 4.0.

[17] If CLNP is run over ATM, the same value might well be used to identify a node's NSAP address and its ATM address.

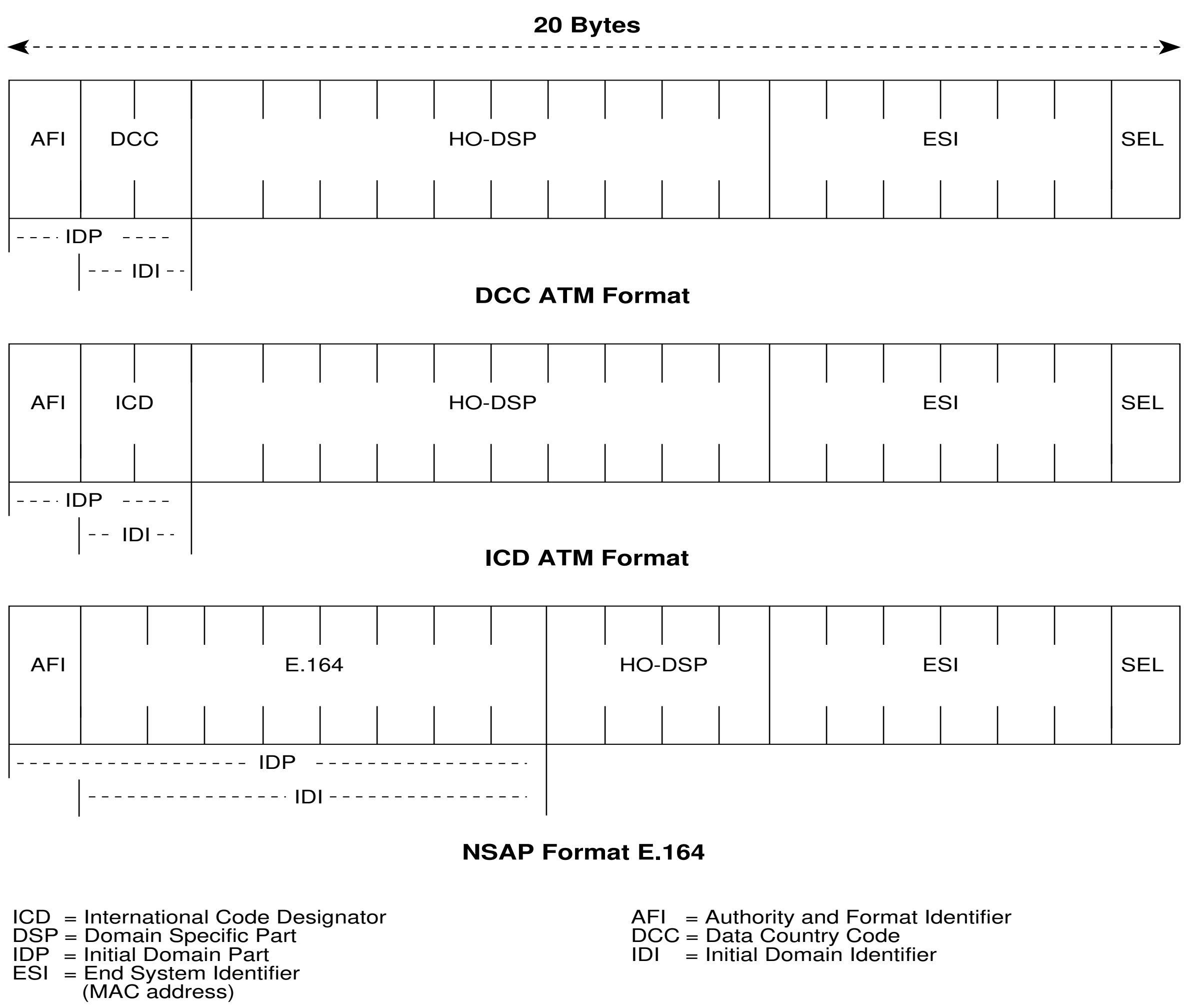



**Figure 10. ATM Private Network Address Formats**

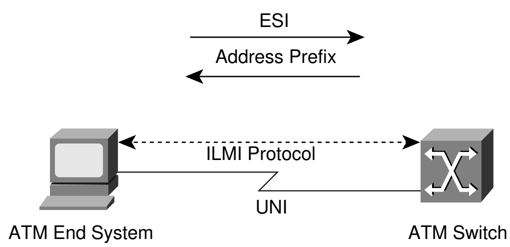



**Figure 11. Address Registration Using the ILMI Protocol**

The ESI field is specified to be a 48-bit MAC address, as administered by the IEEE. This facilitates the support of both LAN equipment, which is typically hardwired with such addresses, and of such LAN protocols as IPX, which rely on MAC addresses. The final, one octet, Selector (SEL) field is meant to be used for local multiplexing within end-stations and has no network significance.

To facilitate the administration and configuration of ATM addresses into ATM end systems across UNI, the ATM Forum defined an address registration mechanism using the ILMI. This allows an ATM end-system to inform an ATM switch across the UNI, of its unique MAC address, and to receive the remainder of the node's full ATM address in return. This mechanism not only facilitates the autoconfiguration of a node's ATM addressing, but may also be extended, in the future, to allow for the autoconfiguration of other types of information (such as higher layer addresses and server addresses).

Note that the addressing formats defined in UNI 3.0/3.1 identify only single end-points. These can also be used to set up point-to-multipoint connections because in UNI 3.0/3.1 such connections are set up a leaf at a time, using unicast addressing. UNI 4.0 will add support for group addresses, and will permit point-to-multipoint connections to be set up to multiple leaves in one request.

The notion of an *anycast* address will also be supported in UNI 4.0. An well known anycast address, which may be shared by multiple end systems, is used to used to route a request to a node providing a particular service [Partridge1], and not to identify the particular node per se. A call made to an anycast address is routed to the "nearest" end-system that registered itself with the network to provide the associated service. Anycast is a powerful mechanism for autoconfiguration and operation of networks since it precludes the need for manual configuration or service locations protocols. While few details of ATM group addressing have yet been determined, the ATM Forum has decided that anycast will be addressed as a special case of group addressing.

Specifically, nodes will use an extension of the ILMI address registration mechanism to inform the network that they support a particular group address (note that this is the opposite of the normal address registration mechanism). As part of this registration, the node also informs the network of the desired scope of registration, that is, the extent of the network to which the existence of the multicast node should be advertised (as part of the ATM routing protocols—see below). This scope is administrative (such as within a single building, within the local site, or within the enterprise network). The network must map this information through administrative policy to the ATM routing protocol's own hierarchy. Once a node has registered its membership within a multicast group, other nodes may set up connections to these nodes.

If the requesting node initiates a point-to-multipoint connection to the group address, the network will connect all nodes that are registered with that particular ATM address. Conversely, if the requesting node specifies a point-to-point connection, the network will set up a connection to the "nearest" registered node. In this way, anycast can be supported as a special case of group addressing, and a new addressing format is not required. However, many details of this procedure, including the format of the group addresses, had yet to be specified as of the time of writing. Routing aspects of group addressing are discussed in Section 4.4.

### 3.1 ATM and the OSI Model

An issue that often causes great confusion is that of to which layer in the OSI 7 layer model ATM corresponds. The adoption of the overlay model by the ATM Forum, as described in the previous section, sometimes cause some to describe ATM as a layer 2 protocol—that is, a data link protocol, akin to a MAC protocol like Ethernet or Token Ring. Yet this description is often contested by others who note that ATM possesses, most, if not all, of the characteristics of a layer 3 or network layer protocol, such as IP or IPX—such characteristics include a hierarchical address space and, as will be described in the next section, a complex routing protocol.

In practice, the question is moot—much of the controversy arises both from limitations of the OSI model, and from an incomplete understanding of the complexities of practical network operation. The basic OSI model did not incorporate the concept of overlay networks, where one network layer must overlay another, though such concepts were later added as addenda to the model. As we discussed in the previous section, such a model is often used where one type of network protocol must be carried transparently across another. Today, for instance, such layer 3 protocols as IP and IPX are often carried (tunneled) across other network layer protocols like X.25—or the telephone network, for instance—since this is generally much simpler than attempting to interoperate the protocols through a protocol gateway.

As noted in the previous section, the ATM overlay model was chosen so as to separate and hence facilitate the engineering efforts involved in both completing the ATM layer protocols, as well the efforts needed to modify existing protocols to operate with ATM. The overlay model also simplifies switch operation, at the arguable cost of redundancy in protocol functions and suboptimality in routing. As we will discuss later, the overlay model also leverages the existing installed application base, and facilitates future application portability, since it builds upon and extends today's ubiquitous network layer protocol infrastructure. Such trade-offs were felt by the Forum to be defensible, but in no way detract from the fact that ATM is indeed a full fledged network layer protocol—one, indeed, that is perhaps at least as complex as any that exists today.

What makes ATM a network layer protocol is indeed the very complexity of its addressing and routing protocols, and this is independent of the fact that other network layer protocols are run over ATM—indeed, as we will discuss later, the LAN Emulation protocols actually operate a MAC layer protocol over ATM, but this does not make ATM a physical layer.

A related issue that also causes confusion is the notion of "flat addressing" and whether or not ATM can be used to build a "simpler" network, in some sense, than today's network layer protocol based routed internetworks. This issue is coupled to the layering issue discussed above because some, as noted, draw a correspondence between ATM and layer 2 MAC protocols. As it happens, the latter do indeed have a flat address space—that is, 48 bit MAC addresses—and it is true that MAC layer internetworking devices—that is, MAC bridges—do offer "plug and play" capabilities, and do not require the complex configuration of layer 3 internetworking devices (that is, routers).

This simplicity comes from the fact that since MAC addresses are indeed flat—that is, they have no logical hierarchy— packets must be flooded throughout the network, using bridging protocols. While this requires no network configuration, it also greatly reduces the scalability—and stability— of such bridged networks. A hierarchical address space, together with address assignment policies that minimize (flat) host routes, permit the use of address aggregation, where reachability for entire sets of end systems can be summarized by a single address prefix (or, equivalently, by subnet masks). Coupled with a routing protocol that disseminates such address prefixes, hierarchical addressing precludes the need for flooding, and greatly reduces the amount of reachability information that must be exchanged.

Protocols with hierarchical, aggregatable address spaces do indeed generally require more configuration for address and subnet assignment, but by the same token this very hierarchy permits the operation of routing protocols, and hence the deployment of much more scalable and stable networks. Flat addressing, by definition, precludes routing and requires bridging, with consequent lack of scalability.

Indeed, very few networks, outside of bridged LANs, actually have a truly flat address space. The telephone network, for instance, which is often thought of as a flat network, actually incorporates a very structured hierarchy within its address space (that is, country code, area code, and so on), and it is only this rigid hierarchy that has permitted the telephone network to scale globally as it has. ATM networks certainly do not have a flat address space—indeed, as discussed in the previous section, the ATM address space has scope for an unprecedented level of hierarchical structure, and this structure is exploited in the ATM routing protocols we discuss below to support greater degrees of scalability within ATM networks than is possible within any other network.

Much of the discussion about flat addressing and ATM actually revolve around the perception that ATM networks can be made easier to administer than existing layer 3 networks. It is true that, for historical reasons, few efforts were made in the development of many current network layer protocols to facilitate ease of administration, though many such efforts are being made today, for instance as with the Dynamic Host Configuration Protocol (DHCP) [Droms], in the case of IP. Ease of administration argues not for flat addressing, however, but for a systematic focus on supporting autoconfiguration within protocols, as is now being done for the IP Next Generation (IPng or IPv6) protocol. This has been a prime focus for the ATM Forum from its inception, and by building on such mechanisms as the ILMI, most of the protocols developed for ATM, as we will discuss later in the paper, do incorporate such support.

## 4.0 ATM Routing Protocols

We now turn to the Network Node Interface (NNI) protocols used within ATM networks to route ATM signaling requests between ATM switches. Since ATM is connection oriented, a connection request needs to be routed from the requesting node through the ATM network and to the destination node, much as packets are routed within a packet-switched network. The NNI protocols are hence to ATM networks, what routing protocols (such as OSPF or IGRP) are to current routed networks.

The ATM Forum has an ongoing effort to define a Private NNI (P-NNI) protocol. The goal is to define NNI protocols for use within private ATM networks—or, more specifically, within networks that use NSAP format ATM addresses. Public networks that use E.164 numbers for addressing will be interconnected using a different NNI protocol stack based upon the ITU-T B-ISUP signaling protocol and the ITU-T MTP Level 3 routing protocol. This work, being carried out by the Broadband Inter-Carrier Interface (B-ICI) subworking group of the ATM Forum [Forum4], and other international standards bodies, is not discussed further in this paper.

The P-NNI protocol consists of two components: the first is a P-NNI signaling protocol used to relay ATM connection requests within the networks, between the source and destination UNI. The UNI signaling request is mapped into NNI signaling at the source (ingress) switch. The NNI signaling is remapped back into UNI signaling at the destination (egress) switch[18].

The P-NNI protocols operate between ATM switching systems (which can represent either physical switches or entire networks[19] operating as a single P-NNI entity), which are connected by P-NNI links. P-NNI links can be physical links or virtual, "multi-hop" links. A typical example of a virtual link is a virtual path that connects two nodes together. Since all virtual channels, including the connection carrying the P-NNI signaling, would be carried transparently through any intermediate switches between these two nodes on this virtual path, the two nodes are logically adjacent in relation to the P-NNI protocols.

The ILMI protocol, first defined for use across UNI links, will also be used across both physical and virtual NNI links; enhancements to the ILMI MIBs allow for automatic recognition of NNI versus UNI links, and of private versus public UNI.

[18] The ingress switch is known as the DTL originator, and the final egress switch as the DTL terminator, since these nodes respectively insert and remove the DTLs used to route the connection request through the network.

[19] A private ATM network, might use proprietary NNI protocols internally, and use the P-NNI protocol for external connectivity and interoperability.

The current P-NNI signaling protocol [Cherukuri] being developed by the ATM Forum is an extension of UNI signaling and incorporates additional Information Elements (IE) for such NNI-related parameters as Designated Transit Lists (DTL). P-NNI signaling is carried across NNI links on the same virtual channel, VCI=5, which is used for signaling across the UNI. The VPI value depends on whether the NNI link is physical or virtual.

The second component of the P-NNI protocol is a virtual circuit routing protocol. This is used to route the signaling request through the ATM network. This is also the route on which the ATM connection is set up, and along which the data will flow. The operation of routing a signaling request through an ATM network, somewhat paradoxically, given ATM's connection oriented nature, is superficially similar to that of routing connectionless packets within existing network layer protocols (such as IP). This is due to the fact that prior to connection set up, there is, of course, no connection for the signaling request to follow.

As such, a VC routing protocol can use some of the concepts underlying many of the connectionless routing protocols that have been developed over the last few years. However, the P-NNI protocol is much more complex than any existing routing protocol. This complexity arises from two goals of the protocol: to allow for much greater scalability than what is possible with any existing protocol, and to support true QoS-based routing.

The current state of the P-NNI protocols will be examined by looking at the manner in which the protocol tackles these challenges. It should be noted, however, that the ATM Forum is not currently scheduled to complete the "P-NNI Phase 1" protocol [Forum5] until August 1995. In the interim, the ATM Forum has defined a so called "P-NNI Phase 0" protocol, the Interim Inter-Switch Signaling Protocol (IISP) [Forum6]. This protocol will be examined after the Phase 1 protocol. Finally, multicast routing, how private and public ATM networks internetwork, and implementation consider-

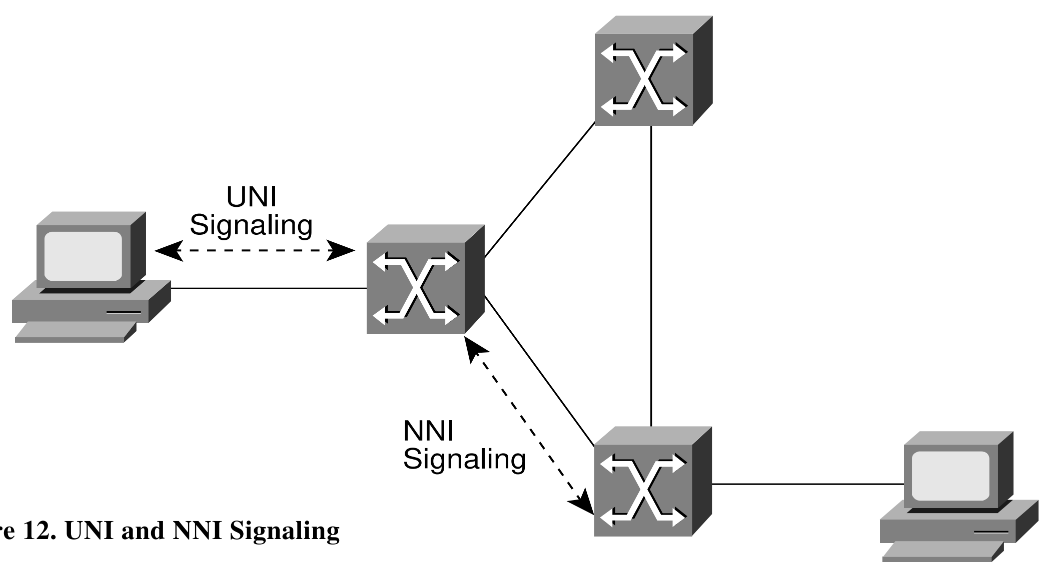



**Figure 12. UNI and NNI Signaling**

ations for P-NNI are discussed. Note, however, that since the P-NNI Phase 1 Protocol is still under development, the description given here may change before the specification is finalized.

Both the P-NNI Phase 1 protocol, and the IISP protocol, currently only will interface with, and support the capabilities of, UNI 3.0/3.1 signaling. In particular, neither of these protocols will support such aspects of UNI 4.0 signalling as leaf-initiated joins, group addressing, or ABR connection parameter negotiation. Such functionality will be added to the P-NNI protocols as part of a possible future P-NNI Phase 2 protocol specification.

## 4.1 P-NNI Phase 1: QoS Support

One of the great advantages of ATM is its support for guaranteed QoS in connections. Hence, a node requesting a connection set up can request a certain QoS from the network and can be assured that the network will deliver that QoS for the life of the connection[20]. Such connections are categorized into various types of ATM QoS types: CBR, VBR, ABR, and UBR, depending upon the nature of the QoS guarantee desired and the characteristics of the expected traffic types (see Appendix A). Depending upon the type of ATM service requested, the network is expected to deliver guarantees on the particular mix of QoS elements that are specified at the connection set-up (such as cell loss ratio, cell delay, and cell delay variation).

To deliver such QoS guarantees, ATM switches implement a function known as *connection admission control* (CAC). Whenever a connection request is received by the switch, the switch performs the CAC function. That is, based upon the traffic parameters and requested QoS of the connection, the switch determines whether setting up the connection violates the QoS guarantees of established connections (for example, by excessive contention for switch buffering). The switch accepts the connection only if violations of current guarantees are not reported. CAC is a local switch function, and is dependent on the architecture of the switch and local decisions on the strictness of QoS guarantees.

The VC routing protocol must ensure that a connection request is routed along a path that leads to the destination and has a high probability of meeting the QoS requested in the connection set up—that is, of traversing switches whose local CAC will not reject the call.

[20] In UNI 3.0/3.1, the traffic parameters and requested QoS for a connection cannot be negotiated at set-up, or changed over the lifetime of the connection. UNI 4.0 will support connection QoS negotiation; how this will be supported within P-NNI is for future study.

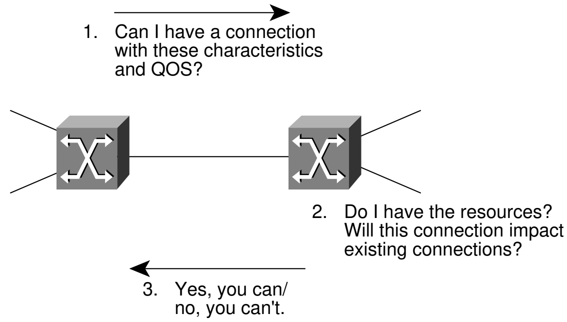



**Figure 13. Connection Admission Control**

To do this, the protocol uses a topology state routing protocol in which nodes flood QoS and reachability information so that all nodes obtain knowledge about reachability within the network and the available traffic resources within the network. Such information is passed within P-NNI topology state packets (PTSP), which contain various type-length-value (TLV) encoded P-NNI topology state elements (PTSE). This is similar to current link state routing protocols such as OSPF. Unlike these, however, which only have rudimentary support for QoS, the P-NNI protocol supports a large number of link and node state parameters that are transmitted by nodes to indicate their current state at regular intervals, or when triggered by particular events.

There are two types of link parameters: non-additive link attributes used to determine whether a given network link or node can meet a requested QoS; and additive link metrics that are used to determine whether a given path, consisting of a set of concatenated links and nodes (with summed link metrics), can meet the requested QoS.

The current set of link metrics are:

- Maximum cell transfer delay (MCTD) per traffic class[21].
- Maximum cell delay variation (MCDV) per traffic class
- Maximum cell loss ratio (MCLR) for CLP=0 cells, for the CBR and VBR traffic classes
- Administrative Weight: This is a value set by the network administrator and is used to indicate the desirability or otherwise of a network link.

The current set of link attributes are:

- Available Cell Rate (ACR): A measure of the available bandwidth in cells per second, per traffic class

[21] Note that it is implicitly assumed that nodes can ensure adequate levels of separation between the different types of traffic passing through the node so that one traffic class does not consume the resources reserved for another traffic class.

- Cell Rate Margin (CRM): A measure of the difference between the effective bandwidth allocation per traffic class, and the allocation for sustainable cell rate; this is a measure of the safety margin allocated above the aggregate sustained rate
- Variance Factor[22] (VF): A relative measure of CRM margin normalized by the variance of the aggregate cell rate on the link

All network nodes can obtain an estimate of the current state of the entire network through flooded PTSPs that contain such information as listed above. Unlike most current link state protocols, the P-NNI protocol advertises not only link metrics, but also nodal information. Typically, PTSPs include bidirectional information about the transit behavior of particular nodes based upon entry and exit port, and current internal state. This is particularly important in cases where the node represents an aggregated network (that is, a peer group—see below). In such a case, the node metrics must attempt to approximate the state of the entire aggregated network. This internal state is often at least as important as that of the connecting links for QoS routing purposes.

The need to aggregate network elements and their associated metric information also has important consequences on the accuracy of such information, as discussed below.

Two approaches are possible for routing a connection through the network: hop-by-hop routing and source routing. Hop-by- hop routing is used by most current network layer protocols such as IP or IPX, where a packet is routed at any given node only to another node—the "next hop"—closer to the final destination. In source routing, the initial node in the path determines the entire route to the final destination.

Hop-by-hop routing is a good match for current connectionless protocols because they impose little packet processing at each intermediate node. The P-NNI protocol, however, uses source routing for a number of reasons. For instance, it is very difficult to do true QoS-based routing with a hop-by-hop protocol since each node needs to perform local CAC and evaluate the QoS across the entire network to determine the next hop. Hop-by-hop routing also requires a standard route determination algorithm at each hop to preclude the danger of looping.

[22] There is currently some controversy as to whether the CRM and VF add much value to the GCAC— the traffic passing through ATM switches may prove to be so irregular (for example, cell peaks may be bunched) that such second order statistics may prove to be too volatile and yield little useful information. Calculating such statistics is also non-trivial, particularly in the presence of aggregation.

However, in a source-routed protocol, only the first node would ideally need to determine a path across the network, based upon the requested QoS and its knowledge of the network state, which is gained from the PTSPs. It could then insert a full source routed path into the signaling request that would route it to the final destination. Ideally, intermediate nodes would only need to perform local CAC before forwarding the request. Also, since it is easy to preclude loops when calculating a source route, a particular route determination algorithm does not need to be standardized, leaving this as another area for vendor differentiation.

This description is only ideal, however. In practice, the source routed path that is determined by a node can only be a best guess. This is because in any practical network, any node can have only an imperfect approximation to the true network state because of the necessary latencies and periodicity in PTSP flooding. As discussed in the next section, the need for hierarchical summarization of reachability information also means that link parameters must also be aggregated. Aggregation is a "lossy" process, and necessarily leads to inaccuracies. Furthermore, as noted above, CAC is a local matter. In particular, this means that the CAC algorithm performed by any given node is both system dependent and open to vendor differentiation.

The P-NNI protocol tackles these problems by defining a Generic CAC (GCAC) algorithm. This is a standard function that any node can use to calculate the expected CAC behavior of another node, given that node's advertised additive link metrics, described above, and the requested QoS of the new connection request. The GCAC is an algorithm that was chosen to provide a good prediction of a typical node-specific CAC algorithm, while requiring a minimum number of link state metrics. Individual nodes can control the degree of stringency of the GCAC calculation involving the particular node by controlling the degree of laxity or conservativeness in the metrics advertised by the node.

The GCAC actually uses the additive metrics described above; indeed these metrics were selected to support the GCAC algorithm chosen for the P-NNI protocol. Individual nodes (physical or logical) will need to determine and then advertise the values of these parameters for themselves, based upon their internal structure and loading. Note, however, that the P-NNI Phase 1 GCAC algorithm is primarily designed for CBR and VBR connections; variants of the GCAC are used depending upon the type of QoS guarantees requested and the types of link metrics available, yielding greater or lesser degrees of accuracy.

The only form of GCAC done for UBR connections is to determine whether a node can support such connections. For ABR connections, a check is made to determine whether the link or node is authorized to carry any additional ABR con-

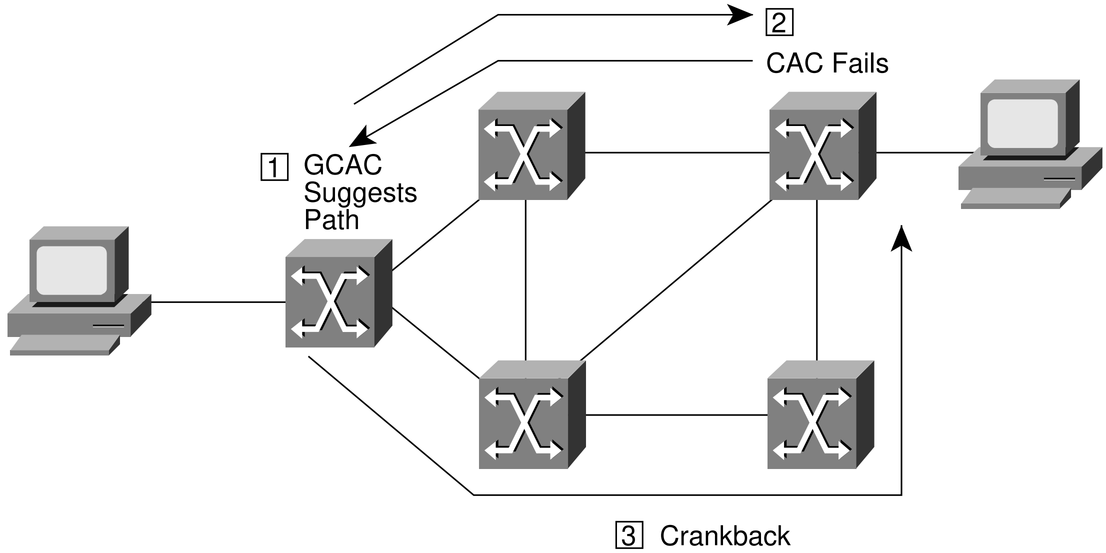



**Figure 14. Operation of Crankback**

nections and to ensure that the ACR for the ABR traffic class for the node is greater than the Minimum Cell Rate specified by the connection.

The details of the GCAC are described in [Forum5].

Using the GCAC, a node presented with a connection request (which passes its own CAC) processes the request as follows:

1. All links that cannot provide the requested ACR, and those whose CLR exceeds that of the requested connection, are "pruned" from the set of all possible paths using the GCAC.
2. From this reduced set, along with the advertised reachability information, a shortest path computation is performed to determine a set of one or more possible paths to the destination.
3. These possible paths are further pruned by using the additive link metrics, such as delay, and possibly other constraints. One of the acceptable paths would then be chosen. If multiple paths are found, the node may optionally perform tasks such as load balancing.
4. Once such a path is found (note that this is only an "acceptable" path to the destination, not the "best" path, the protocol does not attempt to be optimal), the node constructs a designated transit list (DTL) that describes the complete route to the destination (the structure of the DTL is described below) and inserts this into the signaling request. The request is then forwarded along this path.

