Connection Management for a Prototype Fast Packet ATM B-ISDN Network

Mike Gaddis, Rick Bubenik, John DeHart

Applied Research Laboratory
Department of Computer Science
Washington University
St. Louis, Missouri 63130

ABSTRACT

We describe a call/connection model for multipoint connections in an ATM/B-ISDN network, along with an access protocol that allows clients of the network to create, manipulate and destroy these connections. One-to-one, one-to-many and many-to-many communications are supported by this model. The underlying ATM/B-ISDN network is composed of a collection of interconnected high-speed packet switches or *nodes*. The implementation of our protocol requires *connection management* software at each node to control the operation of the underlying switch and to implement the higher level network functionality that binds the nodes together into a cohesive network. We discuss the preliminary design of the software architecture intended for this purpose.

1 Introduction

We describe the multipoint connection/-call model and the access protocol used to create, manipulate and destroy connections in the prototype ATM B-ISDN (Asynchronous Transfer Mode, Broadband-Integrated Services Digital Network) network under development at the Applied Research Laboratory (ARL) at Washington University in St. Louis, Missouri. This research is part of a cooperative research agreement (called the Fast Packet Project (FPP)) between Washington University, Southwestern Bell Telephone Company (SWBT), SBC-Technology Resources, Incorporated (SBC-TRI) and NEC America. The goal of the project is to establish an ATM prototype network linking the laboratories of SWBT, SBC-TRI and Washington University for the purpose of exploring potential applications of B-ISDN technology. Initial applications focus on multi-media workstations, sub--network interfacing (Ethernet and FDDI) and high bandwidth medical applications (tele-radiology) [1].

The characteristics of ATM/B-ISDN networks that differentiate our work from previous work are the relatively high bandwidth of these networks (typically 100 Mbps to 2.4+ Gbps), the connection-oriented nature of the packet switches necessary to support this bandwidth [4, 6, 7, 8, 11, 12, 13], and the multicast connection capability inherent in our prototype switch [11, 12]. These characteristics allow for entertainment broadcast video (including HDTV), multicast video teleconferencing, video lectures and a variety of next generation high-speed computer applications. Our call model and access protocol are designed to support all of these types of applications.

In the remainder of this paper, we briefly describe the components of a B-ISDN network and the prototype ATM switch under development at ARL to provide a basis for understanding the requirements of connection management in B-ISDN networks. We then describe the protocol architecture and the network access protocol of our connection management system. After showing an example demonstrating client interaction with the network, we describe the details of the software architecture used to implement the system. Finally, we list future directions for our prototype system and describe mechanisms for porting the system to other switch architectures and integration with other B-ISDN protocols.

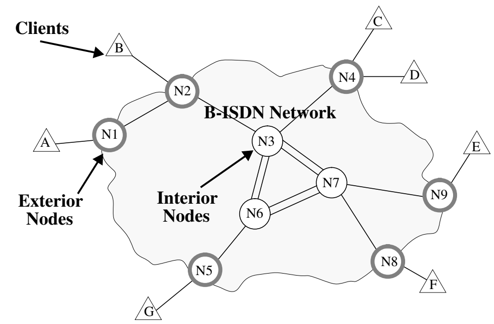

Figure 1. Example B-ISDN Network

2 Broadcast Packet Network Overview

An example B-ISDN network is shown in Figure 1. The network consists of *interior nodes* (nodes that interface to other nodes only), *exterior nodes* (nodes that interface to clients) and network *clients*, all interconnected by *fiber optic links* (FOLs). A client signals the network to set up connections with other clients by sending control messages to exterior nodes, which process and implement the requested client operations. If the request requires the services of other nodes, then signaling messages are exchanged between nodes (exterior and interior) in order to satisfy the request. The signaling messages used by the client are part of a *client-network interface* access protocol. A goal of an access protocol is to hide the complexity of the network so that it appears to the client as a single monolithic entity.

