TOPS: An Architecture for Telephony over Packet Networks

Nikolaos Anerousis, *Member, IEEE*, R. Gopalakrishnan, Charles R. Kalmanek, *Member, IEEE*, Alan E. Kaplan, William T. Marshall, Partho P. Mishra, Peter Z. Onufryk, *Member, IEEE*, K. K. Ramakrishnan, *Member, IEEE*, and Cormac J. Sreenan, *Member, IEEE*

***Abstract*— Packet telephony is of increasing interest in both the telecommunications and internet communities. The emergence of packet telephony will create new services, and presents an opportunity to rethink how conventional telephony services are implemented. In this paper, we present an architecture for telephony over packet networks (TOPS). TOPS allows users to move between terminals or to use mobile terminals while being reachable by the same name. TOPS users can have multiple terminals and control how calls are routed to them. TOPS allows for terminals with a range of capabilities such as support for video, whiteboard, and other media with a variety of coding formats. TOPS retains the necessary information on terminal capabilities to determine the appropriate type of communication to be established with the remote terminal. The architecture assumes that the underlying network supports the establishment of end-to-end connectivity between terminals, with an appropriate quality of service.**

The components of TOPS are a *directory service*, an *application layer signaling protocol*, and a *logical channel abstraction* for communication between end-systems. The directory service maps a user's name to a set of terminals where the user may be reached. A user can control the translation operation by specifying profiles that customize how his name is mapped to a set of terminals where he can be reached. Terminal capabilities are also stored in the directory service. The application layer signaling protocol establishes and maintains call state between communicating terminals. The logical channel abstraction provides a shared end-to-end context for a call's constituent media and control streams, while isolating the applications from the details of the network transport mechanisms. In addition to supporting simple point-to-point calls, the architecture supports both centralized and decentralized *conferencing*. We also introduce a simple *encapsulation format* for voice.

***Index Terms*—Communication system signaling, directory service, telephony.**

I. Introduction

PACKET telephony is of increasing interest in both the telecommunications and Internet communities. This interest stems from the desire to integrate multiple modes of communication using a single infrastructure. Packet networks allow the integration of data and voice and provide a straightforward way to offer a variety of new communication services. Increased intelligence and more elaborate user interfaces at the terminals[1] allow increased flexibility in handling not only voice but also other forms of communication. In addition, the combination of packet networks and enhanced terminal capabilities provide an opportunity to reconsider both the nature and the implementation of telephony services. Over time, we believe these services will evolve to support substantially enhanced human communication, with voice continuing to be an important mode of communication.

In this paper, we present an architecture for telephony over packet networks (TOPS) that exploits the flexibility provided by packet switching and the greater intelligence in end-systems. The key elements of the TOPS architecture are as follows.

- A *directory service* to flexibly map user names to terminals where users can be reached.
- An *application layer signaling* protocol to allow end-systems to negotiate capabilities, establish and maintain call state, and support advanced telephony features.
- A *logical channel* abstraction and an efficient *encapsulation* format to insulate telephony application from the underlying network transport capabilities.
- Mechanisms to support a variety of *conferencing* modes.

One goal of the TOPS directory service is to make it more convenient to reach a user. A packet telephony service requires a directory service function that can translate between telephone numbers and network layer addresses, such as IP or ATM addresses. This simple functionality is enhanced in TOPS to allow callers to reach a person using a *name* rather than an address by storing information in the directory service about user names and the set of terminals where they can be reached. Callers obtain terminal addresses of users by issuing a name resolution query to the directory service. As users move between terminals, they can securely modify their directory entries to reflect the locations where they can be reached. Thus user mobility is supported as a fundamental capability of the TOPS architecture. Since the directory service returns information about individual users, it provides a mechanism for each of them to control how a name resolution query is handled. Thus users can

Manuscript received February 1998; revised July 1998. This paper was presented at the 8th International Workshop on Network and Operating Systems Support for Digital Audio and Video, Cambridge, U.K., July 8–10, 1998

The authors are with AT&T Labs-Research, Florham Park, NJ 07932-0971 USA (e-mail: nikos@research.att.com; gopal@research.att.com; crk@research.att.com; aek@research.att.com; wtm@research.att.com; partho@research.att.com; pzo@research.att.com; kkrama@research.att.com; cjs@research.att.com).

Publisher Item Identifier S 0733-8716(99)00004-9.

[1] Throughout the paper, we use the term end-system, terminal, and packet telephone interchangeably.