This, however, is not the end of the story. Each node in the path still performs its own CAC on the routed request because its own state may have changed since it last advertised its state within the PTSP used for the GCAC at the source node. Its own CAC algorithm is also likely to be somewhat more accurate than the GCAC. Hence, notwithstanding the GCAC, there is always the possibility that a connection request may fail CAC at some intermediate node. This becomes even more likely in large networks with many levels of hierarchy, since QoS information cannot be accurately aggregated in such cases. To allow for such cases, without excessive connection failures and retries, the P-NNI protocol also supports the notion of *crankback*.

Crankback is where a connection which is blocked along a selected path is rolled back to an intermediate node, earlier in the path. This intermediate node[23] attempts to discover another path to the final destination, using the same procedure as the original node, but uses newer, or hopefully more accurate network state. This is another mechanism that can be much more easily supported in a source-routed protocol than in a hop-by-hop protocol.

One of the concerns with P-NNI route generation is that most commonly used routing algorithms (such as Dijkstra calculations) were designed for single, cumulative metrics such as link weightings or counts. Since P-NNI uses a number of complex link parameters for link pruning, path selection may often not generate any acceptable paths. In such cases, sophisticated algorithms may use a technique known as *fallback,* where particular attributes (such as delay) are selectively relaxed, and paths are recalculated in order to find a path that meets some minimal set of desired attributes. In general, path selection, like CAC, is an area with considerable scope for vendor differentiation.

[23] Only nodes that actually construct DTLs perform crankback, as described below.

## 4.2 P-NNI Phase 1: Scalability and Reachability

In addition to providing true QoS support, the ATM Forum has also set the goal of universal scalability for the P-NNI Phase 1 protocol. The P-NNI Phase 1 protocol is being designed to be capable of being applied both to small networks of a few switches and to a possible future global ATM Internet comprising millions of switches. Such scalability is well beyond that of any single routing protocol today. The Internet, for instance, supports many different types of routing protocols—intra-domain routing protocols, such as IGRP or OSPF, which scale to large enterprise networks, and inter-domain protocols, such as BGP or IDRP, which interconnect such lower level networks. By building upon the many years of experience gained in the development of such current protocols, however, the ATM Forum hopes to build a single protocol that could perform at all levels within a network.

The key to such a scalable protocol is hierarchical network organization, with summarization of reachability information between levels in the hierarchy. Protocols such as OSPF implement such mechanisms, but only implement two level of hierarchy, which is inadequate for very large networks. The P-NNI protocol, however, uses the 20-byte NSAP addresses to identify levels in the network hierarchy to support an almost limitless number of levels: a maximum of 105 (the number of bits in the 13 high-order bytes of the NSAP address, excluding the ESI and SEL fields), though no more than a half dozen or so will likely ever need to be used, and even then only within the very largest, global networks.

To support this hierarchy, the P-NNI model defines a uniform network model at each level of the hierarchy. The P-NNI hierarchical model explains how each level of the hierarchy operates, how multiple devices or nodes at one level can be summarized into the higher level, and how information is exchanged between levels. The model is recursive in that the same mechanisms used at one level are also used at the next level.

Each level in the hierarchy consists of a set of logical nodes, interconnected by logical links. At the lowest level, each logical node represents a physical switching system consisting of a single physical switch, or a network of switches that internally operate a proprietary NNI protocol and support the P-NNI protocol for external connectivity. At this lowest level, each switching system must be assigned a unique ATM NSAP address.

Nodes within a given level are grouped into sets known as a *peer group*. The definition of a peer group is a collection of nodes that all obtain the identical topological database and exchange full link state information with each other. While all nodes within a peer group have complete state information on each other, peer groups cannot be extended too widely since this would lead to excessive PTSP traffic and processing. Hence, peer groups are organized hierarchically and are associated with a higher level parent peer group.

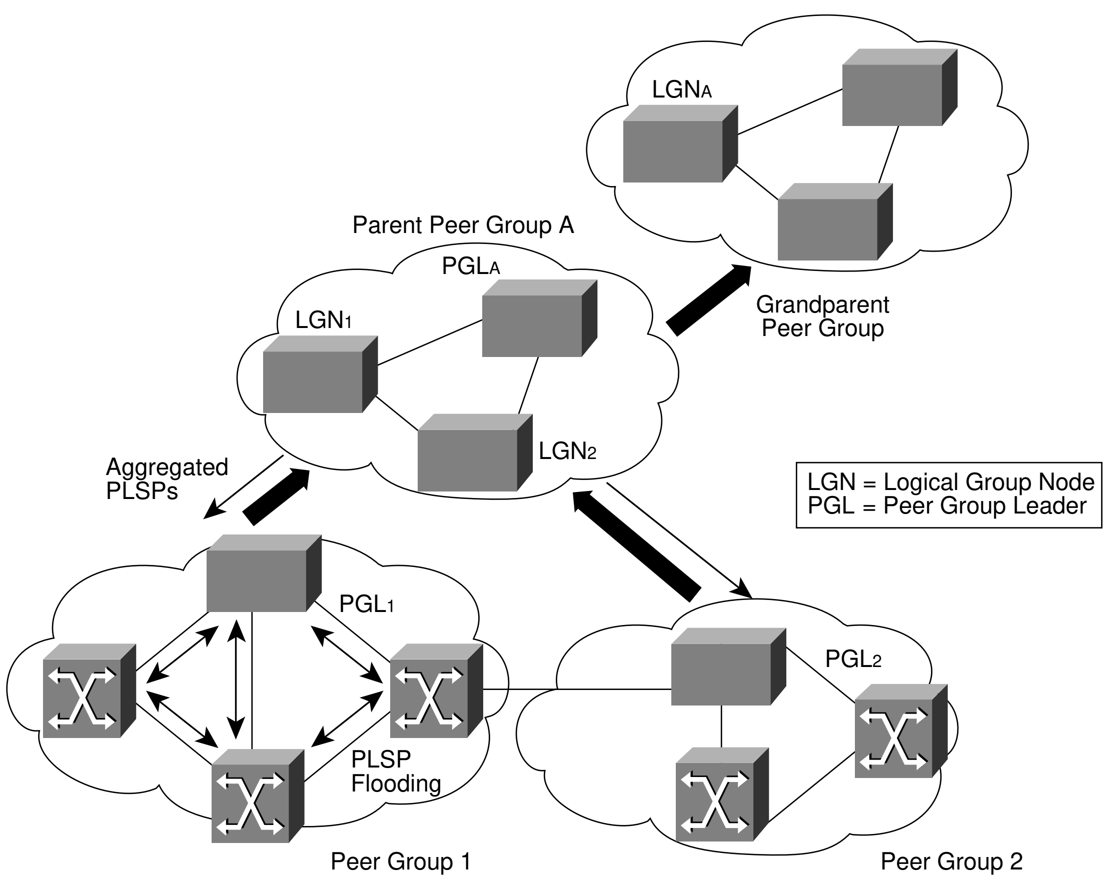



**Figure 15. The P-NNI Network Hierarchy Model**

Within its parent peer group, each peer group is represented, by default, as a single logical node, known as the *logical group node*. Within the parent peer group, the logical group node acts as a normal node, exchanging PTSPs with the other nodes within the parent peer group. The peer groups represented by logical group nodes within a parent group are known as the *child peer groups* of that group.

Normally, peer groups are identified by strict prefixes of private ATM addresses. At the lowest level, where switching systems consist of actual switches, and where by default, all end systems connected to a switch obtain their network address prefix from that of the switch (which implies that end system reachability defaults to switch reachability), the default peer group ID is the high 12 order bytes of the switch NSAP address. This allows for up to 256 switches within this lowest level peer group, without requiring any manual configuration of peer group IDs of the switches or configuration of the end systems.

At higher levels, the default for a peer group ID is a prefix on a lower level peer group ID. The peer group ID of a parent must be shorter than the prefix of its child peer group ID; this makes it easy to determine the relationship between two peer groups, and precludes the formation of a peer group hierarchy loop. Hence, the peer group ID becomes smaller as the hierarchical level becomes larger.

Nodes within a peer group are identified by a 22-byte node identifier. At the lowest level, this is essentially the same as the switching system's ATM address. At higher levels, the node ID (which now identifies logical group nodes) includes two level indicators that indicate the hierarchical level (that is, prefix length) of both the associated peer group and the child peer group, plus the peer group ID.

In addition to nodes, the P-NNI protocol also requires that links be identified since links between peer groups need to be identified in PTSPs and may also be optionally specified in DTLs. Since ATM link attributes can be asymmetrical (since connections may be asymmetrical), links are identified by a combination of a transmitting node ID and a locally assigned port ID. Nodes exchange such port IDs between themselves (using the Hello protocol discussed below) and hence together identify particular links. In practice, link identification is somewhat more complex, since multiple physical or virtual links[24] may need to be aggregated. (Refer to [Forum5] for more details.)

[24] The P-NNI protocol supports redundant links between switching systems, where the switches can locally perform connection level load sharing across the links. Note, however, that a single connection cannot be split across multiple links, since cell sequencing must be preserved within ATM connections; ATM cells do not carry sequence numbers.

Each peer group elects a single node[25] within the group to perform the functions of the logical group node. This node, known as the *peer group leader* (PGL), is selected through an election mechanism and is based upon a "leadership priority" and the switches' node ID. Each PGL is identified by a unique ATM address; if a node acts as a PGL within multiple levels of peer groups, then it must have a unique ATM address at each of those levels.

PGLs within each peer group have the responsibility of formulating[26] and exchanging PTSPs with their peer nodes within the parent peer group to inform those nodes of the child group's reachability and attributes[27]. Similarly, recursive information obtained by the PGL about the parent group and that group's parent groups are then fed down by the PGL into the child group. The child nodes can then obtain knowledge about the full network hierarchy, in order to construct full source routes.

Note, however, that the information that is fed down from the top level peer group all the way to the lowest level groups represent more and more aggregated (summarized) information. Hence, at the lowest level, the nodes will have full information about its own peer group, aggregated information about its parent group, more aggregated information about its "grandparent" group, and so forth. In order for PGLs to communicate with each other, however, they must have reachability information about the way in which the peer groups are linked together. This information is gathered by the P-NNI bootstrap procedure, using the P-NNI Hello protocol operating across P-NNI links.

P-NNI Links—be they physical or virtual—are further categorized within the P-NNI model. Horizontal, or inside, links connect two nodes within the same peer group. Exterior links connect nodes within a peer group to other exterior nodes that do not operate the P-NNI protocol. Outside links connect together two border nodes within two different peer groups, where *border nodes* are those nodes within a peer group that have links to nodes—"outside neighbors"—within other peer groups.

[25] However, the information advertised by the logical group node is a function of the state of the entire peer group, and is hence independent of the identity of the PGL.

[26] This also requires the PGL to determine, based upon the PTSPs exchanged within the peer group, and local (unspecified) algorithms, the corresponding link state parameters for the entire aggregated peer group.

[27] This does not mean, however, that PGLs need to process all requests traversing the peer group—this is done only by the border nodes of the peer group through which a connection request enters and leaves the peer group, and the intermediate switches connecting the two, as described below. A border node, however, could also act as a PGL.

Nodes first discover each other through a P-NNI Hello protocol in which nodes exchange Hello packets at regular intervals[28] with their immediate neighbor nodes.

If two neighbors discover that they are within the same peer group, by comparison of their peer group IDs, they start to send PTSPs to each other and synchronize their reachability databases. Once the nodes have synchronized their databases, they flood PTSPs throughout the peer group (i.e. across horizontal links) to ensure rapid convergence.

The P-NNI Hello packets and PTSPs are sent on a well known virtual channel, VCI=18 within VPI=0 for physical links, and within the appropriate VPI value for logical links. Mechanisms such as flooding, sequence numbers, "lock-step" acknowledgments, and checksums are used (instead of an ATM-specific data link protocol, such as SSCOP) to ensure reliable and timely delivery of PTSPs. As with other link state protocols, PTSPs are sent at regular intervals or when triggered by a significant event[29] (such as a quantum of change within bandwidth allocation on a link).

Two border nodes will also discover each other, across an outside link, through the Hello protocol, which will show that the two nodes have different peer IDs. Two border nodes exchange peer ID information across an outside link to determine the lowest level at which the ancestors of the two nodes are themselves peers (i.e. the two nodes must, by definition, have in common some ancestor, be it a parent, grandparent, etc.). Each border node then determines that the outside link is an *uplink* to that outside ancestor peer group. The two border nodes exchange metric information about the outside link in the Hello protocol, then advertise the uplink, and its characteristics, throughout their respective peer groups using PTSP.

At higher levels of the P-NNI hierarchy, multiple outside links may be aggregated together into fewer logical uplinks, but information about the binding between logical uplinks and their constituent outside links must be advertised so that nodes can map a logical inter-peer group link into a physical link.

[28] Hence the Hello protocol can also be used to detect link failures, though lower level mechanisms would generally detect a failed link faster.

[29] Specifically, a PTSP is triggered by a significant change in any topology information group (TIG), of which six are currently defined: nodal information, internal reachable ATM addresses, external reachable ATM addresses, pairwise nodal metrics, horizontal links, and uplinks. A "hold-down" timer is used to ensure that PTSP are not sent at unacceptable high rates. The P-NNI specification defines what a "significant" change is for each of the particular TIGs—refer to [Forum5] for more details.

Border nodes also exchange information about the PGLs of their own peer groups. This allows the PGLs of groups that discover that they are within the same parent peer group to set up connections to each other, across the identified uplinks, and start exchanging their own Hellos and PTSPs. They then discover the existence of yet higher level peer groups, until all nodes discover their entire network hierarchy. Through fed-down PTSPs, containing summarized reachability and uplink information, the PGLs discover full network state. A full example of P-NNI bootstrapping and discovery is given in [Forum5] and [Swallow].

Once full state information is obtained by all nodes, they can then use this to route signaling requests. When a signaling request is received across a UNI by an ingress switch—the DTL originator—the switch will use a shortest path algorithm, such as a Dijkstra calculation, to determine one or more paths that connect the source node to the desired destination, using the algorithm described in the previous section. This calculation will create a hierarchically complete source route, that is, a set of DTLs, which will have: a full, detailed path within the source node's own peer group; a less detailed path within the parent peer group; and even less detail on higher level peer groups, terminating in the lowest level peer group[30] which is an ancestor of both the source and destination nodes.

These DTLs are arranged in a stack within the P-NNI signaling request where each DTL contains the path elements for one level in the hierarchy. This comprises a list of node and, optionally, link IDs, together with a pointer that indicates which element in the list is to processed next. Within a given peer group, that peer group's DTL is processed by nodes until it reaches a node that is a border node to the next peer group on the path. At this point, the DTL of that peer group is exhausted, since the final element in that DTL is the ID of the border node. The border node then removes that DTL, notes that the next DTL points to the neighbor peer group (possibly at a different level in the hierarchy), and forwards it to its peer border node within that neighbor peer group.

Once the request arrives at that border node within that neighbor peer group, that node discovers that the request must be routed through that node's peer group. Typically, however, the original DTL only has aggregated information about this neighbor peer group. The border node then constructs one or more new DTLs, describing how to route the request through its peer group and "pops" it onto the top of the stack of DTLs. In this way, the request is forwarded to a border node within this peer group, which performs a similar function for the next peer group in the path, and so on, until the final destination peer group is reached.

[30] Hence a request does not need to traverse the entire hierarchy—only as high as is necessary to get to a path between the source and the destination.

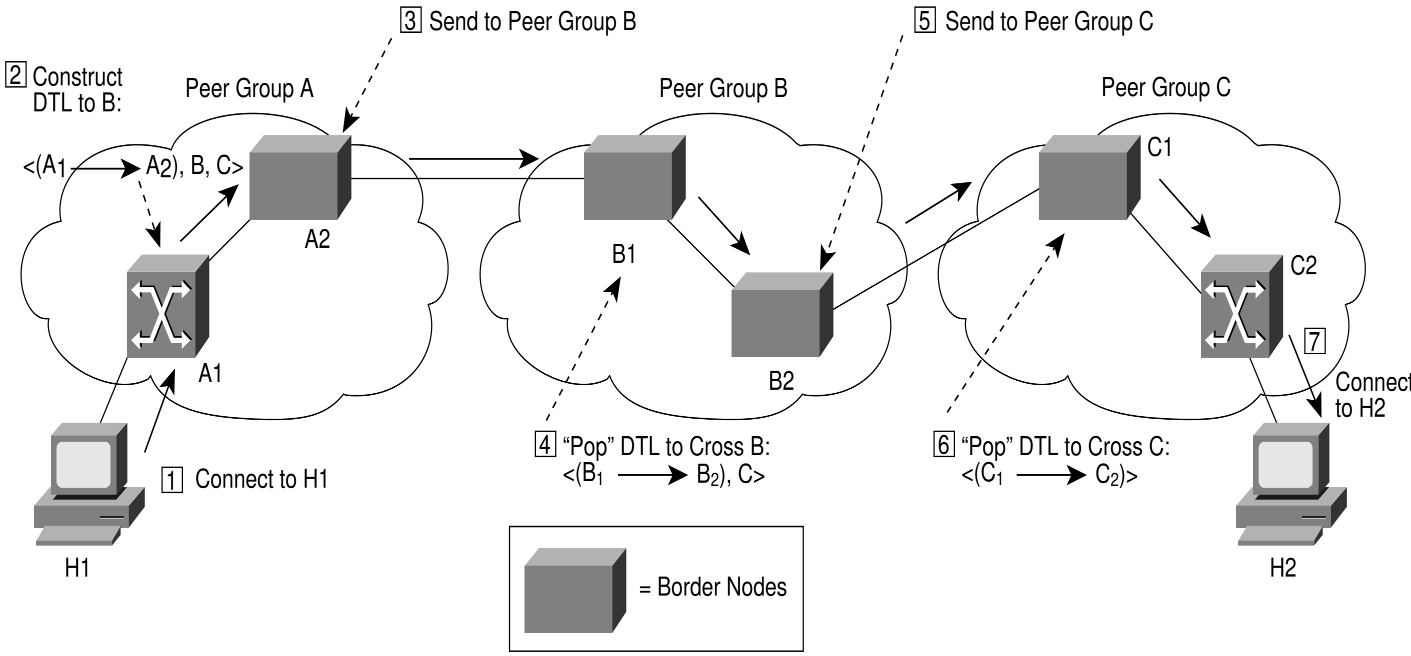



**Figure 16. DTL Processing in Connection Setup**

At this point, the (ingress) border node will construct a DTL that routes the request to the switch on which the destination end system is attached. There, the final switch—the DTL terminator—re-maps the request into UNI signaling and forwards it across the appropriate UNI link. DTLs are hence only created by the source node and by border nodes. Other intermediate nodes only process DTLs and move the DTL pointer forward and pass the request to the next node on the path.

Crankback works within this same mechanism; to make the previous description more precise, connections can only be cranked back to nodes that actually create and insert DTLs into a request—the original source node, or ingress border nodes. Such nodes maintain state information about all requests that they have forwarded until the connection set up is confirmed, or a connection reject is received from the destination end system. If, however, an intermediate node rejects the call (for example, due to local CAC), then the call is rerouted back along the path that it followed to that node to the last node to insert a DTL. If possible, this node then recalculates a new path across its own peer group, avoiding the node that rejected the call, and re-forwards the request.

Good examples of the operation of both P-NNI routing and crankback are given in [Forum5] and are highly recommended, since a proper description of the P-NNI procedures is outside the scope of this paper.

While the procedures outlined here can be scaled to very large networks, it should be noted that the aggregation used to ensure such scalability also fundamentally works against the QoS routing properties of ATM. This is because the QoS metrics discussed in the previous section must also be aggregated to match the aggregation of network topology inherent in the network hierarchy; aggregation, however, is a fundamentally “lossy” process. At the lowest level, such metrics might yield information about the state of particular switch and link combinations. At higher levels, the same metrics must attempt to approximate the “average” state of entire networks, which consists of many individual switches.

Clearly such aggregated information will be much less accurate than information about individual switches. This problem is exacerbated by the fact that at higher levels entire peer groups are represented by single nodes (that is, logical group nodes). Advertising metrics about such nodes imply an assumption about the symmetry and compactness of the topology of the child peer group and its traffic flows, which is very unlikely to be accurate in practice.

To ameliorate this problem, the P-NNI protocol allows a peer group to be modeled at higher levels, for advertising purposes, not as a single node but as a “complex node,” with an internal structure. The Phase 1 P-NNI protocol allows complex nodes to be modeled as a star of nodes that consists of a “pseudo-node” connected to a group of border nodes across “pseudo-links,” each with an identical radius[31] for each link parameter. These nodes need not necessarily correspond to any actual physical node, but the hope is that the “radius” advertised for this abstract network better represents the metrics across the actual peer network, than by modeling it by a single node. Modeling peer groups in this fashion

[31] Some of the pseudo-links could also be marked as “exceptions” and could advertise a different radius, though at the cost of ever increasing complexity in the PTSPs. Border nodes can also optionally advertise metrics for direct connections between themselves, bypassing the central node, hence forming a (partial) mesh.

require much more information to be advertised and modeled within PTSPs. There are more complex and possibly more accurate ways to model a peer group other than a star (such as a mesh or spanning tree). Future phases of the P-NNI protocol might allow for these alternate models of complex nodes.

In addition to summarized addresses, a number of other elements of reachability information are also carried within PTSP. Routes to external networks, reachable across exterior links, are advertised as external addresses. Peer groups may also include nodes with non-aggregatable addresses, which must also be advertised, as must registered group and anycast addresses. Generally none of these types of information can be summarized, since they fall outside the scope of the default P-NNI address hierarchy.

Note that the scope of advertisement of the group addresses is a function of how the network administrator maps the administrative scope of a registered node to the corresponding P-NNI hierarchy.

The P-NNI protocol also has support for "soft permanent virtual connection" set-up [Grossman]. The latter is a means of setting up PVCs and permanent virtual paths (PVP) using P-NNI procedures. Through network management, a PVC or PVP is established only across the source and destination UNI, but not across the entire network. Then, through network management the first (ingress) switch is instructed to route a connection across the network to the destination (egress switch) using P-NNI. This is done with the usual P-NNI procedures, but hooks in the signaling instruct the destination switch to terminate the connection on the pre-established PVC/PVP, rather than forwarding a UNI signaling request to the destination end-system.

Given the need to use permanent connections (because end-systems do not support signaling, for instance), soft connection set-up is a much more convenient and reliable way to set up such connections rather than using hop-by-hop configuration. This also allows permanent connections to be set up with a specific QoS using the P-NNI procedures.

## 4.3 The IISP Protocol

While the P-NNI Phase 1 protocol is extremely powerful, it is also quite complex. For this reason, the ATM Forum's work on the protocol is unlikely to be completed until the second half of 1995. Actual interoperable implementations are unlikely to be widely deployed until well into 1996. For instance, as of the time of writing, many vendors currently had yet to fully roll out implementations of UNI 3.0 signaling, despite the fact that this standard was completed in September 1993. Clearly, the P-NNI Phase 1 protocol is much more complex than UNI 3.0.

Unfortunately, without a P-NNI protocol, there is no standard way for users to build interoperable multivendor ATM networks. Many users are not willing to wait until 1996 for such interoperability since they have pressing needs to test multiple vendor's switches within the ATM test beds that they are currently running. To solve this short-term protocol, Cisco Systems proposed to the ATM Forum that it develop a very simple, UNI-based signaling protocol for switch interoperability [Alles1].

Originally designated the P-NNI Phase 0 protocol, this was later renamed the Interim Inter-Switch Signaling Protocol (IISP) to avoid confusion with the P-NNI Phase 1 protocol. This protocol was recently completed and approved by the ATM Forum [Forum6]. The IISP, as the name suggests, is essentially a signaling protocol for inter-switch communication. Given the fact that the UNI 3.0/3.1 signaling procedures are essentially symmetrical, it uses UNI signaling for switch-to-switch communication, with nodes arbitrarily taking the role of the network and user side across particular switch-to-switch links (known as IISP links).

Signaling requests are routed between switches using configured address prefix tables within each switch, which precludes the need for a VC routing protocol. These tables are configured with the address prefixes that are reachable through each port on the switch. When a signaling request is received by a switch, either across a UNI or an IISP link, the switch checks the destination ATM address against the prefix table and notes the port with the longest prefix match. It then forwards the signaling request across that port using UNI procedures.

The IISP protocol is very simple and does not require modification to UNI 3.0/3.1 signaling or any new VC routing protocol. It can leverage current development efforts on UNI signaling and hence can be deployed very quickly. The IISP, however, does not have anywhere near the same scalability as the Phase 1 protocol. For instance, manually configuring prefix tables limits its applicability to networks with only a small number of nodes. This is adequate for now, given that most ATM switches today are deployed in small test beds and not in large scale production networks.

IISP implementations will not be interoperable with P-NNI Phase 1 implementations[32] because IISP only uses UNI and not NNI signaling. Users will need to upgrade their switches when P-NNI Phase 1 becomes available. This was deliberately done to simplify the specification and accelerate the deployment of IISP, and to emphasize its interim nature.

[32] A P-NNI Phase 1 node will treat an IISP link as an exterior link, and will advertise the address prefixes reachable through that link as external addresses.

The IISP also does not support QoS-based routing, although nodes may implement CAC; it does not support crankback, though nodes can be configured with redundant or alternate paths (the selection of such paths being a local matter). These limitations of the IISP, however, are not as restrictive as might first be imagined. While the Phase 1 protocol has extensive support for QoS routing, this is required only for routing VBR and CBR connections, where end systems can request a specific QoS. End systems that request either Unspecified Bit Rate (UBR) or Available Bit Rate (ABR) connections, however, can specify only very limited QoS capabilities. As such, the P-NNI protocol metrics do not apply to such connections and must be routed using some other criteria—such as shortest path[33].

Most data traffic on ATM networks will likely use UBR or ABR connections in the short to medium term, since higher layer protocols cannot specify QoS (and hence use VBR connections). Given these factors, it is likely that IISP will be widely deployed prior to the final specification and deployment of the P-NNI Phase 1 protocol, though it will certainly by supplanted by the latter as it becomes available.

## 4.4 Multicast Routing

In the first instance, with UNI 3.0/3.1, point-to-point connections will be set up a leaf at a time, with each add-leaf request addressed by the leaf's unicast ATM address. Hence such connection requests will be routed by IISP and the P-NNI Phase 1 protocol in the same manner as point-to-point connections.

The only difference is that the signaling procedures will ensure that no new connections are set up across a link for a particular add-leaf request if a branch of the point-to-multipoint connection already exists across that link. Ideally, a new branch of the tree will be added only at the point "closest" to the new leaf, where the connection must branch off to the new leaf. In terms of the P-NNI Phase 1 operation, this may impact the selection of possible routes during the route pruning phase.

Through this support of point-to-multipoint connections, the P-NNI Phase 1 and IISP protocols will support existing UNI 3.0/3.1 multicast mechanisms such as multicast servers and overlaid point-to-multipoint connections.

With UNI 4.0, support will need to be added for group addressing. Reachability information about registered group addresses can be advertised within PTSP in the Phase 1 protocol, and can be configured within the IISP protocol. This does not address, however, the support of such new UNI 4.0 mechanisms as leaf-initiated joins and the addition of multiple leaves in a single point-to-multipoint connection request. Such issues were deferred by the P-NNI group to a possible Phase 2 effort.

This effort may tackle ways to automatically configure[34] groups of ATM end-points into some form of multicast group, based upon their registration of membership within the multicast group. Support will also be needed for a multicast routing protocol to allow for point-to-multipoint connections to group addresses, since the P-NNI protocols will then need to generate a source rooted tree linking the source to each of the leaves. Such a protocol may build upon such existing multicast protocols as Protocol Independent Multicast (PIM) [Deering2].

## 4.5 Public Network Internetworking

One area in both the P-NNI Phase 1 and IISP protocols that is still not fully specified is that of public network internetworking. The interconnection of private ATM networks across public ATM networks poses particular challenges because of the current lack of public SVC services, and the likely nature of such services when they are deployed.

Currently, many public network service providers are considering the deployment of public ATM networks, which will offer an ATM interconnect service across public UNI to private ATM systems. In the first instance, it is likely that the service offered across such networks will not be a pure ATM service, but will be ATM-based variants of such existing WAN technologies as Frame Relay or the Switched Multimegabit Data Service (SMDS). These services will be described in Section 8.0. Here, however, we consider private-public ATM internetworking, assuming that the public network does indeed offer a native ATM service.

The first problem likely to be faced with such internetworking is that, for various technical, administrative, and tariffing reasons, it is likely that the majority of initial public ATM services will not support switched virtual connections across public UNI[35]. This is a cause for concern since most private ATM networks primarily use SVCs. A method must be found to at least convey ATM signaling information between two private network switching systems across the public network, even if the public network does not process the signaling

[33] Some have proposed that the P-NNI protocol should attempt some sort of network load balancing for UBR and ABR connections by routing such connections along paths with the smallest number of such pre-established connections. It is not clear what benefits this would provide since one link may have a large number of such connections, each of which uses little bandwidth; another link may have a few such connections that use very large amounts of bandwidth.