Each node in the network contains an ATM packet switch. The switch architecture used in our prototype *Broadcast Packet Network* (BPN) [11, 12] contains a *Copy Network* (CN) concatenated to a *Routing Network* (RN) (Figure 2). ATM cells (packets) enter the switch on the left. Multicast cells, destined for several locations, are replicated by the CN, then routed to the appropriate destination by the RN. Point--to-point cells follow an arbitrary path through the CN, then are routed by the RN. Cells leave the switch on the right, where they traverse fiber optic links to other switches or clients.

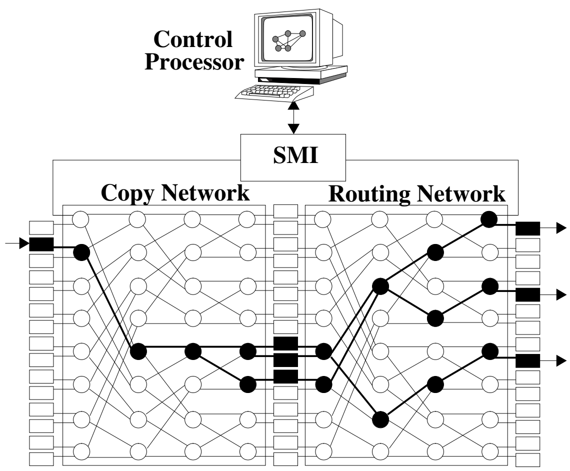

Figure 2. Architecture of a BPN Switch Demonstrating Multicast Routing

The switch is controlled by the control processor (CP) connected to the switch via a switch Module Interface (SMI). The CP configures the switch hardware to route incoming cells to the appropriate outgoing links by modifying tables within the switch, thus establishing connections. The SM routes control cells from clients or other nodes (as distinguished by the header) to the CP via port 0.

The only distinction between an exterior node and an interior one is the software suite implemented in the exterior node CPs to handle client access requests.

3 The Call Model

The *call model* governs the manner in which communication channels are created, manipulated and destroyed. We describe the model, along with Addressing and Resource Allocation and Management.

3.1 The Call/Connection Model

A network client establishes communication paths by creating and manipulating *calls*. A call is, in effect, a distributed object maintained by the network. Initially, a call has a single communication channel which we call a *connection*. The owner of a call (the creator) can add connections to the call. This provides for multiple connections within a single call, where, for example, one connection might be used for video and another for voice. Initially, the owner is the only *endpoint* in the call. Additional endpoints may be added by: 1) invitation from the owner, where the invited party has the option of refusing the invitation, 2) request from a client not currently in the call to be added, where the owner has the option of denying the request, or 3) request from a third party, not necessarily in the call, to add a client, where both the owner and the client being added have the option to refuse.

The creator of a call defines the accessibility of the call to new endpoints. It may be defined as: 1) CLOSED, where no new endpoints may be added except at the invitation of the owner, 2) QUERY OWNER, where the owner has the option of refusing any attempt to add an endpoint, or 3) OPEN, where any endpoint may be added without the owner having the option to refuse. A separate call characteristic is defined which allows the owner to be notified when an endpoint is added.

A connection within a call is defined to be bi-directional, uni-directional or non-directional at each endpoint participating in the call. A bi-directional connection is used for receiving and transmitting. A uni-directional connection is used for either receiving or transmitting but not both. A non-directional connection at an endpoint is a connection on which the endpoint can neither transmit nor receive even though it is participating in the call to which the connection belongs.

3.2 Client Addressing

Network addresses consist of the (CCITT E.164) ISDN international address. It is a five part address that consists of: an International Code (5 BCD digits), Area Code (3 BCD digits), Local Exchange (3 BCD digits), Local Extension (4 BCD digits) and a four byte unformatted ISDN Sub-Address field. The total address field contains 8 bytes for the BCD numbers and 4 bytes for the sub-address field, yielding a total address field of 12 bytes (Figure 3).

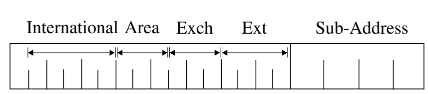

Figure 3. ISDN Address Format

3.3 Call Element Entities at the Client to Network Interface

A client connected to the network may participate in multiple (disjoint) calls and may have more than one instance of a call mapped to his access point. Furthermore, a client interface may have more than one network address associated with it. The call model must discriminate between these call elements.