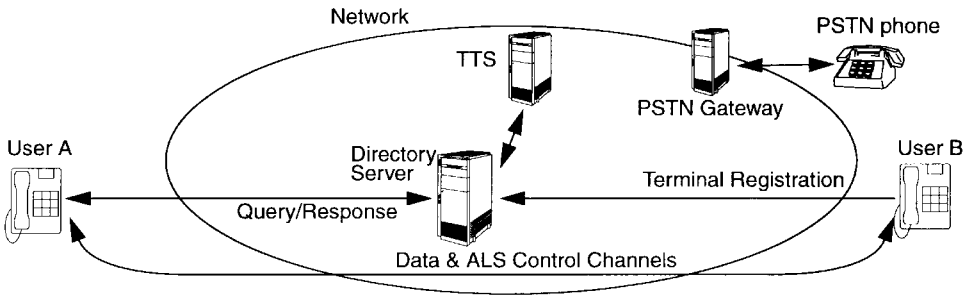


Fig. 1. TOPS components.

customize access to their information. The directory service also helps callers by providing them with a flexible set of call appearances and options. Compared to existing telephony features such as "personal phone numbers" and "intelligent 800-number" services, this approach is more flexible, while also giving more control to the user. TOPS *terminals* range from computers running telephony applications to low-cost packet telephony appliances. Terminal capabilities are stored in the directory and returned to the caller as part of name resolution so that appropriate communication resources can be set up. Fast terminal mobility is handled by interfacing the directory service to a *terminal tracking server* (TTS) that deals with mobility, thus isolating the user from its complexity.

Traditional telephone signaling was designed to support relatively simple terminals. The functions of routing, number translation, connection establishment, call control, terminal control, usage recording, and advanced feature support are all implemented within the network and are tightly intertwined. As a result, the complexity of the signaling software, the reliability requirements of this integrated system, and the potential for interactions among features make it difficult to deploy new services. In contrast, packet networks already provide connectivity efficiently while services are implemented at the application layer. We therefore view the remaining pieces of signaling required to establish a call as application-layer functions. An application-layer signaling (ALS) protocol sets up and manages the necessary state at the end-systems and supports the negotiation of terminal capabilities and media types. ALS has the advantage that it can evolve to support a wide range of functions, including multiple media, independently of the underlying network architecture. ALS uses logical channels (LC's) as pipes for communicating both control and media data, to abstract the details of the underlying transport. LC's can be used to multiplex control and media channels onto a single shared context at the transport layer. The capability negotiation function included in ALS allows for flexibility; there is no restriction to use only predefined media channels and encodings. The ALS protocol is also used to transfer and tear down calls, dynamically add or remove media channels, and perform conferencing control functions.

The third element of the TOPS architecture is a *logical channel* abstraction and an efficient *encapsulation* format to insulate telephony applications from the underlying network transport capabilities. The encapsulation format has been designed to contain only the overhead essential for transport of audio data, while allowing for synchronization among multiple media streams. This format avoids some of the fields, such as timestamps, that are typically included in a more general real-time transport protocol such as the Internet real-time protocol (RTP) [17]. However, header extensions may be used to carry additional information, if required.

TOPS also supports conferencing. Mixing can either be done at a bridge or in a distributed manner by the end-systems. Both tightly and loosely coupled conference control modes are supported. TOPS uses the directory service to distribute the information necessary for conference control, and allows the choice between centralized and decentralized conferencing to be nearly transparent to the client application.

Packet phones in TOPS use a gateway to interoperate with the public switched telephone network (PSTN). The gateway acts as an end-system for the ALS protocol on the packet network and translates signaling messages between the packet network and the PSTN. It also performs media transcoding when needed. Fig. 1 illustrates the various components of the TOPS architecture and their relationship.

The rest of this paper is organized as follows. Section II describes our vision for how person-to-person communication might evolve. Section III discusses the directory service. Section IV focuses on application-layer signaling. Section V describes our approach to end-to-end transport of control and real-time data. Section VI discusses conferencing, including the tradeoffs between centralized and decentralized mixing. Section VII discusses interworking with the existing telephone network and the status of our implementation efforts. Finally, Section VIII discusses related work and Section IX concludes the paper.

II. Evolution of Telephony Services in Packet Networks

The combination of packet networks and sophisticated end-systems will enable a variety of new services. Packet networks offer a variety of new capabilities not present in plain old telephone service (POTS). Examples of these capabilities are: support for multiple concurrent streams at end-systems, much richer signaling, and the ability of end-systems to interact with network servers. The emergence of packet telephony is also likely to result in a much greater diversity of packet telephones than that of POTS telephones. Packet telephones will range from general-purpose computers running telephony software to low-cost single-function packet telephone appliances. The user interface for these devices will also vary widely, from

rich point-and-click graphical user interfaces with a mouse and keyboard, to the standard twelve push-button telephone interfaces. Even the simplest packet telephones will contain a processor and memory. These new network and end-system capabilities not only allow the implementation and evolution of existing telephony services in packet networks, but also enable the creation of entirely new services. Furthermore, these new capabilities allow a migration of some telephony services from the core network, where they are currently implemented, to intelligent end-systems and servers. This migration of service-specific functions to end-systems not only simplifies provisioning and maintenance of the core network, but also increases the speed with which new services can be deployed.