[34] Protocols such as LAN Emulation, which today use multipoint connections, have defined their own mechanisms for determining multicast group membership in the absence of any ATM specific mechanism.

information. One way in which this might be done is through a technique known as “Permanent Virtual Path (PVP) tunneling.” In this method, two private ATM networks are linked across the public network using a virtual path in which the public network transparently trunks the entire collection of virtual channels in the VP between the two sites.

Signaling requests from one private network at the Public UNI would then be mapped into the appropriate virtual channel (that is, VCI=5) within the VP from the usual (VPI=0, VCI=5) virtual channel by the egress private network switch, and carried transparently across to the ingress switch in the other private network. At this point, this switch would map the signaling request back into the usual channel and propagate it across the destination network. Note that if the two networks were also running the P-NNI (or IISP) protocols, then this PVP across the public network would be treated as a virtual link. Hence the link between the private and public network would simultaneously be a Public UNI and a virtual P-NNI link. The only change PVP tunneling requires in normal node operation is that procedures must be used by the ingress and egress switches to allocate particular channels within the PVP to particular connection requests (as opposed to VPI=0, which is the normal operation), as they are passed.

While PVP tunneling does at least allow for signaling to be passed across the public network, it still requires manual configuration (such as through subscription) of connections across the Public UNI. To eliminate this restriction and permit ubiquitous connectivity (at least within the policy and administrative restrictions imposed by the public network service provider), signaling needs to be supported across the Public UNI. One complexity in doing this, however, is P-NNI internetworking, or the lack therefore, across the Public UNI.

It is likely that most public network service providers will not, in fact, support the P-NNI protocol within their networks, since they usually do not wish to display their internal network structure to users. As discussed above, public networks typically operate only with E.164 numbers, not NSAP format private ATM addresses, and internally run their own NNI protocols. This raises two issues: how private networks can obtain reachability information about the public network and how private network addresses can be carried through the public network.

With respect to the first problem, there have been proposals that variants of border routing protocols such as the Inter-Domain Routing Protocol (IDRP) be used to insert public network connectivity information into P-NNI networks as external routes. Alternatively, it has been proposed that the entire public network could be viewed as a single peer group within the P-NNI hierarchy. In general, however, it is likely that public networks will not offer, at least initially, any kind of reachability information at all to private networks. The likely result is that private networks will treat the public network as a subnetwork and will simply tunnel signal requests across it, much as current network layer protocols run across such networks as X.25 or across dial-up networks.

Such tunneling may use the subaddress fields defined in the UNI signaling procedures. At the egress switch from a private network, prior to forwarding the signaling request across the public network, the egress switch will move the destination NSAP format address into the destination subaddress field and will replace the destination address field with the E.164 address that corresponds to the Public UNI of the switch which provides the ingress to the destination private

[35] That is, private network nodes will not be able to request connections across the public network using UNI signaling, but will need to obtain permanent connections across the UNI through subscription. Internally to the public network, however, NNI protocols may be used to provision such permanent connections.

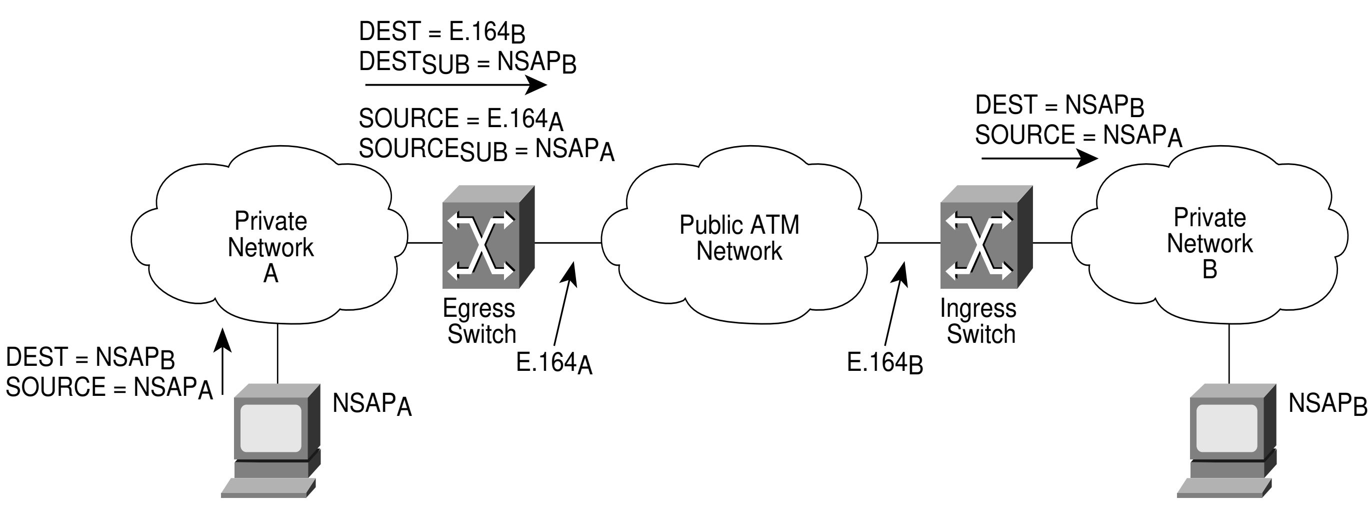



**Figure 17. Address Re-mapping at Public UNI**

network[36]; correspondingly, the source NSAP format address will be moved into the source subaddress field, and replaced with the E.164 number of the egress node's Public UNI.

This signaling request will then be forwarded into the public network, which will then route it, using the destination E.164 number, across to the destination public UNI, using internal NNI protocols. At the ingress switch to the destination private network, the ingress switch will move the destination and source NSAP addresses back into the main address fields, and will process the request as normal. Note that this procedure would be needed to make the initial connection, even if the private networks were to subsequently tunnel the P-NNI protocol across the public network.

The remaining issue with this method is how the private network switches obtain the information to map destination NSAP format addresses to the E.164 numbers of the UNI through which they are reachable. In the first instance, this will almost certainly be done through manual configuration, much as is done today for dial-up lines, for instance. In the future, there have been proposals for a public network directory service, which private network nodes could query to obtain such mappings. In general, however, as of the time of writing, there is little consensus on how public network ATM internetworking would be carried out, and it is likely that variants of all of the schemes discussed above will be deployed, depending upon local public network provider policies.

#### *4.5.1 Firewalls*

One unresolved issue with regard to any method of public network ATM connectivity is that of firewalls. Firewalls are the logical filters that multiprotocol routers implement today to control and restrict access to particular parts of networks. For instance, they might allow FTP access from the public network into a private network, but might preclude Telnet access. Such firewalls today are integral to network security, and while firewalls are implemented throughout networks, they are most common at connection points to the public network. Firewalls are implemented today in routers, which can process not only the layer 3 header information on packets, but can also look at higher layer fields—such as TCP port numbers, in order to determine the information needed to implement the firewalls.

It is not at all clear, however, just how, or whether, it might be possible to implement firewalls in an ATM environment. The problem is that once an ATM connection is set up, no intermediate devices generally interpret or process any of the information sent down that connection; doing so would make them not ATM switches but packet switches. Once a connection is set up between two end nodes, any data could be sent down that connection without visibility to network administration. While firewalls or other security mechanisms could be implemented in the end systems, it is not likely to be a practical solution for most end systems.

There have been proposals that firewall filtering within ATM networks should be done at connection set-up time and not on the transmitted data. Special information elements would be defined within the signaling messages to indicate the actual higher layer application binding that the connection wishes to make (for example, to telnet or to FTP). Then the intermediate switches could filter such connection set-ups based on higher layer information, source, and destination addresses, and so on.

ATM address filtering may be of particular use at the boundary between a private ATM network and a public or shared WAN network. Address filtering could be used at such points to allow connections to be made only to and from particular, trusted addresses (e.g. a remote site of the same administration, for instance), and preclude general connectivity. Such firewalls may be of particular use in conjunction with higher level controls (see Section 6.3), though all address based filtering techniques are also vulnerable to spoofing attacks.

While such techniques may have some utility, they are limited by the fact that little prevents an end system from lying about the use to which a connection would be used, since ATM connections generally terminate at lower levels within end system protocol stacks, and not at the actual applications[37]. Therefore, once a connection is set up, a node could send packets of any protocol type down the connection, and have these demultiplexed at the destination to any supported application, regardless of the identity of the application to which the connection was ostensibly set up to.

The only feasible solution to this problem appears to be to add cryptographic based authentication mechanisms to ATM signaling. Some preliminary work on such security mechanisms has been discussed at the ATM Forum, and elsewhere, but it is likely to be some time before they are fully specified or deployed. In the meantime, many network administrators continue to use routers as security firewalls, particularly at public network boundaries, even to connect two ATM networks to each other. While this has clear performance and service limitations, many network administrators often prefer such a solution to eliminating all existing firewall protections.

[36] An ATM end-system directly attached to the public network would presumably only have an E.164 number and not an NSAP format address. In such a case, a private network node would address this end-system by encoding the E.164 number within an NSAP format address. At the egress switch, this NSAP address would be algorithmically mapped into the corresponding E.164 number.

[37] Direct application interfacing precludes the support of existing protocols such as IP, which, in turn, precludes ATM nodes from communicating outside the ATM network.

## 4.6 Implementation Considerations

One of the concerns with the P-NNI Phase 1 protocol is that its complexity and scale mean that route calculation takes a considerable time, increasing the latency of connection set-up. Unlike current packet switches, which need to process every packet that is relayed, ATM switching systems only need to process a connection set up. Following connection set up, cells can be relayed without route processing. Unlike current link state protocols, however, which tend to generate semi-static routes that can be cached, the P-NNI protocols will likely require a significant proportion of lengthy on-demand route calculations due to the greater variability of its QoS-based routing metrics.

Given these considerations, it is likely that the ATM switching systems that use commercial processors for P-NNI calculation could only support call-set up rates of a few hundred connections per second, if that. Each of these could experience significant call set up latencies, perhaps exceeding hundreds of milliseconds, within large networks. These ATM routing latencies would be increased by any additional address resolutions that may need to be performed to map higher layer addresses to ATM addresses, as described in the following sections.

To reduce these set up latencies, which could significantly degrade perceived network responsiveness, many services operating over ATM have defined, or may define, default data paths that allow data to be transmitted pending the successful set up of direct data paths, or for the transmission of small amounts of data, the volume of which do not justify the cost and latency of a connection set-up. This characteristic will be noted in many of the higher layer services we describe next.

# 5.0 LAN EMULATION

The following sections will discuss the internetworking of existing protocols across ATM networks. Given the vast installed base of LANs and WANs today and the network and link layer protocols operating on these networks, a key to ATM success will be the ability to allow for interoperability between these technologies and ATM. Few users will tolerate the presence of islands of ATM without connectivity to the remainder of the enterprise network. The key to such connectivity is the use of the same network layer protocols, such as IP and IPX, on both existing networks and on ATM, since it is the function of the network layer to provide a uniform network view to higher level protocols and applications.

There are, however, two fundamentally different ways of running network layer protocols across an (overlay mode) ATM network. In one method, known as native mode operation, address resolution mechanisms are used to map network layer addresses directly into ATM addresses, and the network layer packets are then carried across the ATM network. Native mode protocols will be examined in the next section. The alternate method of carrying network layer packets across an ATM network is known as LAN emulation (LANE). The ATM Forum has recently completed a Phase 1 LAN Emulation specification [Forum7]. This section discusses the rationale for LAN emulation and describes the operation of the protocol.

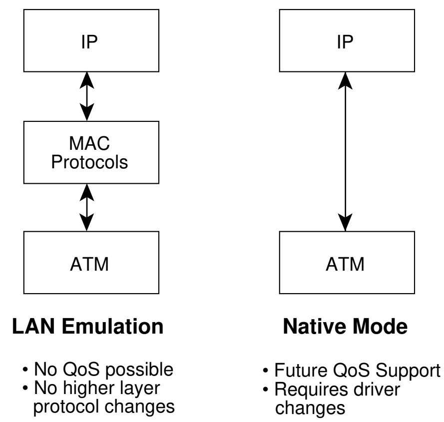



**Figure 18. Methods of ATM Internetworking**

As the name suggests, the function of the LANE protocol is to emulate a local area network on top of an ATM network. Specifically, the LANE protocol defines mechanisms for emulating either an IEEE 802.3 Ethernet or an 802.5 Token Ring LAN.[38]

What LAN emulation means is that the LANE protocol defines a service interface for higher layer (that is, network layer) protocols, which is identical to that of existing LANs, and that data sent across the ATM network are encapsulated in the appropriate LAN MAC packet[39] format. It does *not* mean that any attempt is made to emulate the actual media access control protocol of the specific LAN concerned (that is, CSMA/CD for Ethernet or token passing for 802.5).

In other words, the LANE protocols make an ATM network look and behave like an Ethernet or Token Ring LAN—albeit one operating much faster than a real such network.

[38] The current LANE protocol does not define a separate encapsulation for FDDI. FDDI packet must be mapped into either Ethernet or Token Ring emulated LANs, using existing translational bridging techniques. The two most prominent new LAN standards under consideration, Fast Ethernet (100Base-T) and 802.12 (100VG-AnyLAN) can both be mapped unchanged into either the Ethernet or Token Ring LANE formats and procedures, as appropriate, since they use the same packet formats.

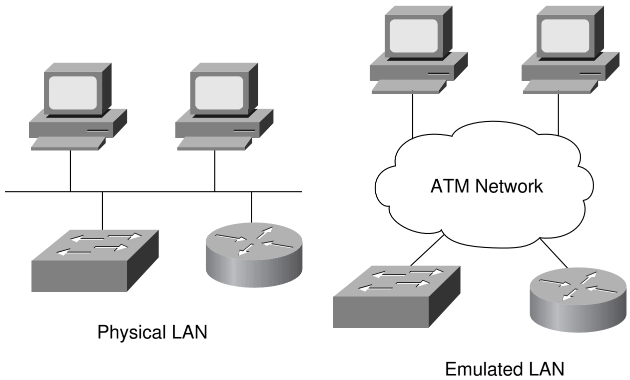



**Figure 19. Physical and Emulated LANs**

The rationale for doing this is that it requires no modifications to higher layer protocols to enable their operation over an ATM network. Since the LANE service presents the same service interface of existing MAC protocols to network layer drivers (for example, an NDIS- or ODI-like driver interface), no changes are required in those drivers. The intention is to accelerate the deployment of ATM, since considerable work remains to be done in fully defining native mode operation for the plethora of existing network layer protocols.

It is envisaged that the LANE protocol will be deployed in two types of ATM-attached equipment:

[39] The LANE protocol supports a range of maximum packet (MPDU) sizes, corresponding to maximum size Ethernet, and 4 Mbps and 16 Mbps Token Ring packets, and to the value of the default MPDU for IP over ATM (see Section 6.2). Typically the appropriate MPDU will be used depending upon what type of LAN is being emulated—and is supported on the LAN switches bridged to the ELAN. An ELAN with only native ATM hosts, however, may optionally use any of the available MPDU sizes, even if this does not correspond to the actual MPDU in a real LAN of the type being emulated. All LECs within a given ELAN must use the same MPDU size.

**a. ATM Network Interface Cards (NIC):** ATM NICs will implement the LANE protocol and interface to the ATM network, but will present the current LAN service interface to the higher level protocol drivers within the attached end system. The network layer protocols on the end system will continue to communicate as if they were on a known LAN, using known procedures. They will, however, be able to use the vastly greater bandwidth of ATM networks.

**b. Internetworking and LAN Switching Equipment:** The second class of network gear that will implement LANE will be ATM-attached LAN switches and routers. These devices, together with directly attached ATM hosts, equipped with ATM NICs, will be used to provide a *virtual LAN* service, where ports on the LAN switches will be assigned to particular virtual LANs, independent of physical location [Cisco]. LAN emulation is a particularly good fit to the first generation of LAN switches that effectively act as fast multiport bridges, since LANE is essentially a protocol for bridging across ATM. Internetworking equipment, such as routers, will also implement LANE to allow for virtual LAN internetworking, as will be discussed later.

Note that the LANE protocol does not directly impact ATM switches. LANE, as with most of the other ATM internetworking protocols we will discuss later in this paper, builds upon the overlay model. As such, the LANE protocols operate transparently over and through ATM switches, using only standard ATM signaling procedures. ATM switches may well be used as convenient platforms upon which to implement some of the LANE server components, which we discuss below, but this is independent of the cell relay operation of the ATM switches themselves. This logical decoupling is one of the great advantages of the overlay model, since they allow ATM switch designs to proceed independently of the operation of overlying internetworking protocols, and vice versa.

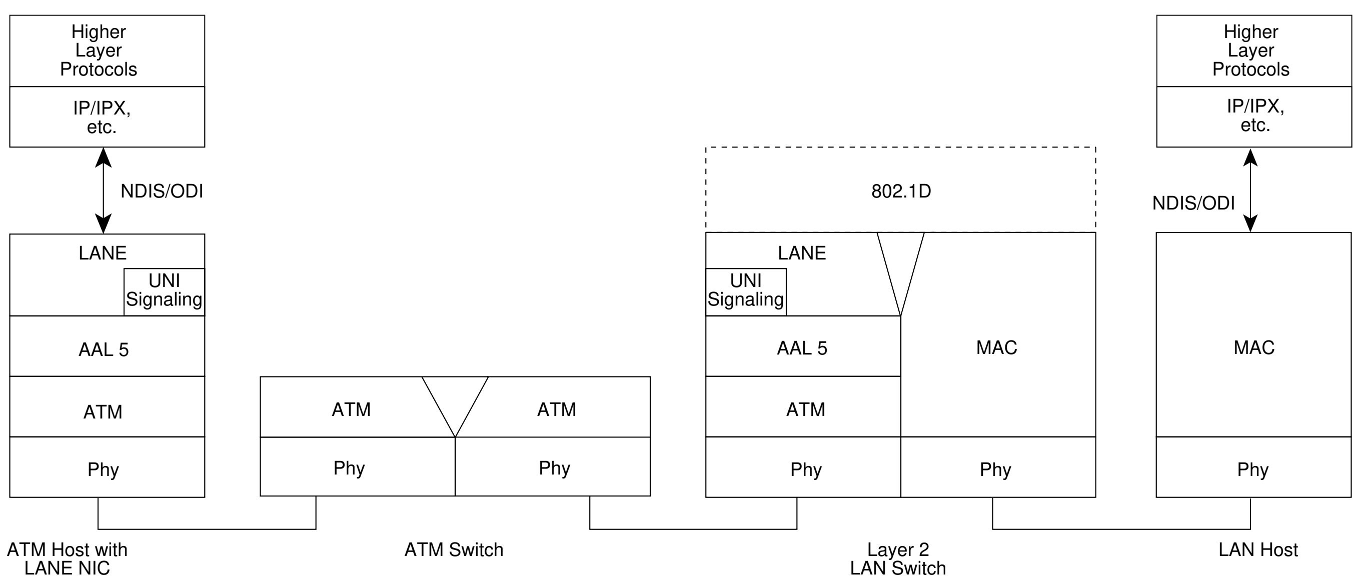



**Figure 20. LANE Protocol Architecture**

The basic function of the LANE protocol is to resolve MAC addresses into ATM addresses. By doing so, it actually implements a protocol for MAC bridging on ATM, hence the close fit with current LAN switches. The goal of LANE is to perform such address mappings so that LANE end systems can set up direct connections between themselves and forward data. The element that adds significant complexity to LANE, however, is supporting LAN switches—that is, LAN bridges. The function of a LAN bridge, as defined in [ISO] and [IEEE], is to shield LAN segments from each other. While bridges learn about MAC addresses on the LAN segments to which they are connected, such information is not propagated. How LANE resolves this problem will be discussed shortly.

## 5.1 LANE Components and Connection Types

The LANE protocol defines the operation of a single emulated LAN (ELAN). Multiple ELANs may coexist simultaneously on a single ATM network since ATM connections do not "collide." A single ELAN emulates either Ethernet or Token Ring, and consists of the following entities:

- ***LAN Emulation Client (LEC):*** A LEC is the entity in an end system that performs data forwarding, address resolution, and other control functions for a single end-system within a single ELAN. A LEC also provides a standard LAN service interface to any higher layer entity that interfaces to the LEC. An ATM NIC or LAN switch interfacing to an ELAN supports a single LEC for each ELAN to which they are connected. An end-system that connects to multiple ELANs (perhaps over the same UNI) will have one LEC per ELAN.

  Each LEC is identified by a unique ATM address, and is associated with one or more MAC addresses reachable through that ATM address. In the case of an ATM NIC, for instance, the LEC may be associated with only a single MAC address, while in the case of a LAN switch, the LEC would be associated with all the MAC addresses reachable through the ports of that LAN switch which are assigned to the particular ELAN. Note that in the latter case that this set of addresses may change, both as MAC nodes come up and down, and as particular paths are reconfigured by logical or physical changes in the LAN network topology (e.g. through the use of a spanning tree protocol, for instance).

Note that while the current LANE specification defines two types of emulated LANs, one for Ethernet, and one for Token Ring, it does not permit direct connectivity between a LEC that implements an Ethernet ELAN and one that implements a Token Ring ELAN. In other words, LANE does not attempt to solve the mixed media bridging problem, which is particularly intractable for Ethernet-to-Token Ring interconnection. Two such ELANs can only be interconnected through an ATM router that acts as a client on each ELAN, as discussed below.

- ***LAN Emulation Server (LES):*** The LES implements the control function for a particular ELAN. There is only one logical LES per ELAN, and to belong to a particular ELAN means to have a control relationship with that ELAN's particular LES. Each LES is identified by a unique ATM address. The operation of the LES is described below.
- ***Broadcast and Unknown Server (BUS):*** The BUS is a multicast server (see Section 2.0) that is used to flood unknown destination address traffic and forward multicast and broadcast traffic to clients within a particular ELAN. Each LEC is associated with only a single BUS per ELAN, but there may be multiple BUSs within a particular ELAN that communicate and coordinate in some vendor-specific manner; this action is outside the scope of the Phase 1 LANE protocol. The BUS to which a LEC connects is identified by a unique ATM address. In the LES, this is associated with the broadcast MAC address ("all ones"), and this mapping is normally configured into the LES.
- ***LAN Emulation Configuration Server (LECS):*** The LECS is an entity that assigns individual LANE clients to particular ELANs by directing them to the LES that correspond to the ELAN. There is logically one LECS per administrative domain, and this serves all ELANs within that domain.

The LANE protocol does not specify where any of the server components described here should be located; any device or devices with ATM connectivity would suffice. For the purposes of reliability and performance, however, it is likely that most vendors will implement these server components on networking equipment, such as ATM switches or routers, rather than on a workstation or host. This also applies to all other ATM server components described in the remainder of this paper.

The LANE protocol specifies only the operation of the LAN Emulation User to Network Interface (LUNI) between a LEC and the network providing the LANE service. This may be contrasted with the "LAN Emulation NNI" (LNNI) interface, which operates between the server components within a single ELAN system. The Phase 1 LANE protocols specify only the LUNI operation; furthermore, the phase 1 LANE protocol does not allow for the standard support of multiple LESs or BUSs within an ELAN. Hence these components represent both single points of failure and potential bottlenecks. The interactions between each of the server components in the LANE Phase 1 protocol are currently left unspecified, and will be implemented in a proprietary manner by vendors.

The ATM Forum is currently working on a Phase 2 LANE protocol, which will specify LNNI protocols, so as to allow for redundant LESs and replicated BUSs [Alles2], in order to address concerns about these limitations. The LNNI protocols will specify open interfaces between the various LANE server entities—LES/LES, LES/LECS, and BUS/BUS—and will allow for hierarchies of BUSs for greater scalability[40] within ELANs. This work is not expected to be completed until 1996, however.

The Phase 1 LANE entities communicate with each other using a series of ATM connections. LECs maintain separate connections for data transmission and control traffic.

The control connections are as follows:

- *Configuration Direct VCC:* This is a bidirectional point-to-point VCC set up by the LEC to the LECS.
- *Control Direct VCC:* This is a bidirectional VCC set up by the LEC to the LES.
- *Control Distribute VCC:* This is a unidirectional VCC set up from the LES back to the LEC; this is typically a point-to-multipoint connection.

The data connections are as follows:

- *Data Direct VCC:* This is a bidirectional point-to-point VCC set up between two LECs that want to exchange data. Two LECs will typically use the same data direct VCC to carry all packets between them, rather than opening a new VCC for each MAC address pair between them, so as to conserve connection resources and connection set-up latency. Since LANE emulates existing LANs, including their lack of QoS support, data direct connections will typically be UBR or ABR connections, and will not offer any type of QoS guarantees.
- *Multicast Send VCC:* This is a bidirectional point-to-point VCC set up by the LEC to the BUS.
- *Multicast Forward VCC:* This is a unidirectional VCC set up to the LEC from the BUS, this is typically a point-to-multipoint connection, with each LEC as a leaf.

[40] Note, however, that the fundamental limit to the scalability of an ELAN is not the number of BUSs, but the fact that all broadcast and flood traffic must be sent to all LECs; in the case where the LEC is within a LAN switch, this limits the amount of such traffic to be much less than the speed of the associated LAN, such as 10 Mbps in the case of an Ethernet ELAN.

## 5.2 LANE Operation

The operation of a LANE system and of the components mentioned above will be described in this section through the various stages of operation of a LEC:

### *5.2.1 Initialization and Configuration*

Upon initialization (such as power up), the LEC must first obtain its own ATM address (typically, this will be through address registration). The LEC then sets up a configuration-direct connection to the LECS. To do this, the LEC must first find the location of the LECS by either: using a defined ILMI procedure to determine the LECS address; using a well-known LECS address; or using a well-known permanent connection to the LECS (VPI=0, VCI=17).

After finding the location of the LECS, the LEC will establish the configuration-direct VCC to the LECS. Once connected, a configuration protocol is used by the LECS to inform the

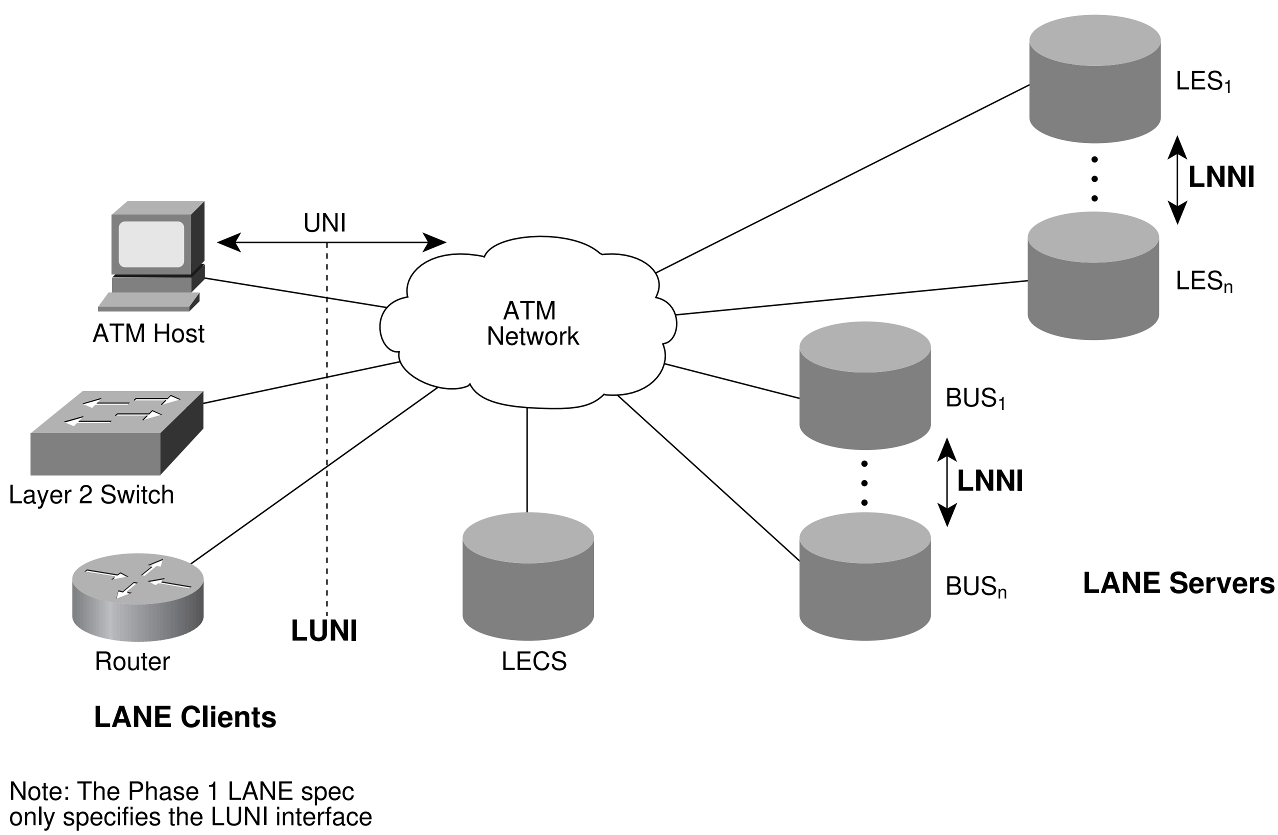



**Figure 21. LANE Protocol Interfaces**

LEC of the information it requires to connect into its target ELAN. This includes the ATM address of the LES, the type of LAN being emulated, maximum packet size on the ELAN, and the ELAN name (a text string for display purposes). The LECS is generally configured by network management with this information, which effectively indicates which virtual LAN (where a virtual LAN corresponds to an ELAN) to which the LEC belongs.