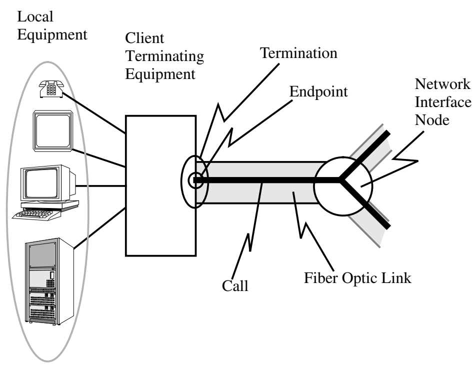

Figure 4. Client/Network Interface

Figure 4 shows components in the client to network interface. The local equipment can be addressed by different ISDN addresses or using the same address with a different ISDN sub-address field. The elements that the network uses to identify calls are *terminations*, *global call identifiers*, *local call identifiers* and *endpoints*. A termination is the end of a fiber optic link at the first piece of client premises equipment. If multiple addresses are associated with a physical termination then the network considers that port to have multiple logical terminations. A *local call identifier* is assigned to a call when the owner creates it. The *global call identifier* (call ID) is the concatenation of this local call identifier and the call owner's termination ID. The call ID uniquely identifies the call throughout the network. An endpoint is an instance of a call within a termination. A call associated with a termination may have multiple endpoints but only one call identifier.

3.4 Resource Allocation/Management

A connection setup request by a call owner explicitly specifies the bandwidth requirements. This information allows the network to make performance guarantees to the clients. The client specifies the bandwidth requirements as integral units of: (1) maximum bandwidth, (2) average bandwidth and (3) burst length. The maximum and average bandwidth are specified as measures of ATM cells per second. The Burst Length is the number of ATM cells in a maximum burst.

This method of bandwidth allocation does not force clients to select a bandwidth from a set of predefined bandwidth or service classes. Rather, all clients are free to define the resource requirements as an integral increment of cells per second. Building service classes on top of this continuous model is straightforward.

4 Protocol Architecture

When we embarked on this project we decided against extending current ISDN signaling protocols (for example Q.931, [3] or SS7), opting for a fresh approach that was not restrained by a pre-existing telecommunications framework. This allowed us to address the problems associated with multipoint connection oriented protocols, a problem not fully addressed in existing protocols and standards.

The Connection Management Access Protocol (CMAP) allows clients to create and maintain multipoint connections. It is not specific to our BPN prototype network, but rather is adaptable to many ATM switch architectures that support multipoint communications.

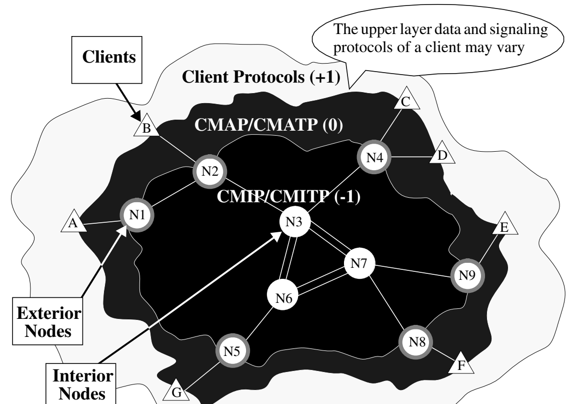

Figure 5. Layered View of the Protocol Architecture

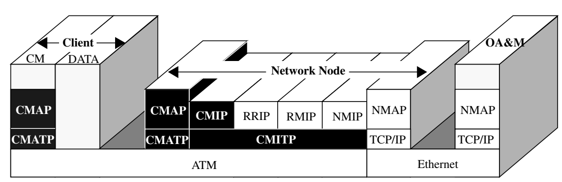

Figure 6. Connection Management Protocol Diagram

The protocol architecture (Figures 5 & 6) consists of a client-network interface called CMAP and an internal protocol, Connection Management Internal Protocol (CMIP), that allows CPs (Control Processors) to communicate with one another. CMAP provides relatively high-level network commands that clients use to setup, maintain and tear down calls with other network clients. CMIP is used by CPs to implement the CMAP commands while shielding clients from most of the details required to realize the call manipulation requests.