### *5.2.2 Joining and Registration*

Once the LEC obtains the LES address, it may optionally clear the configuration-direct VCC to the LECS; then it sets up the control-direct VCC to the LES. Once this is done, the LES assigns the LEC with a unique LEC Identifier (LECID). The LEC then registers its own MAC and ATM addresses with the LES. It may optionally also register any other MAC addresses[41] for which it is proxying—such as learned addresses in the case of spanning tree bridge.

[41] Generally, the support of a (source routed) Token Ring ELAN is the same as that of an Ethernet ELAN, except that all operations performed within an Ethernet ELAN on MAC addresses are correspondingly performed within the Token Ring ELAN on route descriptors; as such, the description of ELAN operation given here only considers the ELAN case. Refer to [Forum7] for a fuller description of Token Ring ELAN operation. More advanced issues such as that of ring number allocation within a network of bridged physical and emulated Token Ring segments is outside the scope of this paper.

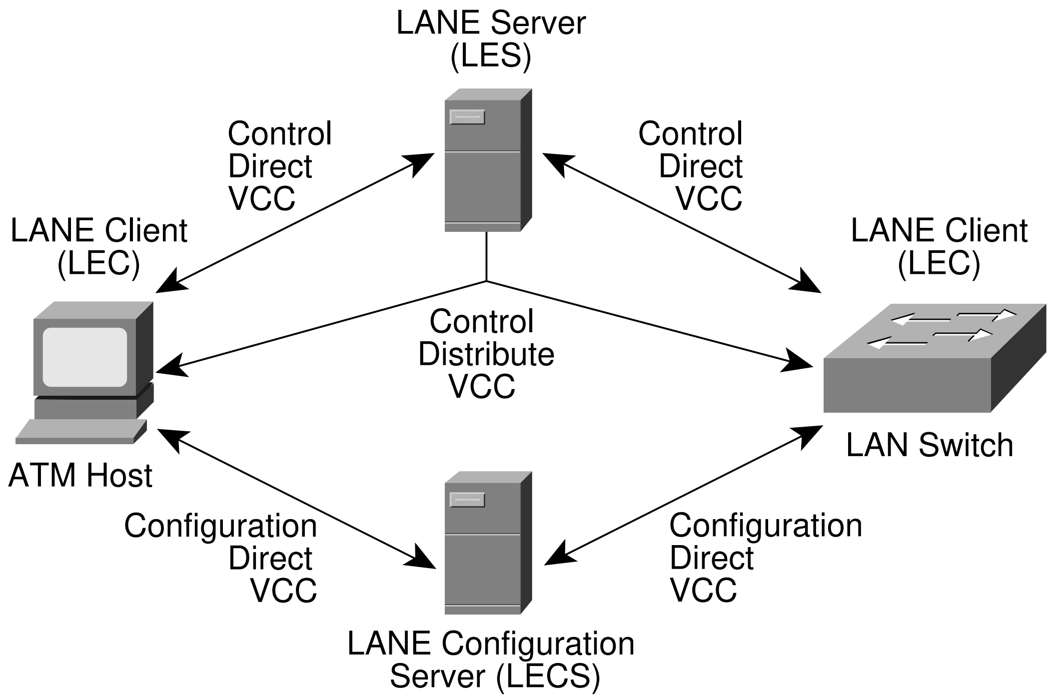



**Figure 22. LANE Control Connections**

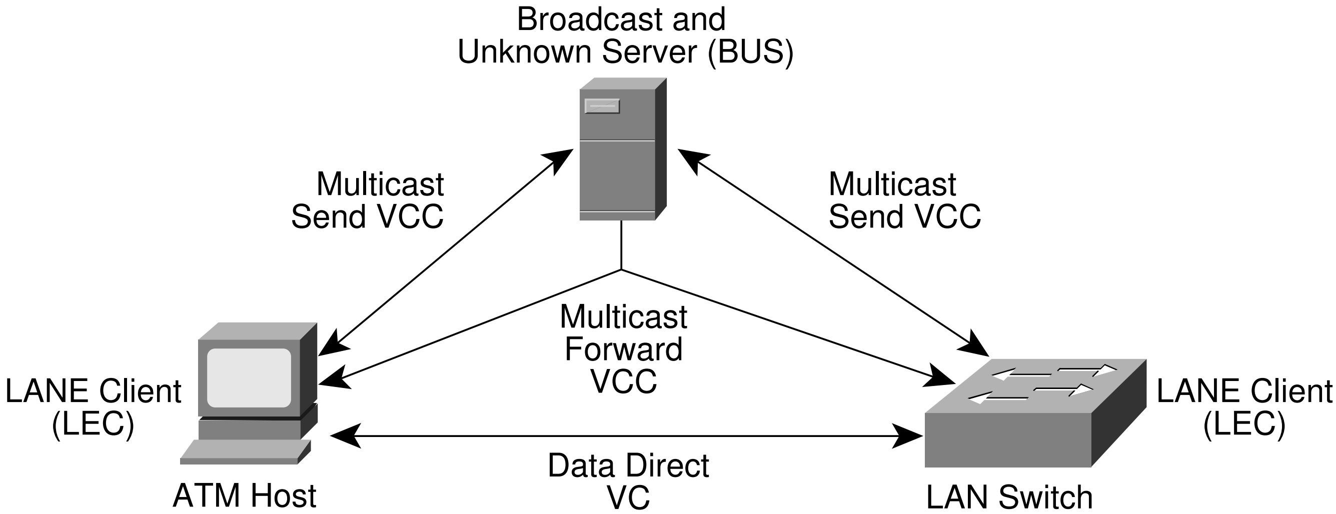



**Figure 23. LANE Data Connections**

The LES then sets up, back to the LEC, the control-distribute VCC. The control direct and distribute VCCs can then be used by the LEC for the LAN Emulation ARP (LE_ARP) procedure for requesting the ATM address that corresponds to a particular MAC address. To do this, the LEC formulates a LE-ARP and sends it to the LES. If the LES recognizes this mapping (because some LEC registered the relevant MAC address) it may choose to reply directly on the control-direct VCC. If not, it forwards the request on the control-distribute VCC to solicit a response from a LEC that knows the requested MAC address.

The typical reason why the LES would not know a mapping is because the address is "behind" a MAC bridge, and the bridge may not have registered the address[42]. An ATM NIC, on the other hand, would presumably only support one or a small number of MAC addresses, all of which could easily be registered. Typically, any MAC address not known to the LES would be found only in a LEC within a bridge, and not within a NIC, and only the LECs within such devices need necessarily receive re-directed LE-ARPs.

To accommodate this, LECs may register with the LES as a "proxy" node, indicating that it may proxy for other addresses and needs to obtain LE_ARPs. The LES then has the option of setting up the control distribute VCCs so that LE_ARPs are only sent to such proxy LECs—for example, through two point-to-multipoint connections connecting the LES to all of the proxy nodes, and one to all of the non-proxy nodes. This is not a requirement, however, and the LES may choose to simply distribute the LE_ARP to all LECs.

In any case, if a LEC can respond to a LE_ARP, because it is proxying for that address, it responds to the LES on the control direct VCC. The LES will then forward this response back either only to the requesting LEC, or, optionally, on the control distribute VCC to all LECs[43], so that all LECs can learn and cache the particular address mapping (and hence perhaps save future LE_ARPs).

To complete initialization, a LEC uses this LE_ARP mechanism to determine the ATM address of the BUS. It does this by sending an LE_ARP for the MAC broadcast address to the LES, which responds with the BUS's ATM address. The LEC then sets up the multicast send VCC to the BUS. The BUS, in turn, sets up the multicast forward VCC back to the LEC, typically by adding the LEC as a leaf to a point-to-multipoint connection. The LEC is now ready for data transfer.

### *5.2.3 Data Transfer*

During data transfer, a LEC either receives a network layer packet to transmit from a higher layer protocol (in the case of NIC) or receives a MAC packet to forward across a LAN port (in the case of a LAN switch[44]). In the first instance, the source LEC will not have the ATM address of the destination LEC through which the particular destination MAC address can be reached. In this case, the LEC first formulates and sends to the LES a LE_ARP response.

While waiting for a response from this LE_ARP, the LEC also forwards the packet to the BUS, using a defined encapsulation. The BUS will, in turn, flood the packet to all LECs. This must be done because, in the case of a passive device behind a LAN switch, no LEC may know where the MAC address is located[45]. Additionally, resolving a LE_ARP may take some time and many network protocols are intolerant of either loss (if the LEC chose to discard the packet while awaiting the LE_ARP response) or latency (if the LEC chose to buffer the packet). In this mode, the BUS provides the analog of the flooding procedure used by spanning tree bridges for unknown destination packets, hence its name.

If an LE_ARP response is received, the LEC then sets up a data-direct VCC to the destination node, and uses this for data transfer rather than the BUS path. Before it can do this, however, the LEC may need to use the LANE "flush" procedure to ensure that all packets previously sent to the BUS were delivered to the destination prior to the use of the data direct VCC. In this mechanism, a control cell is sent down the first transmission path, following the last packet; not until the receipt of this flush cell is acknowledged by the destination is the second path used to send packets. This mechanism is the guaranteed way to meet current LAN standards that require LAN bridges to strictly preserve frame ordering.

If a data direct connection already exists to the LEC (in the same ELAN) through which a particular MAC address is reachable, the source LEC may optionally choose to re-use this same data direct connection, so as to conserve connection resources and save on the connection set-up latency.

[42] Since bridge tables may have thousands of entries that are continuously being learned, aged out, moved, and so on, a bridge typically would only register static entries.

[43] If the LES maintains two control distribute VCCs, one to proxy nodes, and one to non-proxy nodes, it would then need to replicate such responses before forwarding onto each connection.

[44] A LAN switch only needs to invoke the LANE procedures if either its MAC bridging tables indicate that the destination is not local to the switch, or if it does not know where to send the packet and hence must flood it. Most LAN switches will locally switch traffic between local ports.

[45] As with a learning bridge, a LEC will learn the location of the device if and when it responds to the flooded packet.

If a response is not received to a LE_ARP, the LEC will continue to send packets to the BUS, but will regularly re-send LE_ARPs until a response is received. Typically once a packet is flooded through the BUS, and the destination responds to the source, some LEC will learn the location of the destination, and then respond to a subsequent LE_ARP.

A LEC will locally cache any MAC address to ATM addresses mapping it learns through a LE_ARP. If and when the LEC receives for transmission another packet to that same MAC address, it will then consult that local cache table and use the cached mapping, rather than sending out another LE_ARP. Such cached entries are normally aged out over a configurable time period (typically 5 minutes). Similarly, data direct connections will be cleared if the connection remains inactive over a configurable period (typically 20 minutes). There are circumstances, however, when cached ARP information may be aged out at a much faster rate—this is discussed below.

The BUS is also used by LECs for broadcast and multicast packets. Such packets are forwarded to the BUS, which then redirects them to all LECs. This implies that the source LEC may receive a copy of its own broadcast or multicast packet. Since some LAN protocols cannot tolerate such a condition, the LANE packet encapsulation requires that all MAC packets be prefixed with the LECID. LECs can then filter on this field for all frames that are received from the BUS to ensure that it never receives its own frames.

### 5.3 LANE and Spanning Tree

The LANE protocol was developed recognizing that typically a spanning tree protocol ([IEEE], [OSI]) would be run within each ELAN, and the set of external networks (such as LAN switch LAN ports) bridged to the ELAN, so as to preclude loops within the network. This is particularly important in the case where LAN switches are interconnected by an ELAN, while the external networks connected to the LAN switches may themselves be interconnected by external bridges[46]. LECs within LAN switches will exchange spanning tree bridge packets (BPDU) between themselves, multicasting the packets through the BUS (hosts will ignore these bridge packets).

If a LAN switch detects a loop, through its spanning tree protocol, then it will turn off either one of the external ports, or the ELAN port, as appropriate, so as to break the loop; in general, since the spanning tree protocol weighs links by their bandwidth, the protocol will tend to favor the LANE port, and will first turn off external ports. Note that even where the ELAN port is turned off, however, full connectivity will still—by definition—be possible through the external bridged path.

The action of the spanning tree protocol, within a complex multi-path bridged network, will typically cause the LECs through which particular external MAC addresses are reachable (that is, through particular LAN switch ports) to change dynamically. As noted above, however, LECs typically will cache ARP information for relatively lengthy periods, hence there is a danger that LECs may end up using stale information for excessive periods until the ARP table entries are aged out—in the meantime, information may be sent to a "black hole," since the LEC to which the data direct connection was originally set up may no longer have any direct connectivity with the intended recipient. Note that this problem is exacerbated by the multiplexing of many data flows (that is, MAC addresses) onto the same data-direct connection.

In order to allow for faster convergence, the LANE protocol supports LE-Topology-Request messages. These are generated by any LEC implementing the spanning tree protocol (typically a LAN switch) upon the detection of any topology change that triggers a BPDU configuration update message. The LE-Topology-Request is sent by the LEC to the LES, which in turn distributes it to all other LECs. Upon receipt of such a message, all LECs will reduce the aging period on their cached ARP information. This, in turn, will age out the cached information faster, causing the LECs to more quickly refresh the ARP information through LE-ARPs that will, in turn, generate more up-to-date reachability information.

Note that LECs will not tear down existing data-direct connections upon the detection of a network reconfiguration. Rather, if and when cached LE-ARP information is refreshed, the data-direct connection may fall idle, if no desired MAC addresses are any longer reachable through the connection. Eventually, then, the LEC will time out the idle connection and clear it.

The LANE protocol also allows for LECs to generate an unsolicited "LE-NARP" message when the LEC detects, through local means, that a particular MAC address, which was once thought to be remote from that LEC, is now reachable through the LEC. Such messages are sent to the LES, which redistributes it to all other LECs; these, in turn, may use such indications to update their address caches. Such messages may speed convergence in some particular conditions, but their use and utility is somewhat controversial.

[46] This description assumes that each LAN port on a layer 2 switch is associated with one, and only one, ELAN, and hence only with one spanning tree protocol instantiation, associated with that ELAN, and any other LAN segments bridged to that ELAN. The case where multiple ELANs—hence possibly, multiple spanning tree protocol instantiations—are associated with a single LAN port—for instance, because one ELAN may be defined per protocol operating across that LAN port—is much more complex, and is outside the scope of this paper.

## 5.4 Intelligent BUS

The description above is only an overview of the operation of the LANE protocol. Many aspects of LANE are open to vendor differentiation—for instance, whether or not the LES chooses to respond to LE_ARPs. One controversial option is known as the intelligent BUS. An intelligent BUS is one that obtains knowledge of the whereabouts of MAC addresses through some means (such as through sharing of the LES registration table). In such a case, the BUS may not flood an unknown destination packet, but may forward it directly to the appropriate LEC across the multicast send VCC (this is why this VCC is bidirectional).

In this mode, an intelligent BUS effectively operates as a connectionless server; in the extreme, this would preclude the need for data-direct VCCs at all, since a “minimal” LEC could send all packets to the BUS for forwarding, and would avoid the need to support some of the more complex elements of the LANE protocol. This is not a desirable mode of operation, however, since the BUS can very quickly become a bottleneck. A minimal LEC used with a normal BUS could also quickly flood the network with packets, since it would not attempt to set up data-direct VCCs at all. To avoid these problems, the LANE protocol, while allowing for intelligent BUSs, does require all LECs to set up data-direct VCCs whenever possible, and also restricts the number of flood (unicast) packets that can be sent to the BUS in any given period.

## 5.5 LANE and Virtual LANs

LANE is used by vendors to provide a virtual LAN service on ATM backbones. Such virtual LANs are implemented on *switched internetworks* that consist of a combination of (bridging) LAN switches, ATM end systems (typically servers, using ATM NICs), and routers with ATM interfaces (“ATM routers”) all connected to an ELAN. The ELAN looks like a normal LAN in every respect except for bandwidth as far as either end systems attached to the LAN ports on the LAN switches, or the higher layer protocols operating within the ATM hosts or routers are concerned. Their operation does not differ in any manner. From the viewpoint of network administration, however, constructing a virtual LAN out of LANE has significant advantages.

In particular, through network management and the use of such mechanisms as the LECS, the network administrator can set up multiple different ELANs across a single ATM backbone and then assign LAN switch ports or ATM hosts[47] to the different ELANs, independent of the physical location of the devices. This is unlike current networks where the physical location of a device generally dictates the physical LAN segment to which the device can be connected. Today, physically co-located users must be placed on the same LAN. This was acceptable in the past where organizational work flows generally reflected actual, physically collocated work groups. Today, however, as organizations re-engineer to flatten organizational hierarchy and reduce compartmentalization, most work flows reflect ad hoc, cross-functional project teams. In such cases, the work flow spans the enterprise, independent of people’s physical location.

Virtual LANs build upon LANE and give network administrators the ability to easily and dynamically create and reconfigure virtual networks, tracking the formation and change of ad hoc project teams. In other words, virtual LANs allow network administrators to adapt the network to organizational work flows, rather than constraining the organization around the physical network, as they must currently do.

Allowing centralized logical reconfiguration of end systems, without requiring physical network reconfiguration, can also help reduce the costs of “moves, add and changes,” which constitute a significant proportion of network support costs, given the increasing dynamism of work groups. For instance, a node could be physically moved, but still retain membership of the same VLAN it used to belong to before, without ending up on the “wrong” side of a network firewall. Conversely, a node could be made a member of a new virtual LAN through a change in its ELAN membership, without requiring any physical network changes. In the latter case, depending upon the protocol, the node may need to change its network layer (e.g. IP) address, though other protocols, such as DHCP, can also help automate this process.

These powerful benefits of virtual LANs will likely spur the widespread deployment of LANE. However, the limitations of LANE must also be understood. As noted earlier, LANE is essentially a LAN bridging standard. As such, much as with physically bridged LANs, ELANs are susceptible to such phenomena as broadcast storms. These factors tend to limit the applicability of ELANs to small workgroups, where virtual LANs also offer the most powerful advantages. This means that a large enterprise network is likely to support a large number of virtual LANs (ELANs).

This implies immediately the need for a means to interconnect all of these ELANs—both to themselves (to interconnect an Ethernet and Token Ring ELAN, for instance), and to existing LAN and WAN networks. The easiest and most common way in which this will be done is through ATM routers. Much as conventional routers connect together physical LANs today, ATM routers will interconnect virtual LANs. They will do so by supporting high performance native ATM interfaces and

[47] Hosts that need to members of multiple virtual LANs (for instance, because they may be servers supporting common applications) may support multiple LECs on their ATM NICs, and hence act as multi-homed hosts on several ELANs. Typically a port on a LAN switch, however, would only be assigned to a single ELAN.

by implementing LANE so that the router supports multiple LECs on each physical native ATM interface, one for each ELAN it interconnects.

End systems on the ELAN will recognize, using local, protocol specific means, when a desired destination is outside the node's local virtual LAN (ELAN). In the case of a node implementing IP, for instance, typically each virtual LAN will be associated with a unique IP subnet number. Hence a node on the ELAN will perform a "mask and match" on a destination node's IP address and determine that the node is not on the source node's own subnet (hence ELAN). The node will then forward the packet, using the LANE protocols, to its default router; this router will also be a member of the ELAN, and will hence be reachable across the ELAN. If the destination node is on the same subnet—hence virtual LAN—direct connectivity will be possible, of course, without requiring any router involvement.

Once the packet reaches the router, it will then consult its own next hop tables to determine where to forward the packet. If these tables indicate that the destination node is reachable through another ELAN of which the router is a member, the router will then forward the packet into that ELAN—possibly over the same physical interface over which the packet was first received, but now into a new ELAN. Note that the higher layer protocol processing within the router is unaffected by the fact that the router is now dealing with emulated and not physical LANs. This is another example of the value of LANE in hiding the complexities of the ATM network.

One obvious limitation of this approach, however, is that the ATM router may well eventually become a bottleneck, since all inter-ELAN traffic must traverse the router. LANE itself, has another limitation. By definition, the function of LANE is to hide the properties of ATM from higher layer protocols. This is good, particularly in the short to medium term, since it precludes the need for any changes to these protocols. On the other hand, LANE also precludes these protocols from ever using the unique benefits of ATM, and specifically, its QoS guarantees. LANE is defined to use only UBR and ABR connections, since it is these that map best to the connectionless nature of MAC protocols.

In the future, higher layer protocols may indeed wish to use these properties (that is, use VBR connections). This topic will be discussed at the end of this paper when other means beyond LANE of supporting virtual LANs are discussed.

## 6.0 Native Mode Protocols

This section discusses the alternate manner of carrying network layer protocols across an ATM network—not through LANE, but with native mode protocols. While all current network layer protocols could be enhanced to run directly across an ATM network, currently, the only protocols for which extensive work has been done is IP. Novell has publicly discussed a protocol known as Connection Oriented IPX (CO-IPX), which will adapt IPX specifically for ATM networks, and will add QoS support, but full development of this protocol is not expected for some time [Bottorff]. This section, therefore, primarily discusses the work of various working groups within the Internet Engineering Task Force

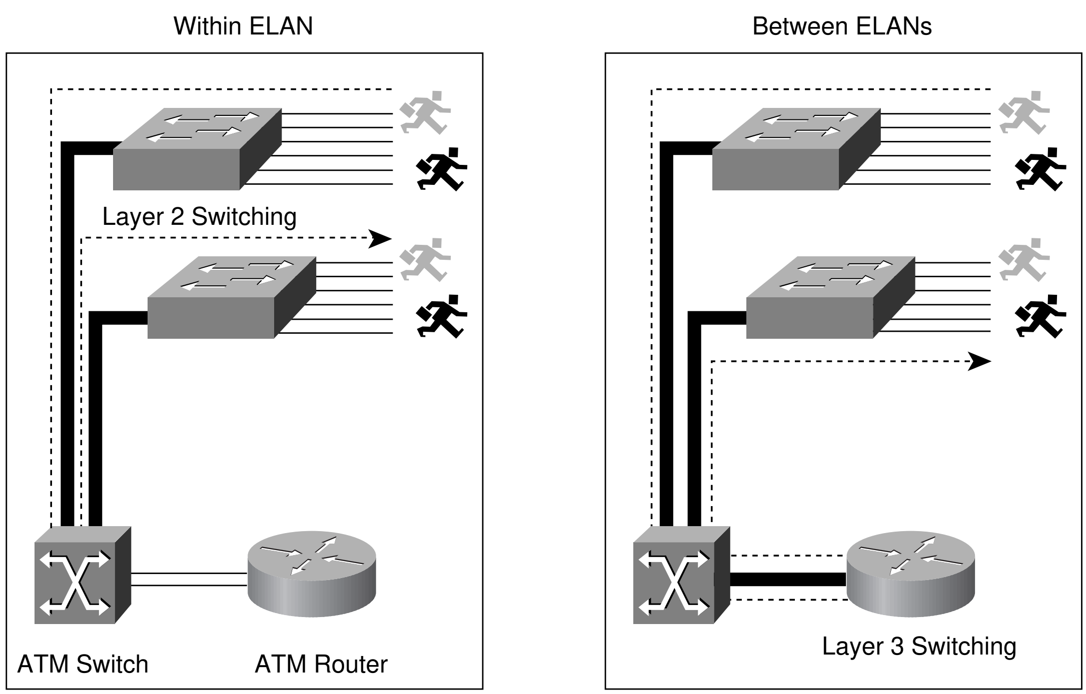



**Figure 24. Internetworking Between ELANs**

(IETF) on running IP over ATM. The next section discusses the current work being done at the ATM Forum on developing a true multiprotocol over ATM standard.

## 6.1 Integrated Services

The main rationale for using a native mode protocol, as opposed to LANE, was hinted at in the conclusion of the previous section. LANE deliberately hides ATM so any network layer protocol that operates over ATM cannot gain access to the QoS properties of ATM and must, therefore, use UBR or ABR connections only. At the moment, this is not a major restriction because all current network protocols were developed for use over existing LAN and WAN technologies, none of which can deliver a guaranteed QoS. Consequently, no existing network layer protocol can request a specific QoS from the network[48], or deliver such to a higher layer protocol or application. Hence, in turn, most network applications today do not expect to receive, and do not request, any guaranteed QoS from the underlying network protocol.

At best, therefore, all current network layer protocols today expect and deliver only a "best effort" service—precisely the type of service that the ABR service was designed to offer. Much as LANE adapts ATM's connection-oriented nature to offer the same type of connectionless service that is expected by network layer protocols, so ABR hides the guaranteed QoS features of ATM to offer the best effort service expected by these protocols. As such, ABR and LANE perfectly complement each other.

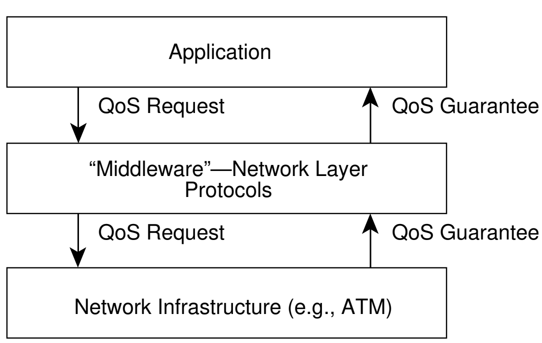



**Figure 25. Application QoS Support Through the Network Layer**

[48] IP has long had optional support for Type of Service (TOS) indications within the IP header, which could theoretically be used to provide a rudimentary form of QOS support. In practice, however, almost no end system or intermediate system IP implementations have any support for TOS since they cannot be mapped into any common underlying networking technology. Few, if any, IP routing protocols use the TOS bits, for instance, and no applications set them.

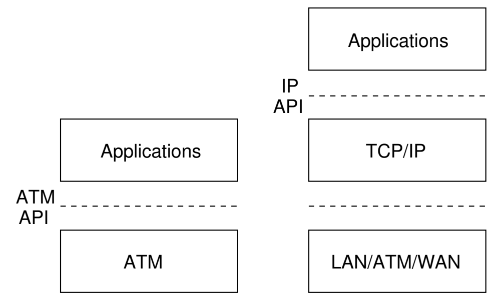



**Figure 26. Native and Conventional Applications**

In the future, however, this situation is unlikely to endure. In the first instance, as ATM networks proliferate, it is likely that demand will grow to utilize their QoS benefits, since this is one of ATM's major selling points. Independent of ATM, moreover, considerable work is being done on building a networking infrastructure capable of supporting a wholly new class of multimedia applications that combine voice, video, image, and data traffic. To support such applications, QoS guarantees are required from the network (for example, to minimize jitter and latency for interactive voice applications).

One way in which such applications could be built is by running the applications or transport protocols directly across ATM, or over a minimal network layer. This is the approach taken in such proposed protocols as TCP and UDP over Lightweight IP (TULIP) and TCP and UDP over Nonexistent IP (TUNIP) [Cole]. The ATM Forum is also working on developing models for an API for direct ATM access within operating systems.

Interest in such minimal protocol stacks was originally sparked by speculation that existing protocol stacks, such as TCP/IP, could not scale to high bandwidth networks. Hence this has caused some to suggest that it would be better not to run network layer protocols such as IP over ATM, but that such protocols should be bypassed in favor of running applications directly over ATM. This reasoning is flawed, for a number of reasons. In particular, the performance concerns apply mostly not to network layer protocols such as IP—which being connectionless have minimal performance impact, given an efficient implementation—but on the much more complex, state-based transport layer protocols such as TCP.

More recent analysis and implementations, however, have shown that efficient and optimized designs of such stacks can indeed operate at the very high data rates of such high speed networks as ATM. As such, much of the original rationale for minimal stacks no longer apply. Refer to [Partridge3] and [Borman] for more details on high speed TCP/IP implementations. Others have expressed concern about "overhead" of the headers of such protocols, but these seem misplaced given the increasing bandwidth of networks such as ATM, and the value of using such header information to facilitate internetworking.

Indeed, the major drawback of minimal stack approaches is that they limit applications which utilize them purely to ATM networks. This may be appropriate in the future, if and when ATM deployment, particularly to the desktop, becomes ubiquitous. Today, however, and in the medium term, when other networking technologies are, and will remain, much more common, such an approach would greatly constrain the deployment options—and the commercial viability—of the applications.

It is sometimes forgotten that one of the principal functions of network layer protocols is to offer universal connectivity, and a uniform service interface, to higher layer protocols—in particular, to transport layer protocols—independent of the nature of the underlying physical network. Correspondingly, the function of transport layer protocols is to provide session control services (e.g. reliability) to applications, so that these can be built without being tied to a particular network type. Unless applications run over common network and transport protocols, interoperability between two applications running on two different networks (e.g. ATM and a conventional network), would be difficult, if not impossible[49].

Hence, other than for a small class of applications that can only ever run on ATM (e.g. because they require more bandwidth than available from any other technology—for instance, studio quality video processing), most multimedia applications will continue to be built upon enhancements of current network layer protocols, and will be deployed on a wide variety of high speed networking technologies.

In the specific case of IP, the IETF has developed the notion of an Integrated Services Internet [Braden1]. This envisages a set of enhancements to IP to allow it to support integrated or multimedia services. These enhancements include traffic management mechanisms that closely match the traffic management mechanisms of ATM. For instance, protocols such as the Resource Reservation Protocol (RSVP) are being defined to allow for resource reservation across an IP network, much as ATM signaling allows this within ATM networks [Zhang].

[49] Internetworking between a native ATM application and an application on a conventional protocol stack may be possible through the use of an application gateway. Gateway functions, however, generally require complex configurations and mapping, hence compromising performance and ease of use, and often cannot provide, in any case, totally transparent service mappings. For such reasons, gateways are generally not considered acceptable for general use.

RSVP is a control protocol, much like ICMP, that will be used by applications within IP end-systems to indicate to nodes transmitting to them[50] the nature (such as bandwidth, jitter, maximum burstiness, and so on) of the packet streams that they wish to receive. Intermediate systems, along the path from the source to the destination IP end-systems, will also interpret RSVP control packets in order to perform admission control (analogous to ATM CAC) and allocate the resources required to support the requested traffic flows. Such systems will maintain "soft-state" about such traffic flows, much as ATM switches maintain connection state, and will perform packet level traffic shaping, scheduling, and so on, in the same manner that ATM switches groom cell streams so as to provide the guaranteed QoS. RSVP can hence be thought of as providing very much the same traffic contract specification functions with respect to packet level traffic flows that ATM UNI and NNI signaling play with respect to cell flows.

RSVP is fundamentally built upon a multicast paradigm, and routes traffic flows along source rooted point-to-multipoint paths (with unicast handled as a special case of multicast). New multicast protocols like Protocol Independent Multicast (PIM) [Deering2], and their associated unicast packet routing protocols, will hence be closely coupled with RSVP, much as VC routing protocols are closely coupled with UNI and NNI signaling.

Such protocols rely upon a flow specification [Partridge2] to characterize the expected traffic patterns for a stream of IP packets between two applications, which the network can process through packet-level policing, shaping, and scheduling mechanisms to deliver a requested QoS. In other words, a flow can be thought of as a layer 3 connection, since it identifies and characterizes a stream of packets between two or more nodes, even though the protocol remains ostensibly connectionless.

The IP Version 6 (IPv6) protocol[51], which the IETF is now developing as a replacement for the current IPv4 protocol, incorporates support for a flow ID within the packet header, which the network can use to identify flows, much as VPI/VCI are used to identify streams of ATM cells. Protocols like RSVP will be used to associate with each flow a *flowspec*

[50] One significant difference between RSVP and ATM signaling is that RSVP uses a receiver oriented model, where the receiving node indicates to the network and the transmitting node the nature of the traffic flow that the node is willing and able to receive, whereas in ATM, the transmitting node indicates to the receiving nodes and network the nature of the cell streams that it desires to transmit. The former model is more application oriented, while the latter is more network oriented. Methods of reconciling these two differing paradigms are currently under study.

[51] IPv6 was formally known as the IP Next Generation (IPng) protocol.

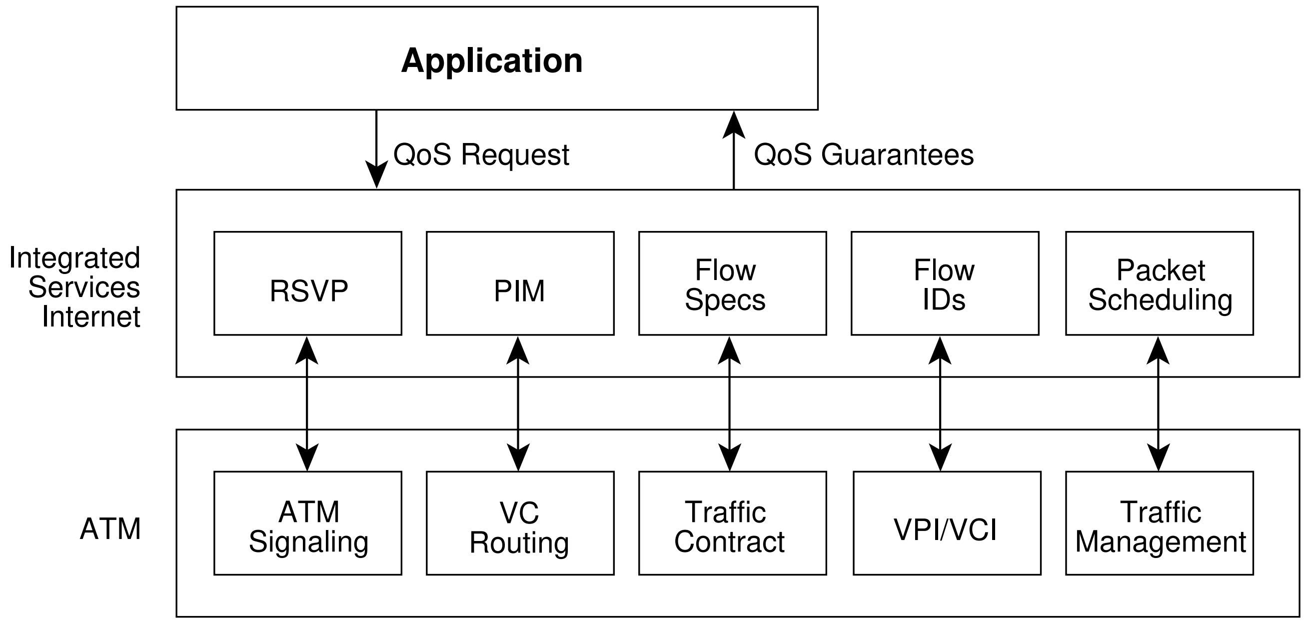



**Figure 27. Mapping of the Integrated Services Internet into ATM**

that characterizes the traffic parameters of the flow, much as the ATM traffic contract is associated with an ATM connection.

It is certain that IPv6 ([Bradner], [Hinden]) will incorporate full support for integrated services through the use of such mechanisms and the definition of protocols like RSVP. Such support might also be extended to the current IPv4 protocol. It is likely that IPv6, and other protocol components of the Integrated Service Internet, will be fully standardized by the end of 1995, and components may be deployed even earlier.

The IETF is also in the process of developing a new transport protocol, the Real-Time Transport Protocol (RTP) [Schulzrinne]. RTP is designed to provide end-to-end network transport functions for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services, and builds upon protocols like RSVP for resource reservation, and upon transport technologies like ATM for QoS guarantees. The services provided by RTP to real time applications include payload type identification, sequence numbering, timestamping and delivery monitoring. Closely tied to the RTP protocol functions is the RTP control protocol (RTCP), to monitor the quality of service and to convey information about the participants in an on-going session. Hence RTP can be used for such applications as multipoint conferencing, building upon the other protocol services of the Integrated Service Internet.

When such protocols are widely deployed and applications are developed to use them, there will certainly be a demand to run such protocols in native mode over ATM. It would be pointless to obtain QoS support from the network layer, only to have LANE preclude that support from being mapped to their equivalents in the ATM network. There is clearly a very clear and natural mapping between the concepts and mechanisms of the Integrated Services Internet and ATM (flow IDs and flowspecs to ATM connections and traffic contracts, respectively, and so on).

Hence the Integrated Services Internet can be thought of as eventually providing the packet level control infrastructure for the physical network infrastructure of ATM, where the former provides application services and the latter realizes the requested QoS guarantees. In this way, the true value of ATM can be exploited, while preserving a network independent service infrastructure for application portability. In order to realize the vision, however, there must be native mode protocol support over ATM.

## 6.2 IP Over ATM

To prepare for this need, the IETF's IP-Over-ATM working group has been working for some time to develop a protocol for IP transport over ATM. This protocol will be described in this section. The transport of any network layer protocol over an overlay mode ATM network involves two aspects: packet encapsulation and address resolution. Both of these aspects have been tackled by the IETF, and are described below:

### *6.2.1 Packet Encapsulation*

The IETF worked first on defining a method for transporting multiple types of network or link layer packets across an ATM (AAL 5) connection and also for multiplexing multiple packet types on the same connection. As with LANE, there is value to reusing the same connection for all data transfers between two nodes since this conserves the (typically scarce) connection resource space, and saves on connection setup latency, after the first connection set-up. This is only possible, however, as long as only UBR or ABR connections are used—if the network layer requires QoS guarantees then every distinct flow will typically require its own (VBR) connection.

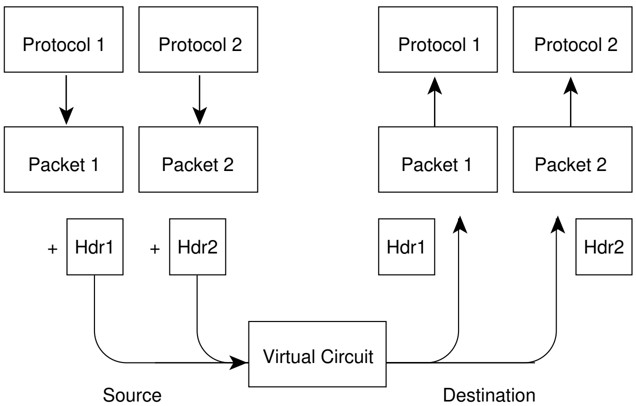



**Figure 28. Packet Encapsulation and Connection Re-use**

In order to allow connection re-use, there must be a means for a node that receives a network layer packet across an ATM connection to know what kind of packet has been received, and to what application or higher level entity to pass the packet to; hence, the packet must be prefixed with a multiplexing field. Two methods for doing this[52] are defined in RFC 1483 [Heinanen1]:

- *LLC/SNAP Encapsulation.* In this method, multiple protocol types can be carried across a single connection with the type of encapsulated packet identified by a standard LLC/SNAP header. A further implication of LLC/SNAP encapsulation, however, is that all connections using such encapsulations terminate at the LLC layer within the end-systems, since it is here that the packet multiplexing occurs.
- *VC Multiplexing.* In the VC muxing method, only a single protocol is carried across an ATM connection, with the type of protocol implicitly identified at connection set-up. As a result, no multiplexing or packet type field is required or carried within the packet, though the encapsulated packet may be prefixed with a pad field. The type of encapsulation used by LANE for data packets is actually a form of VC muxing.

[52] Communication between two devices will require either that two devices agree on a common form of encapsulation (e.g. using indications in signaling messages), or that an internetworking device (e.g. a router) be used to convert between the two forms of encapsulation.

The VC muxing encapsulation may be used where direct application to application ATM connectivity, bypassing lower level protocols, is desired. As discussed earlier, however, such direct connectivity precludes the possibility of internetworking with nodes outside the ATM network.

The LLC/SNAP encapsulation is the most common encapsulation used in the IP over ATM protocols described in the following section. The ITU-T has also recently adopted this as the default encapsulation for multiprotocol transport over ATM, as has the ATM Forum's Multiprotocol over ATM Group, which is discussed below.

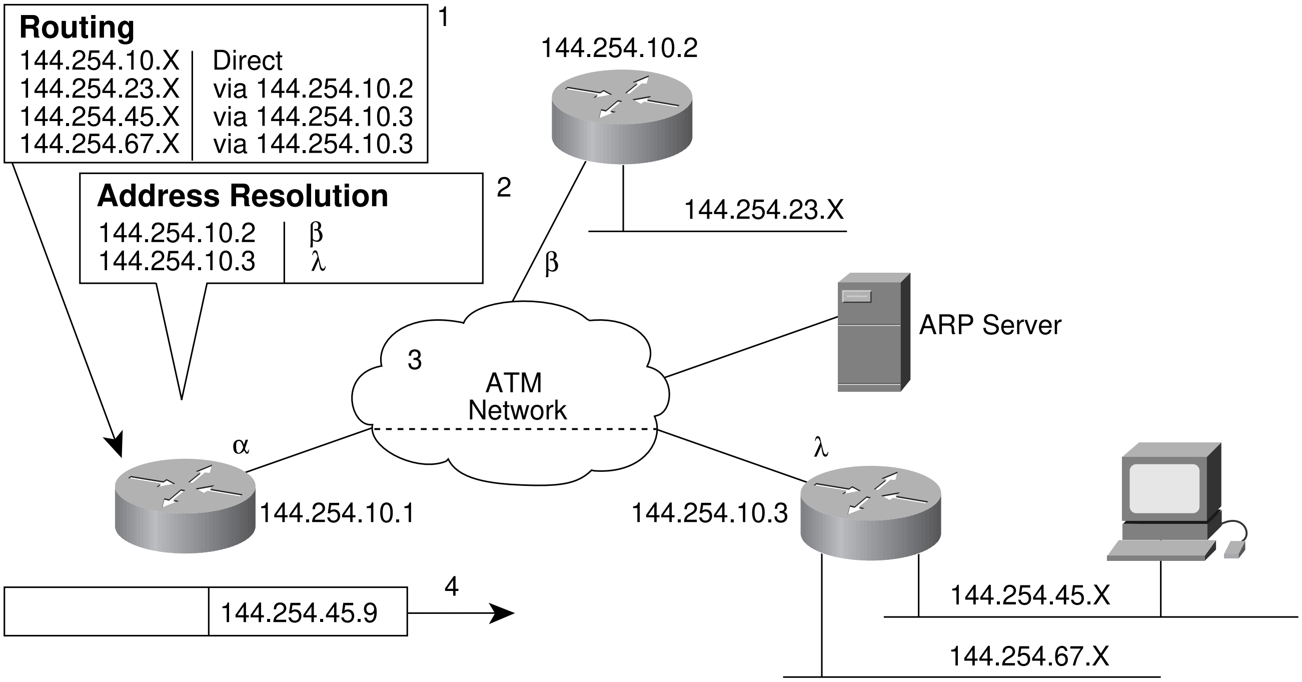
In related work, the IP over ATM group has also defined a standard for a maximum transfer unit (MTU) size over ATM [Atkinson]. This defines the default MTU as 9180 bytes to be aligned with the MTU size for IP over SMDS. It does, however, allow for negotiation of the MTU beyond this size, to the AAL 5 maximum of 64 Kilobytes, since important performance improvements can be gained by using larger packet sizes. This standard also mandates the use of IP Path MTU discovery [Mogul] by all nodes implementing IP over ATM to preclude the inefficiency of IP fragmentation.

### *6.2.2 Address Resolution*

In order to operate IP over ATM, a mechanism must be used to resolve IP addresses to their corresponding ATM addresses. For instance, consider the case of two routers connected across an ATM network. If one router receives a packet across a LAN interface, it will first check its next-hop table to determine through which port, and to what next-hop router, it should forward the packet. If this look-up indicates that the packet is to be sent across an ATM interface, the router will then need to consult an address resolution table to determine the ATM address of the destination next-hop router (the table could also be configured, of course, with the VPI/VCI value of a PVC connecting the two routers).

This address resolution table could be configured manually, but this is not a very scalable solution. The IP-Over-ATM working group has defined a protocol to support automatic address resolution of IP addresses in RFC 1577 [Laubach]. This protocol is known as "classical IP over ATM" (for reasons that are discussed later) and introduces the notion of a Logical IP Subnet (LIS). Like a normal IP subnet, a LIS consists of a group of IP nodes (such as hosts or routers) that connect to a single ATM network and belong to the same IP subnet.

To resolve the addresses of nodes within the LIS, each LIS supports a single ATMARP server, while all nodes (LIS Clients) within the LIS are configured with the unique ATM address of the ATMARP server. When a node comes up within the LIS, it first establishes a connection to the ATMARP server, using the configured address. Once the




1. Routing table maps final destination to next hop
2. Address resolution table or server maps next hop IP address to ATM address
3. Signaling creates ATM virtual connection between routers
4. Forward packet over ATM virtual connection

**Figure 29. Routing Across ATM with the Classical Model**

ATMARP server detects a connection from a new LIS client, it transmits an Inverse ARP[53] request to the attaching client and requests the node's IP and ATM addresses, which it stores in its ATMARP table.

Subsequently, any node within the LIS wishing to resolve a destination IP address would send an ATMARP request to the server, which would then respond with a ATMARP reply if an address mapping is found. If not, it returns an ATM_NAK response to indicate the lack of a registered address mapping. The ATMARP server ages out its address table for robustness, unless clients periodically refresh their entry with responses to the servers Inverse ARP queries.

Once an LIS client has obtained the ATM address that corresponds to a particular IP address, it can then set up a connection to the address. A companion specification [Perez] describes how IP over ATM implementations should use UNI 3.0/3.1 signaling procedures for this purpose.

The operation of the classical model is very simple. It does, however, suffer from a number of limitations. One of these limitations is indicated by the phrase “classical.” What this means is that the protocol does not attempt to change the IP host requirement [Braden2] that any packet for a destination outside the source node's IP subnet must be sent to a default router. This requirement, however, is not a good fit to the operation of IP over ATM, and a whole class of other “non-broadcast multi-access” (NBMA) networks, such as frame relay or X.25. In all such networks, it is possible to define multiple LISs, and the network itself could support direct communications between two hosts on two different LISs.

However, since RFC 1577 preserves the host requirements, in the context of IP over ATM, communications between two nodes on two different LISs on the same ATM network must traverse each ATM router on the intermediate hops on the path between the source and destination nodes. This is clearly inefficient, since the ATM routers become bottlenecks; this also precludes the establishment of a single connection with a

[53] As of the time of writing, the IP over ATM group was discussing eliminating the Inverse ARP request from the ATMARP server, and having the server learn this information by observing client messages. This was so as to preclude the ATMARP server from constantly polling nodes for their address mappings, in cases where the nodes do not wish to participate in the 1577 protocol. This case may arise where the ATMARP server is supported on a platform—for instance, an ATM router—which may support connections to many different types of nodes, many of which may not support the 1577 protocol (for instance, because they support a different network layer protocol other than IP).

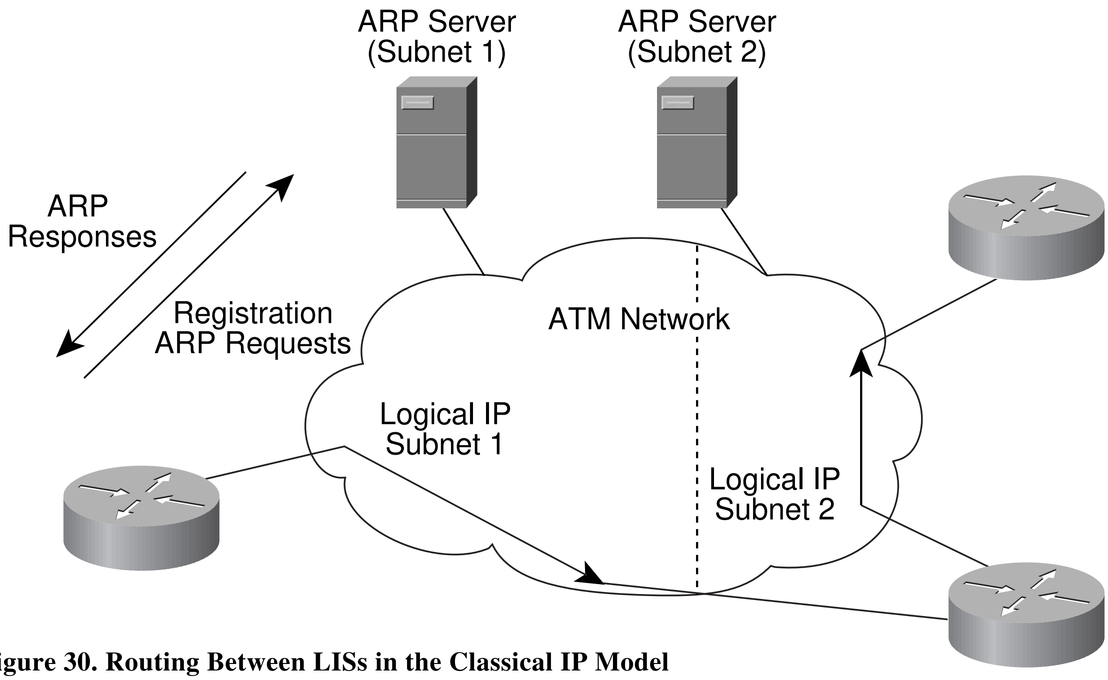



**Figure 30. Routing Between LISs in the Classical IP Model**

requested QoS between the two nodes. The ongoing work on extensions to the classical model to eliminate this limitation is discussed next.

### 6.3 NHRP

As noted above, the classical model for IP over ATM suffers from the limitation imposed by host requirements that preclude "cut-through" routes that bypass intermediate router hops for communications between nodes on the same ATM network, but within two different LISs. The IETF's "Routing over Large Clouds" (ROLC) working group has been working on protocols that overcome this limitation. After considering numerous different approaches [Braden3], the group is now finalizing work on a protocol known as the Next Hop Resolution Protocol (NHRP) [Katz]. In this section we briefly describe the operation of this protocol.

NHRP builds upon the Classical IP model, substituting for the concept of a LIS the notion of a logical "Non-broadcast Multi-access" (NBMA) network—that is, a network technology, such as ATM, Frame Relay, or X.25, which permits multiple devices to be attached to the same network, but which does not easily permit the use of broadcast mechanisms, as are common on LANs. Such a network consists of set of nodes, each of which is attached to the same NBMA network (for the purposes of this paper, this will be an ATM network), and which are not physically or administratively restricted from directly communicating with each other.

Note, however, that a single NBMA network could support multiple administrative domains, within each of which direct connections may be allowed, but between which such connections may be precluded—for example, so as to implement policy firewalls. NHRP is applicable within each administrative region, but will permit direct connections only to the ingress point of another administrative region.

In place of ARP Servers, NHRP uses the notion of a NHRP server (NHS). Each NHS maintains "next-hop resolution" cache tables with IP to ATM address mappings of all those nodes associated with that particular NHS, or for IP address prefixes reachable through nodes (that is, routers) served by the NHS. Nodes are configured with the ATM address of their NHS and then register their own ATM and IP addresses with the NHS, using registration packets, so that the NHS can build its cache tables.

NHSs can be deployed in one of two ways. In the "server" mode, each of the NHSs within a NBMA network are statically configured with the IP addresses of the destinations served by each of the other NHSs in the network. This is adequate for the deployment of NHRP within a small scale NBMA network—for instance, as an upgrade to a network running RFC 1577. The need for configuration of the NHSs, however, restrict server mode deployment to small networks.

In "fabric" mode, the NHSs acquire knowledge of the destinations served by the other NHSs through the use of intradomain and interdomain routing protocols. Furthermore, it is assumed that the NHS serving a particular destination will lie along the routed path to that destination. In practise, this means that all egress routers from the NBMA network must serve as the NHSs for all destinations outside the NBMA network reachable through them, while the routers serving NBMA attached hosts must also act as those host's NHSs.

The mode of the serve deployment, however, is transparent to the end systems—typically hosts or routers—that use the service. The way the protocol works is as follows: when a node determines that it needs to transmit a packet across the NBMA network, and hence needs to resolve a particular ATM address, it formulates and transmits a NHRP request packets and sends it to its NHS. Such requests, as with all NHRP messages, are sent in IP packets.

If the requested destination is served by this NHS it returns the address in a NHS reply to the requester. If it does not, however, the NHS consults its routing table to determine the NHS next on the path to the destination address and forwards the request. At this next NHS, the same algorithm is followed, until a NHS is reached which does indeed know the requested mapping.

This node then returns a NHRP reply, typically traversing, in reverse order, the same sequence of NHSs which lead to it, until the reply reaches the requesting node, which can then set up a direct data connection. The reason the reply generally traverses the return path is so that all the intermediate NHSs can also learn and cache the mapping —then, the next time a node requests that mapping, the NHS can respond directly, without forwarding the request (unless the node requests an "authoritative" mapping, in which case cached information is never used).

While a NHRP request is being processed, the NHRP protocol suggests that a node could optionally forward the packets along the default router path, as opposed to buffering or discarding the packets, so as to reduce latency. The specification does not address, however, any possible packet misordering that this might cause, as and when a direct data connection is eventually set up. While most network layer protocols do not guarantee packet ordering, most implementations implicitly assume this since it greatly improves end system performance.

NHRP also allows for a number of optional features, including route recording, to detect loops within the NBMA network, and fallback, where NHSs, capable also of forwarding packets, along the route to a particular address, can offer to be an intermediate forwarding point for those addresses, in case the actual end-system is not able or willing to support direct data connections.

Another important optional capability is support for address aggregation—NHSs can return not just the NBMA address through which a particular requested IP address is reachable, but also a subnet mask associated with that address. Such information can then be cached, not only by the requesting end system, but also by intermediate NHSs, so that all (non-authoritative) requests for all IP addresses with the same prefix can be responded to with the same NBMA address. Various timers and refresh mechanisms are used to ensure that cached mapping tables do not become stale.

These types of mechanisms can be used to provide firewall protections within an ATM network consisting of multiple administrative domains. In particular, as noted above, an NHRP request would only be forwarded to the ingress NHS of a new administrative domain. Instead of forwarding the NHS request, this ingress NHS could then return the NBMA address of a firewall packet forwarder regulating access to the administrative domain (for instance, a host or router serving as the default exterior gateway). Such a scheme would also rely, however, upon the use of ATM level address filtering to

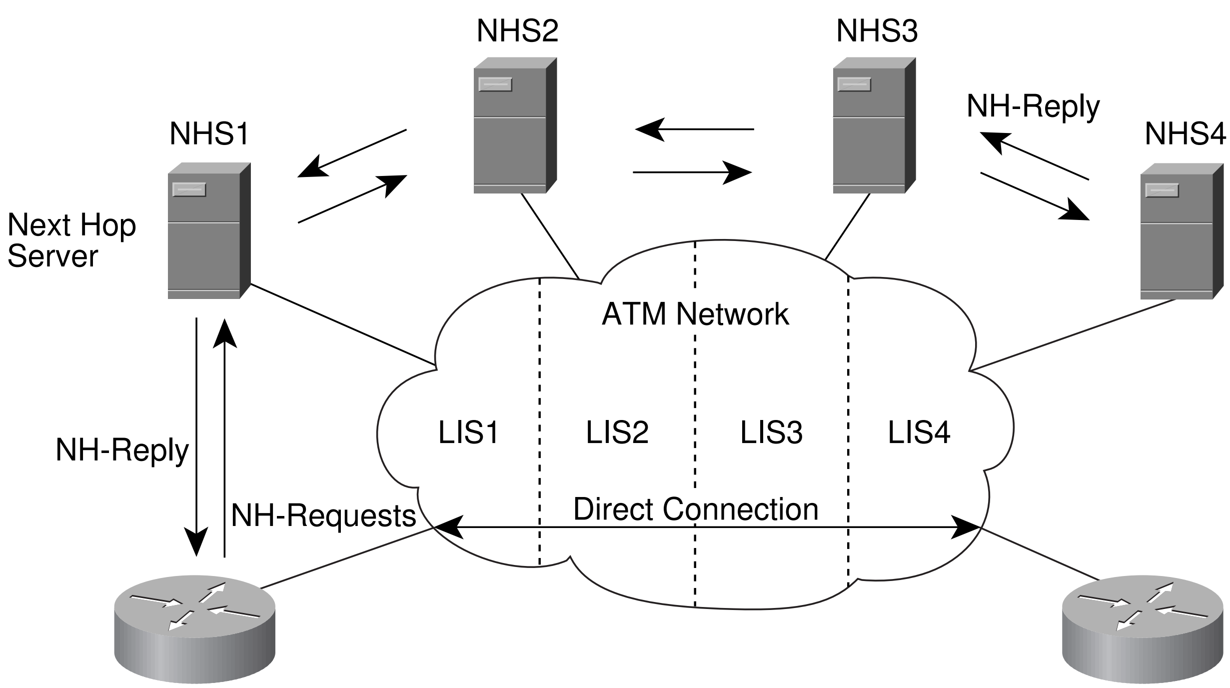



**Figure 31. Operation of NHRP**

preclude direct data connections into the administrative domain in cases where some external entity has learned an ATM address within the domain through other means.