Many CMAP and CMIP requests require more data than can be placed in one ATM payload (48 bytes). For this we provide transport protocols called CMAP transport (CMATP) and CMIP transport (CMITP). These protocols implement fragmentation/reassembly and provide reliable delivery mechanisms. We also provide peer to peer communication over CMITP for Routing (RR), Resource Management (RM) and Operations, Administration and Maintenance (OA&M). We briefly cover CMAP in Section 5.

Network Management (or OA&M) is implemented via the Network Management Access Protocol (NMAP) and its internal counterpart, the Network Management Internal Protocol (NMIP). In BPN, network managers gain access to the network over Ethernet (using TCP/IP) and can send requests to all CPs using NMIP.

5 The Connection Management Access Protocol (CMAP)

The development of a multipoint B-ISDN signaling protocol began at Washington University in 1986 with the work of Turner and Haserodt [5]. At the start of the Fast Packet Project, we decided to extend the previous work and implement CMAP in a manner that allows clients to send data and signaling messages (for connection setup) over the same access link, rather than require separate links for these purposes. Clients view the network as a monolithic entity, capable of connecting to any client addressable from the network. We draw no distinction between clients——a multi--media workstation shares the same interface and connection protocol as an Ethernet or FDDI gateway.

Due to space limitations, the details of CMAP (and CMIP) are not covered here, but can be found elsewhere [2]. We briefly introduce the list of CMAP commands, then give a simple example that illustrates client-network interaction.

5.1 Operations

A brief description of each CMAP operation appears in Table 1. The commands listed in Table 2 are a sampling of CMAP operations that will not be implemented in the prototype but are under review for inclusion in the protocol. Operations are organized into four classes: Commands, Prompts, Notifications and Queries. A Command is an operation that a client uses to request the network to do something. Commands require two messages. A Prompt is an operation that the network uses to give a client the opportunity to do something. Prompts require three messages. A Query may be used by either the network or a client to request some information from the other. Queries require two messages. A Notification is used by the network to inform the client of a change that has occurred. Notifications require one message.

TABLE 1. CMAP Operations

Name	Description
***Commands*:**	
open_call	establish a call between client and network
close_call	terminate a call
add_EP	add an endpoint to a call
drop_EP	delete an endpoint from a call
open_con	add a new connection to a call
close_con	remove a connection from a call
mod_call	modify call parameters
mod_con	modify connection characteristics
mod_EP	modify endpoint permissions
client_reset	inform network of a client reset
***Prompts*:**	
invite	invite endpoint to join a call
***Queries*:**	
check_join	give owner option to grant or deny access
***Notifications*:**	
announce_EP	inform owner that a new endpoint has joined
drop_from_call	inform endpoint that it is being dropped from a call
announce_drop	inform owner that an endpoint has dropped out of a call
announce_change	inform an endpoint that a change has taken place in a call it is part of
network_reset	inform client that a network reset has occurred

TABLE 2. CMAP Extensions

Name	Description
***Commands*:**	
transfer	allow an endpoint to add another endpoint and drop out in one operation
change_owner	transfer ownership to another endpoint
trace	allow endpoint to query the network for information about a call
concatenate	allow two connections of different calls to be combined, supports strong-multicast [10]