The NHRP protocol may be used for communication either between routers or between hosts. There are some pathological conditions, however, under which a direct router to router connection set up by NHRP may lead to a stable routing loop. This is a consequence of the fact that cut through routes violate a fundamental assumption about IP routing, that routing updates be sent across all paths across which data also flows. NHRP violates this assumption since a cut through route established between two routers is only used for data forwarding and does not establish a router adjacency.

It should be noted that nonetheless this may result in stable loops only in cases where multi-homed networks are connected both across a NHRP network and by "back-door" routes across other network paths. In many cases of interest—for example, the interconnection of multiple networks across a common backbone network, such conditions generally do not apply.

Nonetheless, the ROLC group has been actively discussing ways in which this problem could be resolved.The latest NHRP draft defines a "purge" message which a NHS sends to all nodes that have received, and may have cached, reachability information from the NHS. Such purge messages are sent by NHS if and when they detect any topological change that may effect the validity of the cached information, and causes all recipients to clear their caches with the information received from that NHS. NHS responses also contain a bit to indicate whether or not the responding NHS believes the reachability information to be stable; if it does not, the information cannot be cached by any intermediate NHS.

It would also appear that the stable loop problem may only occur at boundaries between two administrative domains, where the use of such inter-domain routing protocols as BGP result in the loss of route metrics, which may, in turn, hide the existence of such a loop. The use of routers at such boundaries, precluding cut-through routes, is hence a simple fix to this problem, pending possible changes to such inter-domain protocols to correct these limitations.

The stable loop problem also does not arise if one or both of the end points is an end system, since end systems do not forward data. Given this, a very similar protocol to NHRP, the NBMA Address Resolution Protocol (NARP) has also been defined [Heinanen2]. This protocol is a functional subset of NHRP which only returns address mappings for IP addresses of nodes directly connected to the NBMA network, thus precluding the router to router case. It is not clear, however, whether NARP will ever see much deployment given the much greater power and applicability of NHRP.

NHRP will likely be deployed on routers, for use within Frame Relay and X.25 networks, amongst others [Cansever], and it is also likely to be used for router to router communication within some ATM networks. Some specific enhancements may need to be made to NHRP, however, for widespread ATM deployment. For example, NHRP has no support for autoconfiguration, though this has always been a prime focus of ATM standardization efforts. As noted below, it also today has no support for multicast/broadcast operation; as discussed previously these pose particular problems within ATM networks. The NHRP mechanisms today are also very IP specific—for instance, all NHRP messages are sent within IP packets.

Notwithstanding these potential limitations, it is likely that NHRP will play an important role within ATM networks, particularly within the context of the Multiprotocol over ATM (MPOA) work currently being done at the ATM Forum. As we describe later, this work will likely involve extending NHRP so as to make it more complete and ATM specific.

An early goal of the ROLC group was to ensure interoperability between a RFC 1577 compliant end system, and one implementing NHRP; it does not appear today, however, that this goal will be met. As such, interoperability between nodes on a RFC 1577 LIS and nodes on a NHRP network will require that the two networks be interconnected by a router. Similarly, interconnection between a network of either such type and an emulated LAN will also require router support. Some work was still being done, however, within the ROLC group, as of the time of writing, to determine migration paths, perhaps involving dual RFC 1577/NHRP stacks within end systems, which would facilitate a migration from RFC 1577 to NHRP.

### 6.4 Multicast Operation

Today, there is no specific support in the classical IP protocol for multicast operation. This has long been recognized as a critical weakness of RFC 1577, particularly in comparison to LANE. While RFC 1577 could be used to resolve a multicast IP address to an ATM address, this addresses neither the question of how nodes within a LIS could register for membership within an IP multicast group, nor how an IP multicast group could be mapped to a form of ATM multicast.

Recently, however, some work has been done to define a mechanism for multicast in RFC 1577 [Armitage]. This work attempts to support the IP multicast behavior described in RFC 1112 [Deering1], by a combination of multicast servers and overlaid point-to-multipoint connections. This work is currently at an early stage of definition, so only a brief overview of this work is presented here. This work, however, may also serve as a model for multicast support in other protocols, possibly including NHRP and MPOA.

[Armitage] introduces the notion of a Multicast Address Resolution Server (MARS), which can be considered the analog of the ARP server in 1577. A MARS serves a group of nodes known as a “cluster.” All end systems within the cluster are configured with the ATM address of the MARS. The MARS supports multicast through “multicast meshes” of overlaid point-to-multipoint connections, or through multicast servers.

When an end-system wants to transmit to a particular multicast group, it opens a connection to the MARS, and issues a MARS_REQUEST message for that particular group. If any other node has not already registered to join that multicast address (that is, indicated a desire to receive traffic on that group address[54]), the MARS then issues a MARS_NAK, informing the requesting node to “silently” drop the multicast packet. If the MARS has already registered one or more other nodes for that multicast address, however, the operation of the MARS is a function of whether the requested multicast address is configured to be served by a multicast server or by a multicast mesh.

In the multicast server case, the MARS returns a MARS_MULTI message that contains a “server map” of the one or more multicast servers serving the group. The requesting node then sets up a connection (point-to-point or point-to-multipoint, depending upon whether a single or multiple multicast server addresses are returned[55]) to the set of multicast servers and transmits its multicast packets[56].

In the case where the multicast address is served by a multipoint mesh, the MARS returns a MARS_MULTI message that contains a “host map” of addresses of other nodes already registered as members of that group, indicating a desire to receive traffic on the multicast address. In this case, the requesting node constructs a point-to-multipoint connection to that set of nodes and begins to transmit packets on that connection. In either case, mechanisms are used to ensure that the address list is transmitted to the requesting node in a reliable manner.

[54] In the IP context, any node can transmit to a multicast address. However, a specific join protocol that uses IGMP must be used to receive data in a multicast group.

[55] Multiple multicast servers may be used either for load balancing or for redundancy purposes; in either case, the interactions between multicast servers is outside the scope of [Armitage].

[56] Note that in this case a node would receive back its own multicast packets; since many applications cannot tolerate receiving back their own data, devices - particularly routers - would need to filter out any multicast packets received from a multicast server containing its own source IP address. A number of mechanisms for facilitating this operation - including, possibly, changes to the RFC 1483 encapsulations - were under discussion as of the time of writing.

The more complex part of the protocol is how the list of nodes that wish membership in the multicast group is collected so as to receive data. In RFC 1112, a node that wishes membership within a multicast group must generate a Internet Gateway Message Protocol (IGMP) Report message and multicast this to the joining multicast group. The function of this message is to inform all multicast routers on the subnet of the existence of a node that wishes membership in a particular group on that subnet. The routers then use that indication to direct multicast traffic to that subnet, using a multicast routing protocol such as PIM [Deering2]. Note, therefore, that routers must listen “promiscuously” on all multicast groups.

Routers, however, also use a reserved multicast group, identified by the IP address 224.0.0.1, to monitor the status of multicast groups within a subnet. All multicast nodes must also be members of this group. Routers periodically send IGMP Queries for the particular multicast groups which they are currently forwarding to the reserved address. Any node on the subnet that is a member of that multicast group must respond with an IGMP Report message on the queried multicast address, unless some other node responds first. Also, all nodes that wish to participate in multicast operation must join the reserved multicast group in order to receive IGMP Queries.

[Armitage] supports these RFC 1112 requirements by also using the MARS server as a multicast server to support two multicast groups for the reserved multicast group: the ServerControlVC, which links all multicast servers, and ClusterControlVC, which links all end systems (including routers) in the cluster.

Any multicast server that wishes to serve one or more particular multicast groups must first register itself with the MARS to indicate its intentions, using a MARS_MSERV message. The MARS uses such registration messages to construct the server map for each multicast address, which contains the ATM addresses of those servers that wish to serve the particular multicast group, to return it in any subsequent MARS_REQUEST message for the group. The MARS also adds any registering server to its ServerControlVC. Multicast servers obtain the list of nodes that wish to receive data on a particular address by sending a MARS_REQUEST to the MARS, just as with any other end system. The MARS, however, recognizes that the requester is a multicast server by noting its address in the server map, and returns the corresponding host map so that the server can construct its point-to-multipoint connection.

Any end node that wishes to join and transmit to any multicast group—for instance, as triggered by an IGMP Report—must first register with the MARS server, using a MARS_JOIN message for the IP address 0.0.0.0. The MARS then adds the node as a leaf of its ClusterControlVC.

The node can the issue another MAR_JOIN message for to request membership in any IP multicast group. The MARS server then stores the address of the requesting node in the host list that is associated with that group, so it can be returned in any subsequent MARS_REQUEST message for the group. The MARS then adds any node that sends a MARS_REQUEST for the group to this VC.

Note that all nodes in the cluster, regardless of whether or not they wish to transmit data to a group, must also send a MARS_JOIN to be added to the multicast group for the reserved address. The subsequent operation of the MARS is then a function of whether the group is being served by a multicast mesh or by multicast servers.

In the former case, where multicast meshes are used, the MARS forwards the MARS_JOIN message on the Cluster-ControlVC to inform any nodes that may already be members of the requested multicast group of the existence of a new member. All nodes transmitting to the group over existing point-to-multipoint connections then add the new requesting node to their connections using add-leaf messages.

Similarly, any node that wishes to leave a multicast mesh multicast group sends a MARS_LEAVE request to the MARS Server. This removes the node's ATM address from the list of ATM addresses registered with the IP multicast address and then forwards the message on its ClusterControlVC. This allows transmitting end systems to remove the leaving node from their point-to-multipoint connection. Transmitting nodes use timers and other mechanisms to clear inactive connections and conserve connection resources.

In the case of group served by multicast servers, the MARS forwards any MARS_JOIN or MARS_LEAVE request to the registered multicast servers using the ServerControlVC. This allows the relevant multicast servers, which serve the group in concern, to either add or delete the requesting node from their own point-to-multipoint connections.

Multicast routers form a special case of end systems since they must, as per RFC 1112, receive IGMP Reports on any and all multicast group addresses. They must promiscuously join all groups by sending a block join message to the MARS for all addresses. Any node that sends a MARS_REQUEST subsequently ends up also transmitting to the router, either through a multicast server, or through its own point-to-multipoint connection. Note, however, that while routers must register to join all multicast groups, they do not need to allocate connections to any groups that do not have transmitting nodes. [Armitage] also proposes mechanisms to allow routers to register and to promiscuously listen to only a subset of multicast connections. Routers must also register to transmit to the reserved group by sending a MARS_REQUEST for the reserved address.

Routers then use the reserved multicast group to transmit IGMP messages. Since all nodes that are members of multicast groups are also members of this reserved group, they monitor such IGMP Queries and respond to the corresponding multicast groups. The routers serving these groups then receive the IGMP Responses.

[Armitage] also presents some discussion of redundant server operation, the operation of "mixed" groups, where a single multicast group is served by a combination of multicast meshes and server, and so on. This work is still currently under development by the IP-Over-ATM working group.

As of the time of writing there had been no formal work on the support of multicast within NHRP or, more generally, on the support of multicast groups within a NBMA domain where cut-through routes are supported. Some preliminary work, however, would appear to indicate that extending the MARS protocol to such an environment should be relatively straightforward, at least for such advanced routing protocols as PIM. Specifically, the cluster notion of [Armitage] would be extended to include nodes from all of the subnets supported on the NBMA fabric, and the multicast distribution connections, be they from multicast servers, or point-to-multipoint meshes, would include requesting nodes from any of the subnets. The multicast routers connected to that domain would be configured to transmit only a single copy of the packets of any requested multicast group onto that fabric, and not one to each of the subnets on which requesting nodes might be, as they would normally. It would appear that PIM, at least, can readily support such a mode of operation.

### 6.5 Direct versus Router Connections

One of the limitations of 1577 is that it does not address the issue of connection set-up latency. Unlike LANE, it does not have a default data path on which data can be sent prior to address resolution, connection routing, and establishment. There has been some recent work [Rekhter] that raises interesting questions about the role of routers in native mode ATM environments.

[Rekhter] proposes that direct ATM paths, either within or between LISs, be used only where the IP flows require the QoS guarantees provided by ATM. In such cases, it is presumed that the high connection set-up latency is acceptable. For all other cases, however, [Rekhter] suggests that all data is to be relayed through one or more routers, even when the data flow is within a single IP subnet (LIS), to avoid this latency. This behavior requires changes to the current operation of routers, since today they would send ICMP redirects for packets that are sent to them for a local subnet. It is also not clear, moreover, that such an operation is optimal since connections that do not require guaranteed QoS might still use more bandwidth than a router can handle.

A better approach might be to segregate direct connections and router-relayed flows by the “volatility” of the data flow along the connection. That is, long lived, high bandwidth flows should use direct ATM connections, independent of whether or not they require guaranteed QoS, while low bandwidth, short lived data flows should be sent through a router since such flows would not justify the latency of a connection set-up. This approach would be a more optimal solution than requiring direct connections for all data flows, especially since many such flows (such as telnet traffic or SNMP traps) in networks consume very little bandwidth but do require low latencies, and hence could easily be handled by routers.

It is likely, therefore, that in many production ATM networks, routers will continue to provide such “connectionless” service, while high volume data transfers (such as FTP) would be done over direct ATM connections using native mode protocols. The NHRP specification does suggest the possible use of local routers as connectionless servers for such traffic flows. The Multiprotocol Over ATM (MPOA) work currently being developed by the ATM Forum will likely support such modes of operation. This work will be discussed in the next section.

## 7.0 Multiprotocol Over ATM

Notwithstanding the work done on native mode protocol support for IP over ATM, there is widespread consensus in the industry that more needs to be done to accelerate native mode protocol development, particularly to correct the limitations of the existing native mode protocols, and to include protocols other than IP. To this end, the ATM Forum has recently set up a working group to consider the development of “multiprotocol over ATM” (MPOA) standards. While this work is at a very early stage, the group has considered various approaches to the problems. These are briefly described here, since they serve to indicate some future directions for internetworking across ATM.

Three very different models have been presented for multiprotocol operation over ATM:

### 7.1 Peer Models

A number of contributions have proposed a new variant of the peer model as a replacement for the current overlay model ([Perkins1], [Perkins2], [Fink]). Unlike the earlier peer model that proposed that ATM networks also use current network layer addressing schemes and routing protocols, these new proposals suggest a different approach. They propose an algorithmic mapping of all network layer addresses into NSAP format addresses, so that the signaling requests that contain such addresses can be routed using the P-NNI protocol. This precludes the need for a separate address resolution protocol. It is not clear, however, whether such peer addressing models would necessarily solve the concerns about sub-optimal end-to-end routing, within a mixed ATM and router environment, since they propose that different routing protocols be run within the two networks.

All ATM switches would also need to support address tables large enough to incorporate not only ATM NSAP addresses, but all other address spaces as well. It is also not clear how well such a peer network would work in an environment that consists of a mixture of ATM and non-ATM, router-based networks. Concerns have been raised, for instance, about the difficulties of properly mapping such subsidiary protocols as ICMP properly into ATM in a peer model.

### 7.2 Integrated P-NNI

The new peer models described above assume that routers outside the ATM network continue to use existing routing protocols. The Integrated P-NNI model (I-PNNI) instead proposes that the P-NNI protocol is to be used by both ATM switches and by packet routers ([Callon1], [Callon2]). This is based on the notion that the P-NNI protocol is a significantly more powerful and scalable routing protocol than any that exist in current routed networks. With a few modifications such as precluding ATM connections being routed through routers (a problem that the peer model may suffer from), it may well prove possible to operate P-NNI throughout a packet or cell-based network.

Routers running I-PNNI would support a hierarchy similar to ATM switching systems, electing PGLs, and so on. ATM switch PTSPs would also be forwarded to routers, to allow them to generate optimal end-to-end routes through both the routed and switched network. The I-PNNI model could accommodate both the overlay and peer models. In the peer model, network layer reachability information would be carried transparently through the ATM network, while in the overlay model, the addresses would be mapped into NSAP addresses and processed by ATM switches as any other set of reachable addresses.

I-PNNI may well hold promise as a routing protocol for the Integrated Services Internet, since it both supports QoS routing, and integrates well with ATM backbones, which will surely be a major component of the new Internet. On the other hand, a number of significant technical and administrative issues (for example, migration from existing, deployed routing protocols) must first be tackled before the Integrated Services Internet can be deployed in any widespread manner, hence it will likely be a couple of years before the significance and role of I-PNNI is fully clarified.

### 7.3 Distributed Router Protocols

A different approach to the multiprotocol over ATM work effort was proposed to the ATM Forum by Cisco Systems [Alles3]. Cisco proposed that the MPOA work should be based around a new vision of virtual LANs, that would extend beyond the first generation of LANE-based VLANs.

As noted in Section 5.0, the first generation of virtual LANs are built around layer 2 LAN switches and support the LANE protocol. As also noted in that section, this approach suffers from two problems: the bottleneck of requiring router hops for virtual LAN interconnection and the inability to run protocols in native mode, which exploit the QoS features of ATM.

Beyond this first generation of LANE-based layer 2 LAN switches, a number of companies have announced plans to develop a new generation of layer 3 LAN switching systems, including Cisco Systems, with its CiscoFusion™ architecture. Such switches would act not as simple bridges—that is, switching packets purely on the basis of MAC address information—but would also switch packets based on their network layer addresses and other higher layer attributes. In essence, a system of such layer 3 switches would constitute a distributed router.

Layer 3 based VLANs would provide a number of advantages over LANE based layer 2 VLANs:

1. They could minimize the need for multiple hops through ATM routers for communications between two nodes on different virtual LANs.
2. As with current routers, layer 3 switches could reduce such link layer phenomena as broadcast storms and yield more robust, scalable and more easily diagnosable networks. In particular, layer 3 based systems, being capable of directly routing all packets, would not need the flooding mechanisms of layer 2 based systems, which tends to be a fundamental constraint upon the scalability of the latter.
3. Layer 3 switches, by allowing operations on higher layer fields, could give network administrators more control over networks through such mechanisms as filtering on higher layer attributes.
4. Layer 3 switches could more easily use the QoS benefits of ATM by running native mode protocols, as described earlier. In particular, layer 3 switches, by being capable of interpreting and processing layer 3 packet headers, could trap control messages from protocols like RSVP, and use these to set up ATM connections with the appropriate QoS. Similarly, they could map layer 3 flows to corresponding ATM connections. Such operation are much more difficult to do within layer 2 switches, since such products typically only process layer 2 packet headers.

To make layer 3 switches cost-effective for work group deployment, however, such devices could not be built in the same manner that routers are built today. Router ports today are rarely dedicated to a single, or small number of users, due to their cost and complexity, which follows naturally from their much greater sophistication, versus simple LAN concentrators or (bridging) layer 2 switches. Layer 3 switches, however, could potentially be built to be much simpler, and hence cheaper, than today's full-function routers. How this is possible can be understood by examining the internal structure of current routers.

A router performs two quite distinct functions:

1. *Route Processing:* This is the processing of routing protocols—such as EIGRP, OSPF, or BGP—to determine reachability information and calculate next hop routing tables (to know where to forward a packet that is received by the router). Route processing represents the "intelligence" of current routers.
2. *Packet Switching:* This is the actual forwarding of a received packet on the basis of the source and destination (layer 3 or layer 2) addresses of the packet, and the next hop routing information in the router. A number of other packet-level functions (such as filtering) may also be performed during the forwarding operation.

In most modern high performance routers, these functions are performed by distinct components. Route processing is a software-intensive function that is typically performed in a fast (often RISC) processor; such processors typically also have a considerable amount of fast memory to accommodate large routing tables. Packet switching, on the other hand, is often carried out by special purpose hardware, and is optimized for packet processing. Such specialized but relatively "unintelligent" hardware is supplied with forwarding information by the route processor.

To make layer 3 switches cost-effective for workgroup deployment, such switches will need a different architecture from existing router designs. In particular, much of the cost of routers today is represented by the high performance processors and memory systems required for route processing. Given the increase in the size of internetworks, it is likely that route processors will need to continue to increase in performance and memory, and cost. On the other hand, because the packet switching function is primarily hardware based, it can ride the ASIC cost curve and will continue to decrease in cost while increasing in performance.

Given this, the most cost effective architecture of a layer 3 switching system would be to have specialized hardware intensive devices for packet forwarding that are distributed to work groups, where such devices would not all have integrated route processors. Rather, many such layer 3 switches would use the services of a centralized route processor, hence reducing their cost. Centralizing route processing would also facilitate centralized management, easing the administrative burden of managing many, distributed routers.

A route distribution protocol would be used by the route processors to download the information required by the layer 3 switches to forward packets received across their (non-ATM) LAN or WAN ports. The following discusses what this information might be and how such a system would operate.

Architectures similar to this have been described by a number of vendors, with the layer 3 switches described variously as Multilayer Switches [Cisco], Edge Routers, or Virtual Routers. The basic principles of operation of each of these, however, is very similar.

Many aspects of such systems—the internal operation of the layer 3 switches, the routing protocols performed in the route processors, and so on—would be beyond the scope of standardization, and would allow for individual vendor value-add and differentiation. There would be value, however, to standardizing the Route Distribution Protocol that is used to communicate between the route processor and the layer 3 switches, since this would allow for open, multivendor layer 3 VLAN networks, mixing route processors and layer 3 switches from multiple vendors[57].

Cisco proposed, therefore, that the ATM Forum MPOA subworking group develop such a route distribution protocol [Alles3]. Cisco also proposed a number of requirements for such a protocol, which were subsequently adopted by the group [Brown], and the MPOA group is now engaged in developing such a protocol (the "MPOA protocol"), based upon these requirements and scope.

In particular, Cisco proposed to the MPOA group a strawman network architecture and protocol reference model, which described the types of problems the MPOA protocol would need to solve, and the types of approaches that could be taken for the protocol [Alles4]. As of the time of writing, a consensus had emerged within the MPOA group upon these aspects, along the lines of the concepts put forward by Cisco. While the MPOA specification was still at a very early stage, as of the time of writing, the general outlines and operation of the protocol were clear; these are described below.

As proposed in [Alles4], an MPOA system would consist of a collection of: Layer 3 switches (called *Edge Devices* in the MPOA specification) which support one or more ports to legacy LAN or WAN networks; ATM-attached end-systems implementing the MPOA protocol (called *MPOA hosts*); and Route Servers, all connected to an ATM network. Edge devices would implement layer 3 packet forwarding, but would not support routing protocols. These would be implemented on the route servers, which would interact with each other, and with conventional routers (either on, or outside, the ATM network), using conventional routing protocols (e.g. EIGRP, OSPF, etc.).

All MPOA-capable devices—MPOA hosts, edge devices, and routers[58]—would support a MPOA client, where each such client would support both one or more layer 3 addresses, and an ATM address. The layer 3 addresses associated with a MPOA client would represent either the layer 3 address of the associated node itself (in the case of a MPOA host, for instance), or the layer 3 addresses (e.g. IP subnets) reachable through the node (in the case of a edge device or router). MPOA clients would connect to a MPOA server, and register their ATM addresses, as well as the layer 3 addresses reachable through them.

[57] Note that what would be standardized would only be a mechanism for transporting routes from the route processor; the protocol is not a substitute for routing. In other words, the route servers still must operate routing protocols, and still represent the "intelligence" in the network. The protocol would allow, however, for third party layer 3 switches to interface to a route server, and hence gain access to this network intelligence.

[58] A router may be differentiated from an edge device in that while they both forward packets on the basis of layer 3 addresses, the latter does not implement routing protocols.

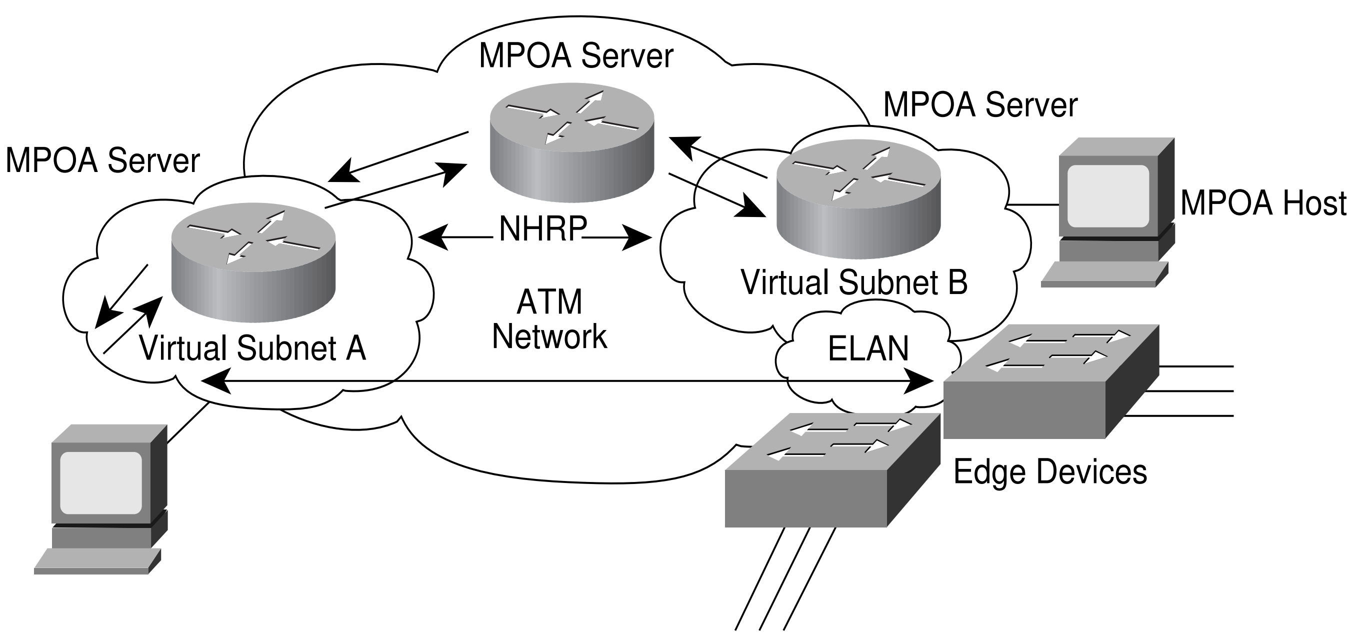



**Figure 32. Architecture of the MPOA Protocol**

MPOA clients would also implement a set of protocols to interact with the MPOA route servers. These protocols would allow either edge devices, or MPOA hosts, to set up direct data connections across the ATM network with each other, based upon the layer 3 addresses of the destination nodes. Such connections, moreover, must allow for cut-through routes—that is, direct connections between two MPOA clients on two different layer 3 subnets (e.g., IP subnets) must be possible, without necessitating an intermediate hop through a router.

In order to set up such connections, the MPOA clients require two pieces of information: next hop layer 3 reachability information, and ATM address resolution.

The former is required in order to determine the layer 3 address of the node (i.e. MPOA client) which either supports the destination layer 3 address, or through which the destination layer 3 address is reachable. Note that since the MPOA protocol must support cut-through routes that this next hop address must be that of the “final” node on the ATM network through which the layer 3 address is reachable, and not that of an intermediate node, such as a router.

Once this final next hop node is identified, the MPOA client then will need to resolve the next hop node’s layer 3 address, to its corresponding ATM address. In practice, these two functions will be combined into a single request. but the functions remain logically disjoint.

Note that this logical client-server structure, and the functions performed by the protocol, are quite analogous to those of the LAN Emulation protocol. Where LANE determines the LEC client through which a particular MAC address is reachable, and the client’s ATM address, so MPOA performs corresponding operations upon layer 3 addresses. Similarly, while the LANE protocol is complicated by the support of MAC bridges, and the extension of ELANs across and between such bridges, so the MPOA protocol is complicated by the need to extend layer 3 subnets across and between edge switches.

Specifically, a layer 2 virtual LAN, as discussed earlier, consists of the set of bridged end-systems reachable through each of the LAN ports on the layer 2 switches configured to belong to a single ELAN. A layer 3 virtual LAN, correspondingly, would consist of each of the end-systems, reachable through the LAN or WAN ports of MPOA edge devices, which share a common layer 3 subnet (e.g. are configured with the same IP subnet prefix).

All of the edge devices supporting ports with end systems with such common layer 3 subnets are considered to belong to the same *virtual subnet*. As of the time of writing, it was not clear whether the MPOA protocol would be capable of allowing MPOA hosts to belong to a virtual subnet—that is, to share a common layer 3 address subnet prefix with a virtual subnet. This is unlike the case with LANE where both ATM attached end-systems, and end-systems reachable through layer 2 switches can be bridged into the same ELAN. In any case, however, the MPOA protocol must support direct connections between edge devices in the same virtual subnet, between two MPOA hosts, regardless of address, and between MPOA hosts and edge devices with MPOA edge devices in different virtual subnets.