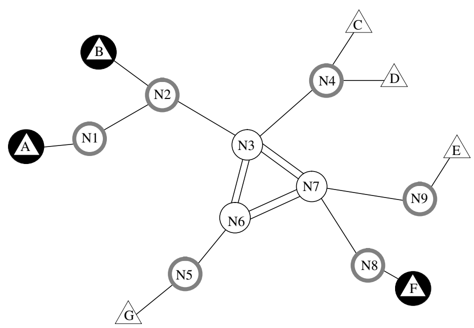

Figure 7. Sample ATM Network

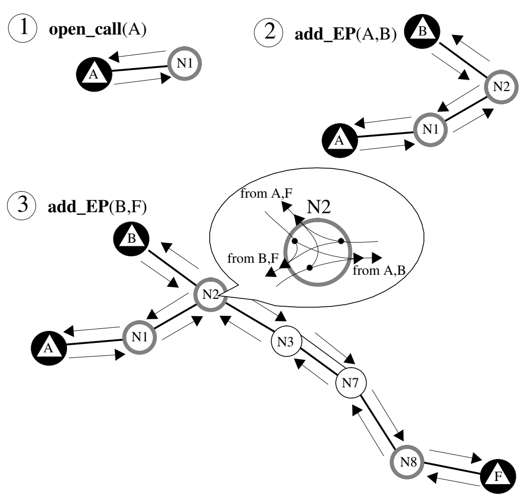

Figure 8. Call Topology After:
(1) open_call(A),
(2) add_EP(A,B) and
(3) add_EP(B,F) Requests

5.2 Connection Setup Example

Consider the network of Figure 7. If client A (the owner) wants to set up a multipoint bi-directional call between himself, B and F, he initiates the call (and is designated as the owner) by issuing the ***open_call*** CMAP command. Clients B and F are added to the call through ***add_ep*** commands that can be initiated by clients A, B or C. Figure 8 shows the progression of the call and the resultant call topology after each request has been processed. Each ***add_EP*** request results in an invitation (***invite***) to the client being added, where the client can either accept or refuse the connection.

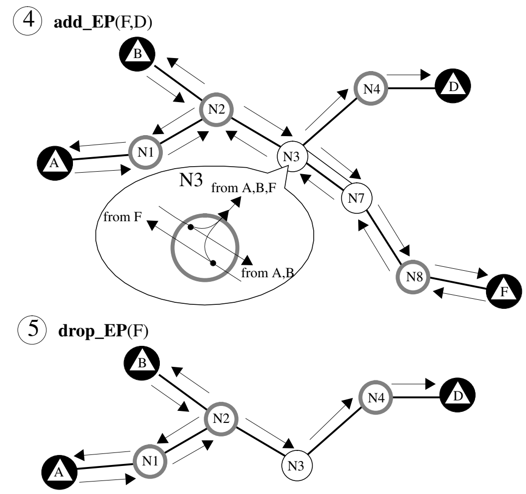

Figure 9. Call Topology After:
(4) add_EP(F,D) and
(5) drop_EP(F) Requests

If client F wants to add client D to the conference call as a uni-directional receiver and drop out of the call, he performs an ***add_ep*** command to add D specifying the permissions as receive only. Client F drops out of the call with the ***drop_ep*** command. Figure 9 shows the call topological changes that occur after each of these requests.

6 Software Architecture

We selected a general purpose UNIX[†] based workstation (currently a SUN Sparcstation[‡] I[+]) and the C++ language (version 2.0) as a platform and language for the control processor software that implements connection management. We also utilize the AT&T C++ task library to provide lightweight tasking control.

The major elements that comprise the control processor software (Figure 10) include: the Switch Module Interface (SMI), the Node Controller (NC), the Connection Manager (CM), the Overseer (for OA&M functions), the Router (RR) and the Resource Manager (RM).

6.1 The Switch Module Interface

The SMI software is embedded in a custom designed interface that communicates to

† UNIX is a registered trademark of AT&T

‡ SparcStaion is a registered trademark of SUN Microsystems

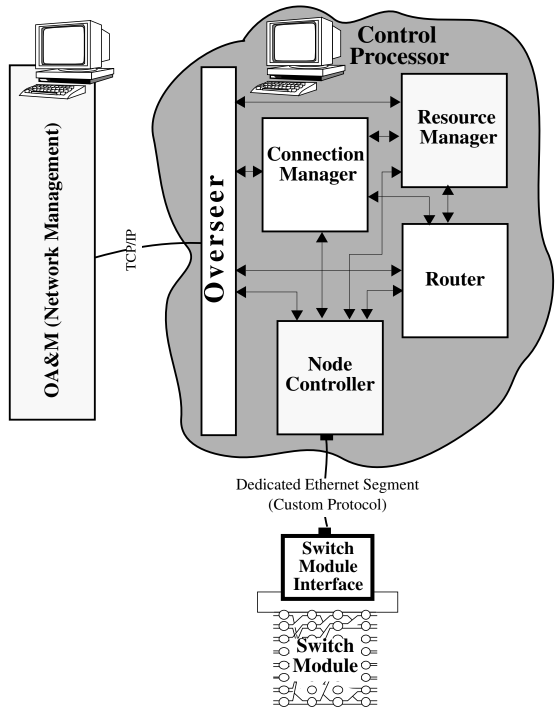

Figure 10. CP Software Architecture

the CP workstation over a dedicated Ethernet segment. The executable SMI program (written in C and assembly) is downloaded from the NC when the system is powered up. The SMI embeds multiple cells from the SF into an Ethernet frame and transmits the frame to the CP. It receives Ethernet frames with multiple cells embedded from the CP and sends the cells to the switch fabric. The SMI handles other functions relating to SF control, for example, power-up, reset and global SF timing.

6.2 The Node Controller

The Node Controller implements the connection management transport protocols, CMATP and CMITP, and a switch management protocol, the *SF-Update* protocol. The connection management transport protocols are used by the higher level components (the Overseer, CM, RR and RM) to communicate with their counterparts on other switch nodes and with clients. The SF-Update protocol is used to modify the routing tables embedded in the switch fabric.

6.3 The Connection Manager

The Connection Manager implements the CMAP and CMIP protocols. The CM controls the creation and maintenance of calls and connections for the node. It utilizes the services of the RM, RR and NC to accomplish its tasks. The CM is the only process that understands the notion of calls and connections and the only process that maintains the information that pertains to these constructs.

6.4 The Overseer

The Overseer has two main functions: (1) handle NMAP and NMIP network management requests and (2) control the operation of the other CP processes. The Overseer processes all NMAP messages from remote network managers and sends them to the appropriate process. It also periodically checks the state of the other CP processes and logs information requested by network managers (for example, number of calls currently active).

6.5 The Resource Manager

The Resource Manager maintains all information about the current state of the switch hardware, including the allocation of bandwidth on the node's fiber optic links and the number of *virtual* and *broadcast* channels in use. The RM also implements the bandwidth allocation algorithm for a node's links.

6.6 The Router

The Router receives addresses from other CP processes and returns the appropriate next hop link to traverse in order to reach that address. The Router also implements the RRIP protocol that allows Routers to communicate network state information with one another for use in making routing decisions.

6.7 Interprocess Communication

Currently each module identified in Figure 10 exists as a heavyweight UNIX process and communicates using a custom designed interprocess communication tool, based on UNIX System V shared memory primitives, that we refer to as a Priority Queue (PRIQUE). The PRIQUE is a unidirectional queue that allows messages (or packets) to be transferred from a *Producer* to a *Consumer* without incurring the overhead of system calls, thereby reducing process blocking and context switching. The queue is implemented as a circular data structure with *Producer* write and *Consumer* read pointers that control access. The shuffling of higher priority messages is performed automatically by the *Consumer* (hidden in the PRIQUE class implementation). The PRIQUE also allows "out-of--band" messages to be sent to the *Consumer*, bypassing the queue's normal delivery policy (for use when the queue is full).

7 Future Directions

The prototype system described in this paper is under development and is scheduled to be demonstrated in mid to late 1991. We hope to continue expansion of the testbed with follow on projects which may provide campus wide broadband connectivity, increase the number of applications using the network and interface the network to other fast packet projects. The following two sections describe how we might port our software to other switch architectures and integrate other B-ISDN protocols with our protocol suite.

7.1 CP Portability

Porting CMAP and CMIP to other ATM switch architectures requires the modification of the network manager, the NC and the RM (highlighted in Figure 5). The Overseer, CM and RR do not contain architecture specific code. As long as the switch architecture supports multicast communications and a host interface for switch updates, the CP software can be ported.

The RM and NC are written specifically for the switch architecture that they control. Components of the RM and NC, however, are architecture independent, as well as the interprocess interface to the RR and CM.