Cisco proposed in [Alles4] that these different types of connections represented different scales of problems, and require different protocol solutions. In particular, the “large scale” problem of determining the next hop address corresponding to a particular destination layer 3 address, and the corresponding ATM address of the MPOA client through which the address is reachable, is essentially similar to the problem space being tackled by the NHRP protocol. The latter, in particular, is designed to solve the joint next hop and address resolution problems, while also delivering cut through routes.

Hence Cisco proposed that the layer 3 next hop and address resolution components of the MPOA protocol be tackled with a protocol based upon NHRP [Benham]. Specifically, the MPOA effort would specify a single query/response protocol that any MPOA client would use, when presented with a destination layer 3 address, to request the corresponding next hop and address resolution information from the MPOA client’s route server. This protocol would essentially be the same as NHRP, albeit with some modifications[59] to make it more ATM, and less IP, specific (e.g to eliminate the current IP packet encapsulation used in NHRP).

MPOA route servers would then operate much as with fabric mode based NHRP servers, operating routing protocols between themselves, and with routers, and forwarding next hop requests between themselves, so as to determine the required next hop address, and to resolve the corresponding ATM address. The goal would be to develop the MPOA protocol such that all directly attached ATM hosts could then adopt the MPOA protocol, in preference to other native mode protocols like NHRP, and hence allow convergence on a single native mode protocol.

The MPOA working group itself, would focus on the support of IP, since this is an open protocol, but it is the hope of the group that other bodies or organizations would use the MPOA work as a template for the native mode support of other protocols. The use of NHRP will likely help accelerate the development of MPOA, since NHRP has already been worked on for some time by the IETF.

NHRP, on the other hand, does not support the notion of edge devices, or distributed virtual subnets, since it assumes that only routers and end-systems are attached to the NBMA network. Edge devices, with virtual subnets, adds the com-

[59] Modifications might also be made to allow whole next-hop tables to be downloaded into edge devices.

plexity of needing to determine through which port, of which edge device, a particular end system may be reachable. This is not a layer 3 routing problem per se, since all of the edge devices in the virtual subnet share a common layer 3 address prefix. Rather, the only way in which the appropriate edge device port can be found is through the use of layer 2 information.

This arises from the two different ways in which subnets are viewed and treated within layer 3 networks. At one level, the function of subnets is to facilitate layer 3 routing, by allowing for address summarization and hierarchical routing. Hence, particular route servers would be associated with particular layer 3 subnets—that is, all MPOA clients linked to that server would share a common subnet prefix—and would report reachability to that prefix using the MPOA protocol.

MPOA *hosts*, on the other hand, would not need to be concerned with the notion of subnets at all (e.g., perform "mask and match" operations or be configured with default router addresses, in the case of IP—since the MPOA protocol would support cut-through routes, obviating the distinction between connections to systems with the same or different subnet prefixes.

On the other hand, subnets are of great importance to edge devices, because they support "legacy" LAN or WAN ports, attached to which are "classical" end systems which, as they do today, are indeed cognizant of subnets. In particular, such nodes typically treat packets to other nodes with the same subnet address differently from those to nodes with different subnet addresses. This is because most protocols, such as IP, have associated subnets not only with address summarization, but also with the operation of broadcast LAN media. Hence, in the case of IP, for instance, hosts act as if an IP subnet is bound to a particular LAN segment, and broadcast ARP packets for nodes within the same subnet, while forwarding off-subnet packets to a default router.

In order to support classical hosts reachable through edge devices, therefore, an MPOA system will hence need to essentially make a particular virtual subnet look, to the classical hosts, like a single broadcast domain. That is, all of the edge device LAN or WAN ports within a single virtual subnet would need to be bridged together. In order to do this, the MPOA protocol must interface with a *layer 2 subnet virtualization protocol*, which provides this bridging function. Cisco noted to the MPOA group that the requirements of this protocol correspond closely with those of LAN emulation, and that some variant of LANE[60] would hence be the natural choice for the virtualization protocol [Finn]. This would allow for a natural evolution path from LANE based VLANs to MPOA, while also allowing for synergies in development. The MPOA group has now accepted this position.

[60] Some changes will likely be necessary to the phase 1 LANE protocol for MPOA purposes (e.g., to support QoS, or to allow possibly for more efficient encapsulations). To this end, the LANE and MPOA subworking groups will likely align their efforts so that the MPOA requirements drive any future enhancement of the LUNI protocol.

It is not clear whether the virtualization protocol will be developed by the MPOA group, or will formally be part of the MPOA protocol. On the other hand, much as LANE was developed in full cognizance of the 802.1d spanning tree protocols, so the MPOA "long range" protocol will need to be developed with a good understanding of the "hooks" required to support a LANE-based virtualization protocol. In particular, this will be necessary in order to efficiently solve the problem of determining through which port of which edge device a particular end system on a virtual subnet is reachable.

While route servers could participate as members of the virtual LAN, and use the flooding mechanisms, to make this determination, efficiencies could be gained by the implementation of the edge devices—even if not the MPOA protocol—coupling the layer 2 and layer 3 operations.

For instance, in the case of IP end systems, the edge devices could monitor packet flows through their LAN and WAN ports and track *responses* to IP ARP messages within a subnet to determine edge device port to layer 3 address mappings. That is, the edge device could determine from observing which port the ARP response for a particular IP address was received from, the port through which that IP address was reachable, as well as the MAC address of the end system supporting that IP address. It could then register this mapping, together with its own ATM address with its MPOA server, so that the MPOA server could respond with the edge device's ATM address upon receipt of a MPOA request for that IP address.

Once a direct data connection is set up from the source MPOA client to that edge device, the latter could then use the IP address to port mapping table to determine which port to send out the received packets, and use the cached MAC address information to construct the required MAC packet for transmission out of the legacy port.

Note, however, that if a particular classical host had never sent a packet through its edge device, then no edge device would have a record of through which port the host was reachable. In such a case, the LANE flooding procedures would be needed to send a packet (for example, an ARP broadcast for the IP address) to that node, in the hope of eliciting a response through which its location could be learned.

Similar operations would be possible with other protocols. In the case of a protocol like CLNP, for instance, which uses advertisements rather than ARPs to determine address mappings, edge devices could trap End Systems to Intermediate System (ES-IS) Hello messages to determine such port mappings. Such operations highlight the fact that while MPOA edge devices may well use variants of LANE for subnet virtualization, this does not mean that such devices will operate in the same manner as layer 2 switches implementing LAN emulation.

In particular, edge devices will need to be capable of observing and processing layer 3 addresses of packets received across legacy ports, to determine whether the destination lies outside or inside the source virtual subnet[61].

In the former case, the edge device would formulate and send to its associated MPOA server a MPOA request for the destination layer 3 address. This, in turn, would operate a NHRP like protocol, as noted above, to determine the corresponding next hop MPOA client layer 3 address, and corresponding ATM address, and return this to the requesting edge device. The edge device would then set up a direct, cut-through connection to the destination MPOA client and forward the data.

Note, however, that if the destination address was reachable through an edge device, on another virtual subnet, then the MPOA server corresponding to that virtual subnet would need to use some of the procedures discussed previously to determine the ATM address of the final edge device.

In the case where the edge device determines that the destination layer 3 address is within the source virtual subnet, it could use the LANE procedures to determine the destination edge device, within that same virtual subnet, through which the address is reachable, and set up a data direct connection.

[61] In the case of IP, for instance, a packet from a classical host on a legacy port would carry the MAC address of the "default router" of the subnet—which may well be a MAC address associated with that edge device's MPOA client—if the packet were addressed outside the host's own subnet.

Even in such a case, however, the edge device may well interpret the layer 3 packet information—for instance, to set up a data direct connection with a requested QoS.

The MPOA protocol may also borrow, or build upon, other LANE mechanisms. For instance, a LECS like MPOA configuration server may well be defined to allow MPOA clients to determine which MPOA servers to register with, depending upon, perhaps, their particular subnet address. As with NHRP, MPOA will also likely support the notion of a default data forwarder, which MPOA clients may choose to use to forward layer 3 packets, pending a successful address resolution. Note that such default data forwarders are essentially routers. MPOA will also need to support mechanisms for multicast address registration—likely building upon the work done for IP over ATM, discussed previously.

Work on defining the MPOA protocol is still at a very early stage, so further details will not be presented here. It is unlikely that the MPOA protocol will be fully specified before early 1996.

It should be noted that while MPOA will build upon NHRP and LANE, it is not clear that MPOA clients will be directly interoperable with nodes that implement these protocols (any more so than 1577, LANE, and NHRP are interoperable). The goal is to develop a single protocol to which all nodes could eventually migrate; in the meantime, however, internetworking devices such as routers, will be needed to interconnect nodes that implement each of these protocols.

The ATM Forum has currently determined that it should focus on the development of the MPOA protocol discussed above, and has deferred work on any peer models. This was based on the realization that the three approaches are solutions to different problems, and could indeed complement each other.

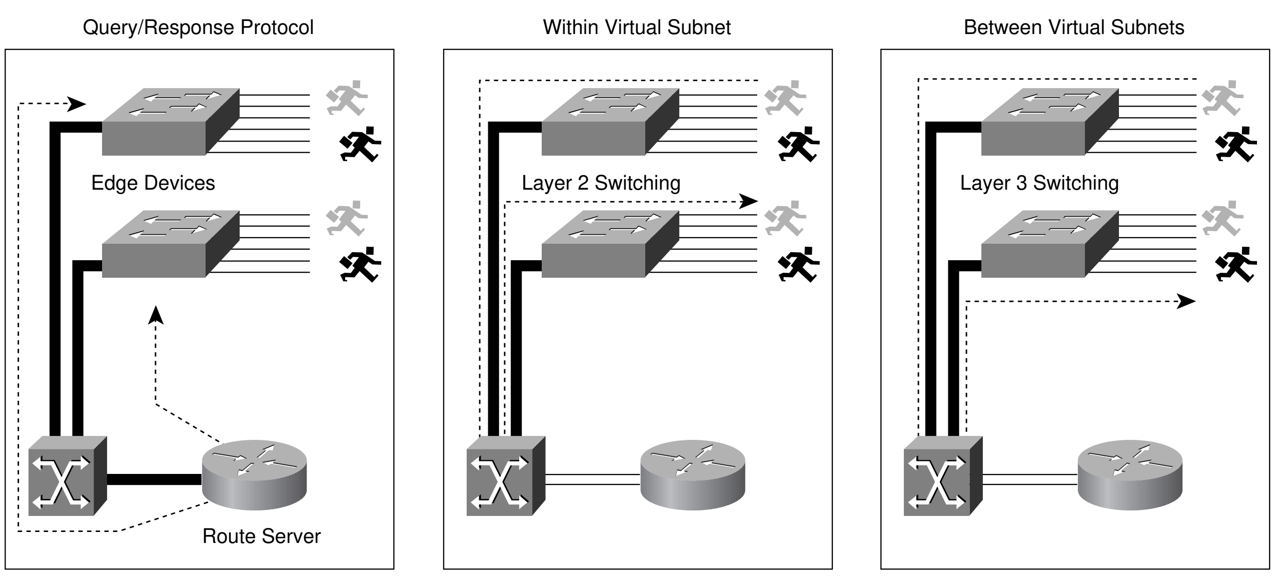



**Figure 33. Distributed Router Operation**

MPOA for instance, is aimed mostly at building distributed routers; this problem is dependent upon the nature of the routing and addressing models used within the ATM network, but also requires the solution to many other independent problems. The I-PNNI model holds great promise, but also probably cannot be fully tackled until the P-NNI Phase 1 protocol is fully defined. The peer model, on the other hand, at best, can be viewed as an optimization of the I-PNNI model, obviating the need for address resolution. Integrated routing, on the other hand, does not necessarily imply or require integrated addressing.

The MPOA group proposes hence to first focus its efforts on the development of the MPOA protocol for overlay ATM networks, while working in parallel to finish the P-NNI Phase 1 protocol, then extend it for I-PNNI. Once this is done, it may reconsider any peer models.

## 8.0 Wide Area Network Internetworking

The previous sections have discussed various ways of internetworking existing LAN and network layer protocols with ATM. There are also, however, a number of existing wide area networking protocols, and some work has also been done on ways in which these protocols could internetwork with ATM. In particular, work has been done on the internetworking with ATM of connection oriented Frame Relay networks and connectionless Switched Multimegabit Data Service[62] (SMDS) networks.

[62] SMDS is a service offered in the United States. In Europe, an almost identical service is known as the Connectionless Broadband Data Service (CBDS). The internetworking scheme described here for SMDS also applies to CBDS.

Together, the Frame Relay Forum and the ATM Forum has specified an implementation agreement [Forum8] to support Frame Relay/ATM PVC interworking based upon the ITU-T I.555 Recommendation [ITU1]. This defines the mapping of Frame Relay packets into AAL5 packets at a Frame Relay-to-ATM interworking unit. The basic operation is very simple: the Frame Relay Data Link Connection Identifier (DLCI) is mapped directly into the VPI/VCI value of the AAL5 packet, and vice versa. Procedures are also defined for mapping various Frame Relay specific header fields into their analogs within the ATM network (for example, the Frame Relay congestion indication bits into the ATM EFCI bit, and the Discard Eligibility bits into the ATM CLP bit), and for mapping the Committed Information Rate (CIR) of Frame Relay connections into VBR traffic parameters.

The only complication in FR/ATM interworking is in the protocol identifiers used for encapsulated packets. Within Frame Relay networks, a Network Layer Protocol ID (NLPID) header is appended to any encapsulated packet to identify its type, as defined in RFC 1490 [Bradley]. Within ATM networks, as noted previously, the LLC/SNAP encapsulation method is more common, as defined in RFC 1483 [Heinanen1]. A FR/ATM interworking unit will need to modify these headers before packets are forwarded.

SMDS internetworking with ATM is also relatively simple, as defined in ITU-T I.364 [ITU2] and the implementation agreements reached between the ATM Forum and the US and European SMDS Interest Groups. SMDS packets are mapped into AAL 3/4 packets at an interworking unit, and are then carried within the AAL 3/4 cells on a well known VPI/VCI value. The SMDS connectionless service is emulated by a connectionless server within the ATM network that receives all SMDS packets sent across the well known VPI/VCI value.

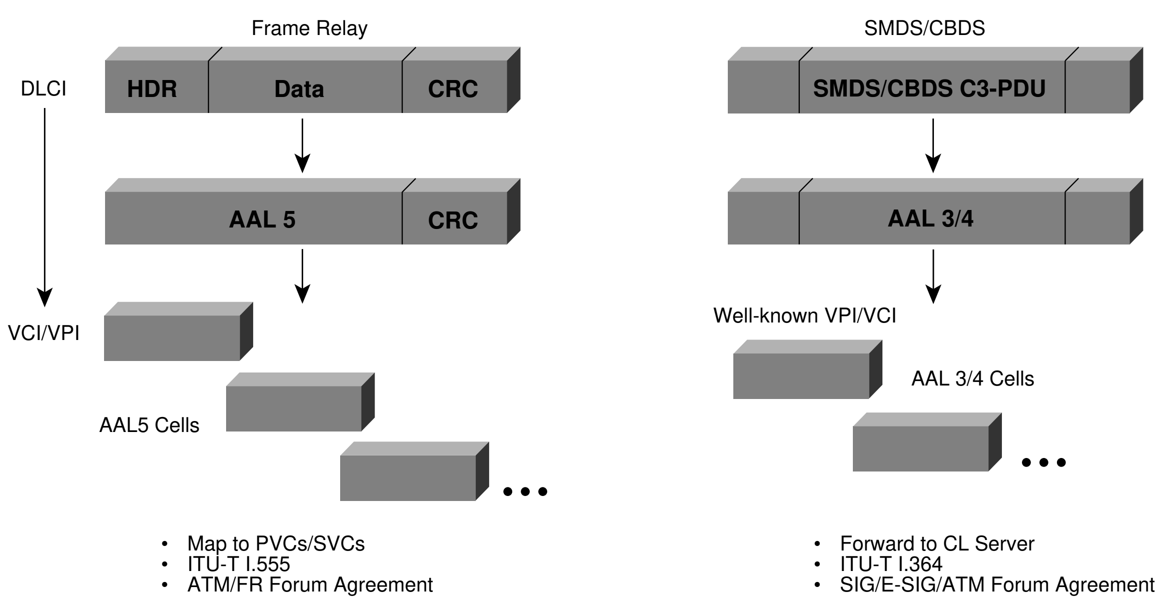



**Figure 34. Frame Relay and SMDS to ATM Interworking**

It then forwards them on the basis of the encapsulated SMDS addresses. Some procedures are also defined for mapping SMDS access classes into ATM traffic parameters.

In addition to the work on Frame Relay and SMDS, some work has recently started at the ATM Forum and within the ITU-T on Narrowband ISDN internetworking with ATM; others have also expressed interest in X.25/ATM internetworking.

Such internetworking specifications will serve two purposes. First, they will be used by public network providers to converge their existing WAN networks together over a common ATM backbone network, realizing economies of scale and preparing for a possible movement to a native ATM service. To allow for such convergence, many public network provides are deploying multiservice platforms that support multiple types of WAN interfaces and interconnect with each other across ATM links.

Second, such specifications may also be used to define services provided across public UNI. Instead of a native ATM service, the public network provider can provide a Frame Relay or SMDS over ATM service interface to the end user. This may facilitate a migration to ATM for existing users of current WAN technologies.

There is also much interest within the public network community on methods of providing LAN interconnect services across public ATM networks. LAN emulation may prove to be one solution to this problem, but concerns have been raised about its scalability, due to the need for flooding through the BUS, and also the reliability issues due to the single point of failure in the LANE Phase 1 protocol. The MPOA protocol may prove to be a better solution in the long term, since it will allow for the scalability and robustness of a routed solution, while allowing for ease of administration, due to the centralizing of the routing functions. Much work remains to be done, however, in fully scoping and specifying such LAN interconnect services.

## 9.0 CONCLUSIONS

The goal of the many protocols described in this paper is to enable the deployment of switched internetworks—networks that consist of a combination of ATM switches, ATM routers, and LAN switches. Switched internetworks will offer significantly greater bandwidth, flexibility, and QoS service support than is possible on today's networks built with shared media hubs and routed internetworks. The deployment of switched internetworks will change the face of networks, in the wiring closet, within the enterprise backbone, and beyond. It is possible today to put together a road map for how such networks would be built and how they will evolve.

The core of such networks will be built with ATM switches. Today, with the UNI signaling protocols, it is possible to deploy small-scale ATM backbone networks; for instance, "router clusters" that consist of multiple collapsed backbone routers interconnected by ATM switches. Such router clusters are often used to replace existing FDDI backbones, since they offer considerably more bandwidth. The development of the IISP protocol will allow such small networks to scale to a dozen or so switches, perhaps spanning a campus, while the full P-NNI protocols will eventually allow such networks to span entire enterprises.

Attached to such ATM backbones will be a combination of ATM workgroup and LAN switches for desktop connectivity. The latter, in particular, are likely to become the dominant desktop networking device, supplanting shared media hubs, since they offer users significantly greater bandwidth, more

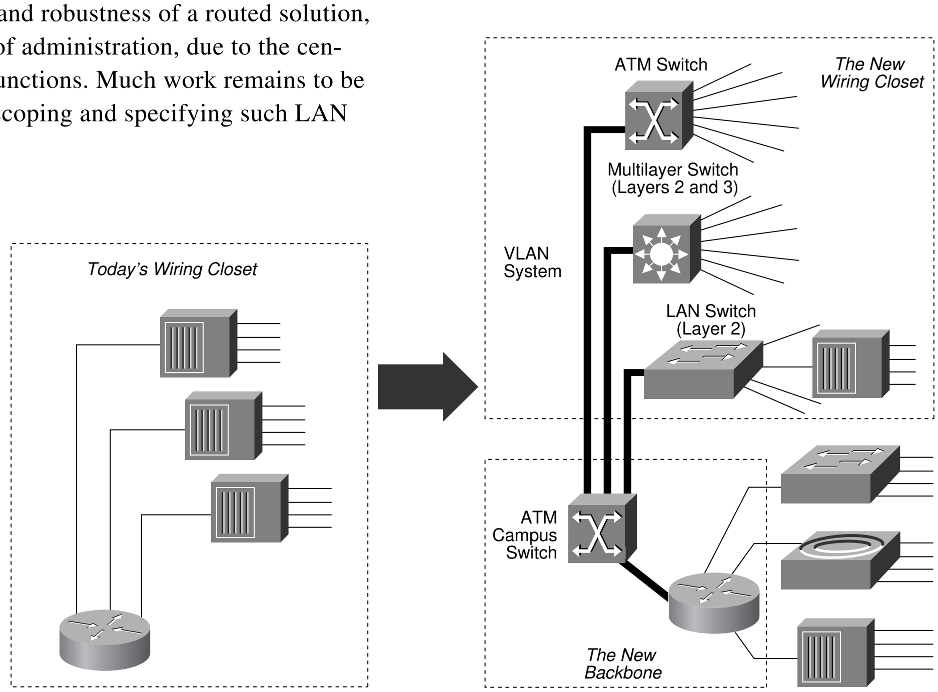



**Figure 35. The Evolution to Switched Internetworks**

than adequate for the vast majority of needs, while still preserving their existing installed based of desktop protocols and NICs. Such LAN switches will support virtual LAN services to facilitate network administration and control. In the first instance, such virtual LANs will be built using the LANE protocols and will interface to the first generation of Layer 2 LAN switches. In the future, as more sophisticated MultiLayer Switches are deployed, protocols like MPOA will gradually supplant LANE.

This evolution will likely be driven by the evolution of existing network layer protocols, as they acquire greater QoS support and interface more effectively to ATM. Native mode protocol support will be important in this evolution; in particular, ATM hosts and routers will likely use such protocols as 1577, and NHRP in addition to LANE. Over time, it is hoped that these will evolve into a common protocol like MPOA.

Despite popular misconceptions to the contrary, multiprotocol routers will still be needed, and will play a number of important roles, within such networks. First, given that most networks are, and will remain, non-ATM for the foreseeable future, they will be used to allow for the interconnection of such networks with newer ATM-based devices. Second, they will be used for virtual LAN interconnection. As discussed previously, ATM routers are critical for internetworking between multiple ELANs. They are also necessary for interconnection between the many different types of virtual LAN protocols like LANE, 1577, NHRP, MPOA, and so on—that are currently being developed. Over time, such routers may end up being distributed to a combination of MultiLayer switches and Route Servers, but the internetworking function will remain, albeit distributed throughout the network.

As discussed previously, routers will also remain important until and unless the ATM firewall problem is solved, and may also be used to provide a local connectionless service, as discussed above. Finally, routers may also be viewed as natural platforms upon which to deploy the many servers (such as LESs or NHRP servers) used with ATM protocols, since routers are high availability, high performance systems. Important synergies could also be drawn between such server functions and the general network state and filtering operations supported by routers today.

While the role of particular physical network elements, such as multiprotocol routers, and ATM and LAN switches, will change as the evolution to switched internetworks proceed, what will remain constant will be the complex software infrastructure that will overlay and link all of these elements. This infrastructure will provide a common service interface, across multiple types of network technology, while facilitating the integration of existing networks, and allowing for the scalable deployment of newer switched technologies.

While the plethora of protocols described here may seem daunting, they reflect the fundamental complexity of the task that is involved in building such large scale, ATM-based switched internetworks. Contrary to some earlier expectations, it is clear that the evolution to ATM will be complex, and will require protocols of the sophistication of those described here, both to exploit the benefits of ATM, and to enable a smooth evolution from existing networks. The long term success of particular ATM vendors—and of the users who partner with them—will hence be at least as much a function of their capability to deliver this evolving software infrastructure, as it will be a function of their particular switch platforms.

## 10.0 References


[Alles1] Alles, A., et al. “*ATM_Forum 94-0568: Proposal for a Phase 0 P-NNI Protocol,”* June 1994.

[Alles2] Alles, A. and Finn, N.—“*ATM Forum 95-0170: Future Topics for LANE SWG,* January 1995.

[Alles3] Alles, A. and Traina, P—“*ATM Forum 94-0803: Proposal for a Work Effort for a Layer 3 Route Distribution Protocol,* September 1994.

[Alles4] Alles, A. and Finn, N.—“*ATM Forum 94-1206: Architectural Overview for a Layer 3 Route Distribution/Query Protocol,* November 1994.

[Armitage] Armitage, G.—“*Support for Multicast over UNI 3.1 Based ATM Networks,* Internet Draft, April 1995.

[Atkinson] Atkinson, R.—*RFC 1626: Default IP MTU for use over ATM AAL 5,* May 1994.

[Benham] Benham, D. and Finn, N.—*“ATM Forum 95-0415: Strawman MPOA Query/Response Content”*, April 1995.

[Borman] Borman, D., et al—*RFC 1323: TCP Extensions for High Performance,* May 1992.

[Bottorff] Bottorff, P.—*IPX Interworking over ATM,* Engineering Conference Notes, Networld+InterOp Engineering Conference, May 1994.

[Braden1] Braden, R., Clark, D. and Shenker, S.—*“RFC 1633: Integrated Service in the Internet Architecture: An Overview,”* July 1994.

[Braden2] Braden, R.—*“RFC 1122—STD 3: Requirement for Internet Hosts—Communication Layers,”* October 1989.

[Braden3] Braden, R., et al—*“RFC 1620: Internet Architecture Extensions for Shared Media,”* May 1994.

[Bradley] Bradley, T., et al—*“RFC 1490: Multiprotocol Interconnect Over Frame Relay,”* July 1993.

[Bradner] Bradner, S. and Mankin, A.—*“RFC 1752: The Recommendation for the IP Next Generation Protocol,”* January 1995.

[Brown] Brown, C.—*“ATM Forum 95-0004r1: Scope and Requirements for Multiprotocol Sub-Working Group”*, February 1995.

[Callon1] Callon, R.—*“ATM Forum 94-0789: Integrated P-NNI for Multiprotocol Routing,”* September 1994.

[Callon2] Callon, R.—*“ATM Forum 94-1093: Hooks in PNNI to Support Integrated Routing,”* November 1994.

[Cansever] Canserver, D.—“*NHRP Protocol Applicability Statement,”* Internet Draft, March 1995.

[Cherukuri] Cherukuri, R. ed.—*“ATM Forum/95-0221R1: Draft PNNI Signaling*”, March 1995.

[Cisco] Cisco Systems—*“CiscoFusion: An Architecture for Switched Internetworks,”* March 1995.

[Cole] Cole, R.G. et al —*“IP over ATM: A Framework Document,”* Internet Draft, April 1995.

[Deering1] Deering, S.—*“RFC 1112: Host Extensions for IP Multicasting,”* August 1992.

[Deering2] Deering, S., et al—*“Protocol Independent Multicast (PIM): Motivation and Architecture,”* Internet Draft, January 1995.

[Dickie] Dickie, M.—*“Routing in Today’s Internetworks,”* 1994, Van Nostrand Reinhold.

[Droms] Droms, R.—*“RFC 1541: The Dynamic Host Configuration Protocol”*, October 1993.

[Fink] Fink, B.—*“IP/ATM Integrated Routing & Addressing (IRA) Model”*, Internet Draft, March 1995.

[Finn] Finn, N. and McCloghrie, K.—*“ATM Forum 95-0352: LAN Emulation and MPOA”*, March 1995.

[Forum1] ATM Forum—*“ATM User-Network Interface Specification Version 3.1,”* ATM Forum Specification, September 1994. This will shortly be available from Prentice Hall.

[Forum2] ATM Forum—*“ATM User-Network Interface Specification Version 3.0,”* ATM Forum Specification, September 1993, Prentice Hall.

[Forum3] ATM Forum—*“ATM Forum 94-1018R2: Draft of UNI Signaling 4.0,”* April 1995.

[Forum4] ATM Forum—*“ATM_Forum 94-180R5: A Draft of the B-ICI Specification Document Version 2.0,”* February 1995.

[Forum5] ATM Forum—*“ATM_Forum 94-0471R7: P-NNI Draft Specification,”* March 1995.

[Forum6] ATM Forum—*“Interim Inter-Switch Signaling Protocol,”* ATM Forum Specification, February 1995.

[Forum7] ATM Forum—*“LAN Emulation Over ATM Specification—Version 1,”* ATM Forum Specification, February 1995.

[Forum8] ATM Forum—*“ATM Forum 94-0996: Frame Relay Forum FR/ATM Service Interworking Implementation Agreement,”* November 1994.

[Forum9] ATM Forum—*“ATM Forum 95-0013R2: Draft Version 3.0 of ATM Forum Traffic Management Specification Version 4.0,”* April 1995.