The NC encapsulates the functionality required to setup connections in the hardware and the RM understands and allocates the hardware resources that are required to implement a connection setup request from the CM. The switch specific data used by the RM is hidden from the CM. The CM has the ability to map the input from one link to the output of many links and understands only link numbers and the division of links into several *virtual channels*.

7.2 Integrating other Protocols

We view CMAP as one component in a larger protocol suite that offers network clients such services as Intelligent Network (IN) signaling, SMDS datagrams and narrow-band ISDN Q.931 connection setup. We envision using CMAP as a core protocol upon which the other services can be built.

Using CMAP (and its companion CMIP) as a core protocol is advantageous because it is resident on every access node of the network, thereby giving all clients access to the extended services. One method for adding these services would have *stub* processes at the CP to receive signaling messages for integrated protocols (such as IN). This process would use the capabilities of CMAP to set up connections with a service provider resident somewhere in the network. The service provider could communicate with other service providers, using CMIP, to satisfy the client's request.

Service integration can use CMAP directly or transparently. For example, direct service integration might allow clients to explicitly request access to an SMDS processor and use the assigned connection to gain access to an SMDS overlay network. Alternatively, we could define well known access channels and CMAP could set up, for all network clients, connections that route client messages to the service providers (presumably, spread throughout the network). This second method has the advantage of providing multiple services at the client access point without requiring the client to have specific knowledge of the service provider.

8 Conclusions

We have described a model for multipoint communication that takes advantage of the distinct qualities of B-ISDN networks. We described the protocols and software components of a connection management system for a prototype B-ISDN development project that will implement one solution (of many possible) to the B-ISDN signaling problem. We believe that the specification of a connection oriented protocol for B-ISDN is needed in the near term. We hope that our ideas for components of a signaling protocol are influential in the upcoming discussions on B-ISDN signaling standards.

References

[1] Blaine, G. J., R. C. Furguson, J. W. Studt and R. A. Whitman, "ISDN: Early Experiments as a Wide-Area Extension to LAN-based PACS," *Proceedings of SPIE*, vol. 1091-02: Medical Imaging IV, February 1990.

[2] Bubenik, R. G., J. D. DeHart and M. E. Gaddis, "Multipoint Connection Management in High-Speed Networks," submitted to *INFOCOM 91*, April 1991.

[3] CCITT Recommendation Q.931 (I.451), *ISDN User-Network Interface Layer 3 Specification*, Geneva, 1985.

[4] Giacopelli, J. N., W. D. Sincoskie and M. Littlewood, "Sunshine: a High Performance Self Routing Broadband Packet switch Architecture," *Proceedings of the International Switching Symposium*, June 1990.

[5] Haserodt, K. and J. S. Turner, "An Architecture for Connection Management in a Broadcast Packet Network," Washington University, Computer Science Department, Technical Report-WUCS-87-03, 1987.

[6] Huang, A. and S. Knauer, "Starlite: a Wideband Digital Switch," *Proceedings*

of Globecom 84, pp. 121–125, December 1984.

[7] Kato, et. al., "Experimental Broadband ATM Switching System," *Proceedings of Globecom 88*, December 1988.

[8] Kato, Y., T. Shimoe and K. Murakami, "A Development of a High Speed ATM Switching LSIC," *Proceedings of the International Communications Conference*, vol. 2, pp. 562–566, April 1990.

[9] Minzer, S. E. and D.R. Spears. "New Directions in Signaling for Broadband ISDN," *IEEE Communications Magazine*, vol. 27, no. 2, pp. 6–14, February 1989.

[10] Tu, S. C. and Leung W. H., "Multicast Connection-Oriented Packet Switching Networks," *Proceedings of the International Communications Conference*, vol. 2, pp. 495–501, April 1990.

[11] Turner, J. S., "New Directions in Communications," *IEEE Communications Magazine*, November 1986.

[12] Turner, J. S., "Design of a Broadcast Packet Switching Network," *IEEE Transactions on Communications*, vol. 36, no. 6, pp. 734–743, June 1988.

[13] Yeh, Y. S., M. G. Hluchyj and A. S. Acampora, "The Knockout Switch: a Simple Modular Architecture for Performance Packet Switching," *International Switching Symposium*, March 1987.