[Forum10] ATM Forum—*"ATM Forum 94-0730R4: QoS Baseline Document,"* March 1995.

[Grossman] Grossman, D.—*"ATM_Forum 94-1190: Permanent VCCs in P-NNI,"* November 1994.

[Heinanen1] Heinanen, J.—*"RFC 1483: Multiprotocol Encapsulation over ATM Adaptation Layer 5,"* July 1993.

[Heinanen2] Heinanen, J.—*"RFC 1735: NBMA Address Resolution Protocol (NARP),"* December 1994.

[Hinden] Hinden, R.—*"IP Next Generation Overview,"* Internet Draft, October 1994.

[Hughes] Hughes, D. and Hooshmand, K.—*"ABR Stretches ATM Network Resources,"* Data Communications, April 1995.

[IEEE] IEEE—"*IEEE Std 802.1D–1990—IEEE Standards for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges*," IEEE, 1991.

[ISO] ISO/IEC 10038: ANSI/IEEE Std. 802.1D: *"Information Processing Systems—Local Area Networks—MAC Sublayer Interconnection (MAC Bridges)."*

[ITU1] ITU-T—*"ITU-T I.555—Frame Relaying Bearer Service Interworking, Com 13 R2-E,"* July 1993.

[ITU2] ITU-T—*"ITU-T I.364—Support of Broadband Connectionless Data Service on B-ISDN,"* March, 1993.

[Jain] Jain, R.—*"Congestion Control and Traffic Management in ATM Networks: Recent Advances and a Survey,"* January 1995, to be published in *"Computer Networks and ISDN Systems."*

[Katz] Katz, D. and Piscitello, D.—*"NBMA Next Hop Resolution Protocol (NHRP),"* Internet Draft, May 1995.

[LANQuest] LANQuest Labs—*"ATM Cell Congestion Loss Across Switch (CCLAS) Throughput Analysis (January 1995)"*, LANQuest Labs, January 1995.

[Laubach] Laubach, M.—*"RFC 1577: Classical IP and ARP over ATM,"* January 1994.

[McDysan] McDysan, D., Sphon, D.—*"ATM—Theory and Application,"* McGraw Hill, 1995.

[Minoli] Minoli, D. and Vitella, D.—*"ATM & Cell Relay Service for Corporate Environments,"* McGraw-Hill, 1994.

[Mogul] Mogul, J. and Deering, S.—*"RFC 1191: Path MTU Discovery,"* November 1994.

[Partridge1] Partridge, C. et al—*"RFC 1546: Host Anycasting Service,"* November 1993.

[Partridge2] Partridge, C.—*"RFC 1363: A Proposed Flow Specification,"* September 1992.

[Partridge3] Partridge, C.—*"Gigabit Networking,"* Addison-Wesley, 1994.

[Perez] Perez, M. et al—*RFC 1755: ATM Signaling Support for IP over ATM*, February 1995.

[Perkins1] Perkins, D. and Liaw, F.—*"ATM Forum 94-0935: Beyond Classical IP—Integrated IP and ATM Architecture Overview,"* September 1994.

[Perkins2] Perkins, D. and Liaw, F.—*"ATM Forum 94-0936: Beyond Classical IP—Integrated IP and ATM Protocol Specification,"* September 1994.

[Prycker] De Prycker, M.—*"Asynchronous Transfer Mode—Solution for Broadband ISDN, 2nd Edition,"* Ellis Horwood, 1993.

[Rekhter] Rekhter, Y., Kandlur, D.—*"IP Architecture Extensions for ATM,"* Internet Draft, January 1995.

[Schulzrinne] Schulzrinne, H. et al—*"RTP: A Transport Protocol for Real-Time Applications*," Internet Draft, March 1995.

[Swallow] Swallow, G.—*"PNNI: Weaving a Multivendor ATM Network,"* Data Communications, December 1994.

[Zhang] Zhang, L., et al—*"Resource ReSerVation Protocol (RSVP) — Version 1 Functional Specification,"* Internet Draft, March 1995.

### Reference Sources

- ATM Forum contributions are available only to Principal Members of the ATM Forum. Published ATM Forum specifications are available for purchase through the ATM Forum. Call the ATM Forum fax server at +1 (415) 525 0182, or send e-mail to *af-info@atmforum.com* for more details on ATM Forum membership.
- The UNI 3.0 specification is published by Prentice Hall and should be available at most technical bookstores.
- Information about Cisco products and services can be obtained from the Cisco Customer Information Online (CIO) WWW Home Page at *http://www.cisco.com/*. Cisco can be contacted at 1 (800) 553 6387, or through any Cisco sales office.

Details on obtaining Internet RFCs through anonymous FTP or e-mail may be obtained by sending e-mail to *rfc-info@ISI.EDU* with the message: *help: ways_to_get_rfcs*

- Internet Drafts are available by anonymous FTP. Internet-Drafts directories are located at:
  - — US East Coast: ds.internic.net
  - — US West Coast: ftp.isi.edu
  - — Europe: nic.nordu.net
  - — Pacific Rim: munnari.oz.au

## APPENDIX A: A SURVEY OF ATM TRAFFIC MANAGEMENT

One of the primary benefits of ATM networks is that they can provide users with a guaranteed Quality of Service (QoS). To do this, the user must inform the network, upon connection set-up, of both the expected nature of the traffic that will be sent along the connection, and of the type of quality of service that the connection requires. The former is described by a set of traffic parameters, while the latter is specified by a set of desired QoS parameters. The source node must inform the network of the traffic parameters and desired QoS for each direction of the requested connection upon initial set-up; these parameters may be different, however, in each direction of the connection.

ATM networks offer a specific set of service classes, and at connection set-up, the user must request a specific service class from the network for that connection. Service classes are used by ATM networks to differentiate between specific types of connections, each with a particular mix of traffic and QoS parameters, since such traffic may need to be differentiated within the network, for instance, by using priorities to allow for the requested behavior. The current set of QoS classes[63], which the Forum is defining for UNI 4.0 is as follows:

1. *Continuous Bit Rate [CBR]:* End systems would use CBR connection types to carry constant bit rate traffic with a fixed timing relationship between data samples, typically for circuit emulation.
2. *Variable Bit Rate—Real Time [VBR(RT)]:* The VBR(RT) service class is used for connections that carry variable bit rate traffic, in which there is a fixed timing relationship between samples; for instance, for such applications as variable bit rate video compression.
3. *Variable Bit Rate—Non-Real Time* [64]*[VBR(NRT)]:* The VBR(NRT) service class is used for connections that carry variable bit rate traffic in which there is no timing relationship between data samples, but a guarantee of QoS (on bandwidth or latency) is still required. Such a service class might be used for Frame Relay internetworking, in which the Committed Information Rate (CIR) of the Frame Relay connection is mapped into a bandwidth guarantee within the ATM network.
4. *Available Bit Rate [ABR]:* The ATM Forum is currently focusing its work on the ABR service ([Forum9], [Jain], [Hughes]). As with the VBR(NRT) service, ABR supports variable rate data transmissions and does not preserve any timing relationships between source and destination. Unlike the VBR(NRT) service, however, the ABR service does not provide any guaranteed bandwidth to the user. Rather, the network provides a "best effort" service, in which feedback (flow control mechanisms) is used to increase the bandwidth available to the user—the Allowed Cell Rate (ACR)—if the network is not congested and to reduce the bandwidth when there is congestion. Through such flow control mechanisms, the network can control the amount of traffic that it allows into the network, and minimize cell loss within the network due to congestion.

   The ATM Forum is currently working on a "rate based" mechanism for ABR congestion control, where Resource Management (RM) Cells or the explicit forward congestion indication (EFCI) bit within ATM cells are used to indicate the presence of congestion within the network to the source system. A specified traffic pacing algorithm, controlling the ACR, is used at the source to control the traffic rate into the network, based either upon the number of RM cells received with a congestion indication or an explicit rate indication from the network. Refer to [Forum9] for more details.

   ABR is designed to map to existing LAN protocols that opportunistically use as much bandwidth as is available from the network, but can either back off, or be buffered in the presence of congestion. ABR is hence ideal for carrying LAN traffic (for instance, using LAN Emulation) across ATM networks.

   The ABR service can optionally provide a guaranteed Minimum Cell Rate (MCR) for an ABR connection, but the exact nature of this guarantee is currently a matter of debate within the ATM Forum.
5. *Unspecified Bit Rate [UBR]:* The UBR service does not offer any service guarantees. The user is free to send any amount of data up to a specified maximum while the network makes no guarantees at all on the cell loss rate, delay, or delay variation that might be experienced. The UBR service is currently the best match to LAN protocols, given that the ABR specification has yet to be completed.

   As of the time of writing, it appeared that the ABR specification would not be completed until well into the second half of 1995. Deployment of ABR compliant equipment will likely take even longer. In the meantime, UBR is the only service currently available for data transport. Since UBR does not have any flow control mechanisms, however, to control or limit congestion, it will be important that ATM switches either implement pre-standard congestion control mechanisms, or support adequate buffering to minimize the probability of cell loss when multiple large data bursts are received concurrently at a switch, as might be expected, for instance, in a typical client-server environment [LANQuest].

There is no explicit priority field associated with ATM connection types, though, as will be discussed, such priorities are required within ATM switches. The only indication of

[63] The ABR and VBR(NRT) classes were not defined in UNI 3.1.

[64] As of the time of writing, it was not clear whether or not the VBR(NRT) service would actually be formally specified within UNI 4.0, since it is not defined by the ITU-T, and its use and utility was controversial within the ATM Forum.

relative priority within an ATM cell is the Cell Loss Priority (CLP) bit that is carried within the cell header; setting this bit to 1 (CLP=1) indicates that the cell may be dropped, in preference to cells with CLP=0. While this bit may be set by end systems, it is more likely to be set by the network, as described below.

Traffic sent along connections of any type are defined by a set of traffic parameters:

- Peak Cell Rate (PCR)
- Cell Delay Variation Tolerance (CDVT)
- Sustainable Cell Rate (SCR)
- Burst Tolerance (BT)
- Minimum Cell Rate (MCR), for ABR only

These parameters define an "envelope" around a traffic stream, but not all parameters are valid for all service classes. For CBR connections, for instance, only the PCR, which determines how often data samples are sent, and the CDVT, which determines how much jitter is tolerable for such samples, are relevant. For VBR connections, the SCR and BT together determine the long-term average cell rate and the size of the maximum burst of contiguous cells that can be transmitted. In the case of the ABR service, the PCR determines the maximum value of the Allowed Cell rate (ACR), which is dynamically controlled by the network, through congestion control mechanisms, to vary between the MCR and PCR.

When setting up a connection, the requesting node informs the network of the type of service required, the traffic parameters of the data flows in each direction of the connection, and the QoS requested for each direction. Together, these form the traffic descriptors for the connection. In UNI 3.0/3.1, the QoS requested for each direction is not explicitly specified. Instead, the network offers a number of specified QoS classes that correspond to some or all of the QoS service types. The network administration has the responsibility of ensuring that the network is configured such that each of the offered QoS classes provides levels of QoS appropriate for each QoS type. The ATM Forum decided, however, that this method was too ambiguous and replaced it in UNI 4.0 with explicit signaling of QoS parameters[65], desired values of which are requested at connection set-up time [Forum10].

The current set of QoS parameters consist of three delay parameters, and one dependability parameter. The three delay parameters are as follows:

- Peak-to-peak cell delay variation (CDV)
- Maximum cell transfer delay (Max CTD)
- Mean cell transfer delay (Mean CDV)

The dependability parameter is as follows:

- Cell Loss Ratio (CLR)

The former three parameters are treated as dynamic, additive metrics, and their expected values through the network will be cumulated in (UNI 4.0 and P-NNI) signaling requests, while the latter is considered to be a configured link and node attribute, which local CAC algorithms will strive to meet. Particular combinations of the CDV, Max CTD, Mean CTD and CLR (for CLP=0 streams only) parameters will be negotiable, depending upon the service class, between the end-system and the network, in UNI 4.0. As with the traffic

[65] UNI 4.0 signaling messages will carry both the QoS service classes and the explicit parameters, so that switches could operate on either, depending upon their own implementation.

**Table 1. Service Classes and Applicable Parameters**

| Attribute | ATM Layer Service Categories | | | | | Parameter |
|---|---|---|---|---|---|---|
| | CBR | VBN (RT) | VBR (NRT) | ABR | UBR | |
| **CLR** | Specified[1] | Specified[1] | Specified[1] | Specified[2] | Unspecified | QoS |
| **CTD and CDV** | CDV and Max CTD | CDV and Max CTD | Mean CTD Only | Unspecified[6] | Unspecified | QoS |
| **PCR and CDVT[5]** | Specified | Specified | Specified | Specified[4] | Specified[3] | Traffic |
| **SCR and BT[5]** | n/a | Specified | Specified | N/A | n/a | Traffic |
| **MCR** | n/a | n/a | n/a | Specified | n/a | Traffic |
| **Congestion Control** | No | No | No | Yes | No | |

1. For CBR and VBR the Cell Loss Ratio may be unspecified for CLP=1.
2. Minimized for sources that adjust cell flow in response to control information.
3. Not subject to CAC and UPC procedures and may use different value from section 3.6.2.4 of the UNI 3.1 specification [Forum1].
4. Represents the maximum rate at which the source can send as controlled by the control information.
5. These parameters are either explicitly or implicitly specified for PVCs or SVCs as defined in section 3.6.2.4.1 of the UNI 3.1/3.0 specifications.
6. The objective of the service is that the network does not excessively delay the admitted cells. Requirement for explicit specification of the CTD and CDV is for further study.

parameters, not all QoS parameters apply to all service classes. **Table 1** summarizes the traffic parameters and QoS parameters applicable to each of the QoS service classes.

An ATM connection that is set up with specified traffic descriptors constitutes a *traffic contract* between the user and the network. The network offers the type of guarantee[66] appropriate to the service class, as long as the user keeps the traffic on the connection within the envelope defined by the traffic parameters. The network can enforce the traffic contract by a mechanism known as *usage parameter control* (UPC), better known as traffic policing. UPC is a set of algorithms performed by an ATM switch on the receipt of cells within a connection that determine whether or not the cell stream is compliant with the traffic contract. The UNI 3.1 specification specified a “dual leaky bucket” algorithm for UPC[67] [Forum1].

In conceptual terms, the dual leaky bucket mechanism[68] can be best thought of as a means of pacing the transmission of cells along a link so that the traffic stream meets the specified PCR and CDVT, and optionally, the SCR and BT for the connection (for various combinations of CLP=0, CLP=1 and CLP=0+1 cell streams). The UNI 3.1 UPC mechanism measures cell arrivals as if they were generated by such a leaky bucket based ‘generic cell rate algorithm’ (GCRA). This does not necessarily mean that the cell transmitted on the connection needs to be so paced. Any type of traffic shaping can be used, as long as the traffic “envelope” fits within the traffic contract parameters. In practice, however, traffic sent across ATM links that are controlled by UPC are sometimes actually shaped by using a leaky bucket algorithm and the requested traffic parameters, which ensures that cells will not be inadvertently marked as non-conformant. Traffic shaping can also help control and reduce congestion within a network - for instance, by limiting the peak rate of a connection to that of the slowest link along the path.

Upon the detection of a non-conformant cell, a switch can choose to either selectively discard the cell, or, if local resources and policies permit, to tag the cell as non-conformant by setting its CLP bit to 1. The cell would then be more likely to be discarded further within the ATM network if further congestion is experienced. UPC is primarily designed to be used across UNI, since passage through ATM switches will change the shape of the traffic stream[69] due to buffering delays and so on. UPC is likely to be used across public UNI, however, since public ATM networks will likely base their tariffs on traffic usage. This may require ATM switches that are connected to public UNI to reshape the traffic sent across public UNI.

As described in Section 4.0, the ATM routing protocols performed by ATM switches use the traffic descriptors associated with a signaling request to both route the connection appropriately to meet the traffic guarantees, and to control connection admission, which ensures that establishing a new connection will not adversely affect established connections. To support multiple traffic classes, ATM switches internally generally must implement a mechanism for isolating the traffic flows of particular connection types from each other. For instance, the switch may allocate different priority levels to the different service classes, so that the cells of some connection types gain preferential access to scarce resources - typically CBR connections receive high priority to minimize the amount of latency and jitter experienced by the cells on such connections.

[66] In UNI 3.0/3.1, the traffic parameters and requested QoS for a connection cannot be negotiated at set-up, or changed over the lifetime of the connection. UNI 4.0 will support connection QoS negotiation; how this will be supported within P-NNI is for future study.

[67] The UNI 3.1 UPC algorithm applies only to CBR and VBR connection types. No UPC mechanism is specified for UBR connections. The Forum is currently considering UPC mechanisms for the ABR service.

[68] Strictly speaking, the dual leaky bucket UPC mechanism models traffic as if it were paced by a single leaky bucket—the size of which determines the CDVT, and is emptied at the PCR. This leaky bucket is then fed by a token bucket that is emptied at the SCR; the size of the token bucket determines the MBS. Refer to [Partridge3] for a detailed discussion of leaky bucket traffic shaping algorithms.

[69] The requested QOS, however, must be met end-to-end.

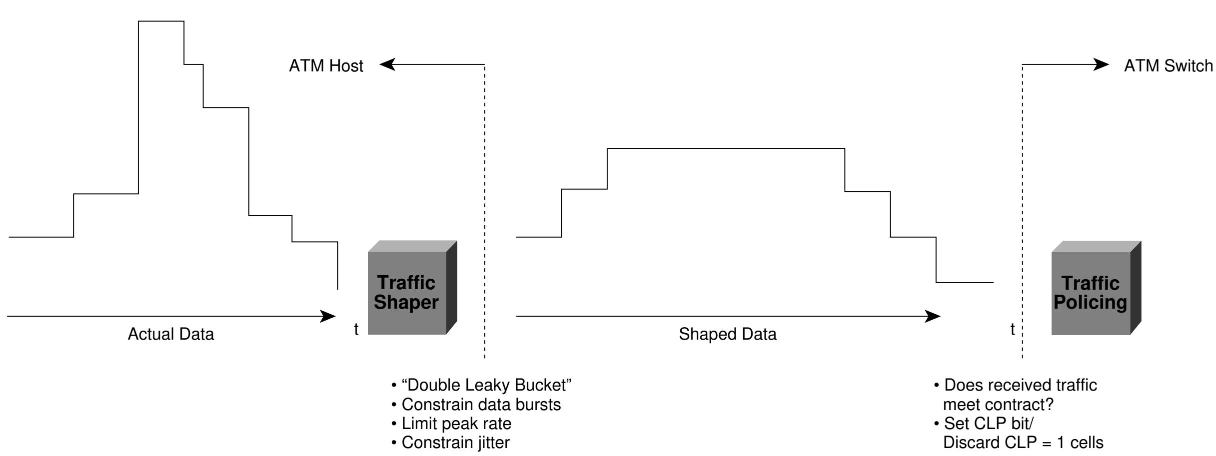



**Figure 36. Traffic Shaping and Policing (UPC)**

## Appendix B: Status of Key ATM Standards and Specifications

Most of the key specifications and standards for private ATM networks are being developed at the ATM Forum and the Internet Engineering Task Force (IETF). The former is strictly an "implementer's agreement" body, clarifying the use of standards developed at other ATM standards bodies, such as the ITU-T and the ANSI T1S1 Committee. In practice, the ATM Forum has considerable extended such standards for private network specific requirements, and has created entirely new specifications, such as LAN Emulation and the P-NNI protocols. The ATM Forum specifications can hence be considered the de facto standards for private network ATM deployment.

The IETF has focused primarily, as might be expected, on aspects of IP interworking over ATM, since most other layer 3 protocols (e.g. IPX, Appletalk) are proprietary. The work of the IETF has been very influential, however, and, as noted in the paper, serve as models for the work of the ATM Forum (in particular, for the Multiprotocol over ATM group).

We list below some of the key completed and pending specifications from the ATM Forum and the IETF. Expected completion dates for pending specifications are naturally best guesses only, as of the time of writing. Refer to Section 10.0 for information on how to obtain the latest drafts of such specifications. Note, however, that the deployment of completed standards will typically lag their final specification, due to necessary development schedules, many of which cannot commence prior to the finalization of the standards.

### B.1 Completed Specifications—ATM Forum

1. UNI 3.0

   *Contents:* Physical layer, ATM layer, OAM cell operation, ILMI, UNI signaling.

2. UNI 3.1

   *Contents:* Bug fixes to UNI 3.0, alignment with completed ITU-T SSCOP and signaling standards.

3. LANE Phase 1

   *Contents:* LUNI protocol

4. IISP

   *Contents:* UNI 3.0/3.1 based static routing NNI protocol

### B.2 Completed Specifications—IETF

1. RFC 1483

   *Contents:* Multiprotocol Encapsulation

2. RFC 1577

   *Contents:* Classical IP Over ATM protocol

3. RFC 1626

   *Contents:* Default MTU for Classical IP

4. RFC 1755

   *Contents:* Signaling guidelines for Classical IP

### B.3 Pending Specifications—ATM Forum

1. P-NNI Phase 1

   *Contents:* QoS based NNI routing, hierarchical network model. Expected Completion Date: Q3 1995

2. ABR Congestion Control

   *Contents:* Best effort traffic class and rate based congestion control mechanism. Expected Completion Date: Q3/Q4 1995

3. UNI 4.0 Signaling

   *Contents:* ABR signaling, leaf initiated joins, QoS negotiation, VP signaling, proxy signaling etc. Expected Completion Date: Q3/Q4 1995

4. MPOA

   *Contents:* Multiprotocol transport over ATM. Expected Completion Date: Q1/Q2 1996

5. LANE Phase 2

   *Contents:* L-NNI specification for redundant servers. Expected Completion Date: Q1/Q2 1996

### B.4 Pending Specifications—IETF

1. NHRP

   *Contents:* Cut through routing extensions to Classical IP model. Expected Completion Date: Q2/Q3 1995

2. Multicast Support in 1577

   *Contents:* Multicast registration services in Classical IP. Expected Completion Date: Q2/Q3 1995

3. IPv6 (IPng)

   *Contents:* Family of specifications for complete IPv6 protocol. Expected Completion Date: Q4 1995

4. RSVP

   *Contents:* Resource reservation protocol for IP. Expected Completion Date: Q3 1995

5. PIM

   *Contents:* Protocol independent multicast protocol for IP. Expected Completion Date: Q3 1995

**Corporate Headquarters**
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

Cisco Systems has over 100 sales offices worldwide. Call the company's corporate headquarters (California, USA) at 408 526-4000 to contact your local account representative or, in North America, call 800 553-NETS (6387).

**Europe**
**European Headquarters**
Cisco Systems Europe s.a.r.l.
Z.A. de Courtaboeuf
16 avenue du Quebec
91961 Les Ulis Cedex
France
Tel: 33 1 6918 61 00
Fax: 33 1 6928 83 26

**Austria**
Cisco Systems Austria GmbH
World Trade Center
A-1300 Vienna Airport
Austria
Tel: 43 1 71110 6233
Fax: 43 1 71110 6017

**Belgium**
Cisco Systems Bruxelles
Complex Antares
71 avenue des Pleiades
1200 Brussels
Belgium
Tel: 32 2 778 42 00
Fax: 32 2 778 43 00

**Denmark**
Cisco Systems
Larsbjoernsstraede 3
1454 Copenhagen K
Denmark
Tel: 45 33 37 71 57
Fax: 45 33 37 71 53

**Germany**
Cisco Systems GmbH
Max-Planck-Strasse 7
85716 Unterschleissheim
Germany
Tel: 49 89 32 15070
Fax: 49 89 32 150710

**Italy**
Cisco Systems Italy
Via Turati 28
20121 Milan
Italy
Tel: 39 2 62 726 43
Fax: 39 2 62 729 13

**The Netherlands**
Cisco Systems
Stephensonweg 8
4207 HB Gorinchem
The Netherlands
Tel: 31 18 30 22988
Fax: 31 18 30 22404

**Norway**
Cisco Systems
Holmens Gate 4
0250 Oslo
Norway
Tel: 47 22 83 06 31
Fax: 47 22 83 22 12

**South Africa**
Cisco Systems South Africa
Prestige Business Center
Sloane Park 90 Grayston Drive
2152 Sandton
South Africa
Tel: 27 11 784 0414
Fax: 27 11 784 0519

**Spain**
Cisco Systems Spain
Paseo de la Castellana, 141, pl 18
28046 Madrid
Spain
Tel: 34 1 57 203 60
Fax: 34 1 57 045 99

**Sweden**
Cisco Systems
Arstaangsvagen 13
11760 Stockholm
Sweden
Tel: 46 8 681 41 60
Fax: 46 8 19 04 24

**Switzerland**
Cisco Systems Switzerland
Grossrietstrasse 7
CH-8606 Naenikon/ZH
Switzerland
Tel: 41 1 905 20 50
Fax: 41 1 941 50 60

**United Arab Emirates**
Cisco Systems (Middle East)
Dubai World Trade Center, Level-7
P.O. Box 9204
Dubai, U.A.E.
Tel: 971 4 313712
Fax: 971 4 313493

**United Kingdom**
Cisco Systems Ltd.
4 New Square
Bedfont Lakes
Feltham, Middlesex TW14 8HA
United Kingdom
Tel: 44 81 818 1400
Fax: 44 81 893 2824

**Intercontinental and Latin American Headquarters**
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
Tel: 408 526-7660
Fax: 408 526-4646

**Asia**
Cisco Systems (HK) Ltd
Suite 1009, Great Eagle Centre
23 Harbour Road
Wanchai, Hong Kong
Tel: 852 2583 9110
Fax: 852 2824 9528

Cisco Systems (HK) Ltd
Beijing Office
Room 821/822, Jing Guang Centre
Hu Jia Lou, Chao Yang Qu
Beijing 100020 P.R.C.
Tel: 86 1 501 8888 x821
Fax: 86 1 501 4531

Cisco Systems (HK) Ltd
New Delhi Liaison Office
Suite 119, Hyatt Regency Delhi
Bhikaiji Cama Place
Ring Road
New Delhi 110066, India
Tel: 91 11 688 1234
Fax: 91 11 688 6833

Cisco Systems Korea
27th Fl., Korea World Trade Center
159, Samsung-dong, Kangnam-ku
Seoul, 135-729, Korea
Tel: 82 2 551 2730
Fax: 82 2 551 2720

Cisco Systems (HK) Ltd
Kuala Lumpur Office
Level 5, Wisma Goldhill
67 Jalan Raja Chulan
50200 Kuala Lumpur, Malaysia
Tel: 60 3 202 1122
Fax: 60 3 202 1822

Cisco Systems (HK) Ltd
Singapore Office
Shell Tower, Level 37
50 Raffles Place
Singapore 0104
Tel: 65 320 8398
Fax: 65 320 8307

Cisco Systems (HK) Ltd
Taipei Office
4F, 25 Tunhua South Road, Section 1
Taipei, Taiwan
Tel: 886 2 577 4352
Fax: 886 2 577 0248

Cisco Systems (HK) Ltd
Bangkok Office
23rd Floor, CP Tower
313 Silom Road
Bangkok 10500, Thailand
Tel: 66 2 231-0600
Fax: 66 2 231-0448

**Argentina**
Cisco Systems Argentina
Av. del Libertador 602 Piso 5
(1001) Capital Federal
Buenos Aires, Argentina
Tel: 54 1 814 1391
Fax: 54 1 814 1846

**Australia**
Cisco Systems Australia Pty., Ltd.
Level 17
99 Walker Street
P.O. Box 469
North Sydney, NSW 2060
Australia
Tel: 61 2 957 4944
Fax: 61 2 957 4077

**Brazil**
Cisco Systems Do Brasil
Rua Helena 218, 10th Floor
CJ 1004-1005
Vila Olimpia - CEP 04552-050
Sao Paulo - SP Brazil
Tel: 55 11 822-5413
Tel/Fax: 55 11 853-3104

**Mexico**
Cisco Systems de México, S.A.
Ejercito Nacional
No. 926, Piso 3
Colonia Polanco
Mexico, D.F. C.P. 11560
Tel: 525 328 7600
Fax: 525 328 7699

**New Zealand**
Cisco Systems New Zealand
Level 16, ASB Bank Centre
135 Albert Street
P.O. Box 6624
Auckland, New Zealand
Tel: 64 9 358 3776
Fax: 64 9 358 4442

**Japanese Headquarters**
Nihon Cisco Systems K.K.
Seito Kaikan 4F
5, Sanbancho, Chiyoda-ku
Tokyo 102, Japan
Tel: 81 3 5211 2800
Fax: 81 3 5211 2810

**North America**
**Canada**
Cisco Systems Canada Limited
150 King Street West
Suite 1707
Toronto, Ontario M5H 1J9
Canada
Tel: 416 217-8000
Fax: 416 217-8099




Lit. #030802 4/95 GP