As an example of the migration of existing services to a packet network, consider the implementation of the familiar *call waiting* feature. In the POTS network, an indication of a second incoming call is provided through the insertion of special tones by the local telephone switch into the call that is being interrupted. This feature not only requires the local telephone switch to insert tones and detect user responses (i.e., a switch-hook), but also requires it to maintain the signaling state for two simultaneous calls. The ability of a packet telephone to support multiple simultaneous streams, coupled with its processing capabilities and enhanced user interface, allows this service to be implemented entirely at the end-system. When a call is placed to a packet telephone which is already participating in a call, the called packet telephone may decide how to handle the incoming call request. It may locally generate a call waiting tone, as in current call waiting, or on packet telephones with an enhanced user interface it may create a pop-up icon with information about the call. This pop-up icon may include information about the call such as the caller's name, the topic of the call, the importance of the call, and so on. Other common POTS services such as: call forwarding, call transfer, three-way calling, and return-call may similarly be implemented entirely by end-systems.

Packet networks allow traditional POTS services to be enhanced without modification to core network switches or their software. An example of such an enhancement is *n-way call waiting*. In this service, a packet telephone's enhanced user interface may be used to accept multiple incoming calls and allow a user to cycle through them in an arbitrary manner. We also envision more imaginative capabilities enabled by the richer signaling allowed by packet networks. For example, a meek caller may indicate in the call request that he wishes to be notified if the called party is already in a call. With this notification, he is given the option of whether or not to interrupt the called party with a call waiting tone, allowing him to choose to not disturb the call in progress.

The ability of packet telephones to access network servers allows the creation of entirely new network services. For example, a directory server can allow calls to be placed by the called party's name rather than by terminal address, as is typically done in the POTS network. This added level of indirection allows users to move and connect to the network at an arbitrary packet telephone and continue to receive incoming calls without the need to inform callers of the new location. This is similar to the personal phone number service already provided in the POTS network, but is much more flexible. The directory server can be personalized to use information such as the identity of a caller, time of day, and current location of the called party to route a user's phone calls to a specific packet telephone or voice mailbox. For example, calls could be routed from 9 am to 5 pm during the week to an office phone, from 5 to 10 pm to a home phone, and at other times to a voice mailbox. The directory server may also be used to provide access control on a call-by-call basis, taking into account information such as the caller's identity and the "urgency" of the call. The added level of indirection offered by such a directory server also allows a seamless integration of wired packet telephones with mobile packet telephones.

Thus far we have considered traditional voice calls, but it is clear that in the future telephone calls will expand to multimedia calls. For example, while users speak they may share a common whiteboard, view the same video clip, or participate in a video call. This creates the need for parties to be able to discover compatible media formats when placing a call and to dynamically modify media formats during a call. Network directory servers may be used to provide information on media formats supported by a called party, simplifying the task of capability negotiation during call setup.

Packet communication is opening the door for a variety of new services. For example, a user may be able to initiate a call from a particular terminal before actually reaching that terminal. Another example is that when a multimedia call is established, the caller may wish to move to a terminal with different capabilities, while saving the context of the current communication. For example, a caller may answer a call in the bedroom saying "Hold-on, I'll take your call on my videophone in the living room." Such a "teleporting" [15] service requires dynamic adaptation of media formats and possibly the transfer of state. Dynamic adaptation of media formats is also required when a call is transferred, for example by a receptionist, who supports a different set of media formats than those required by the ultimate destination of the call. Multimedia calls may benefit from a "virtual hold" service in which parties involved in a call are allowed to pause for an extended time period and later resume the call where they left off. This requires saving of the call state and the ability to reinstate it once the call is resumed. To conserve network resources, connections used by media streams should be released during a virtual hold. This is conveniently achieved in packet networks where the mechanism used to modify a call's state is separate from the mechanism used to establish end-to-end connectivity.

Packet telephony may be integrated with other applications running on an end-system to provide advanced services. An example of such a service is a system which provides the capability to take "notes" during a call. These notes might include both a record of the conversion, audio and video, as well as personal annotations taken during the course of the call. Such a service would also prove useful during conferences to allow users who have just joined a conference, or have had a call transferred to them, to quickly review the prior discussion.

The services outlined in this section are made possible by the ability of packet telephones to support multiple concurrent

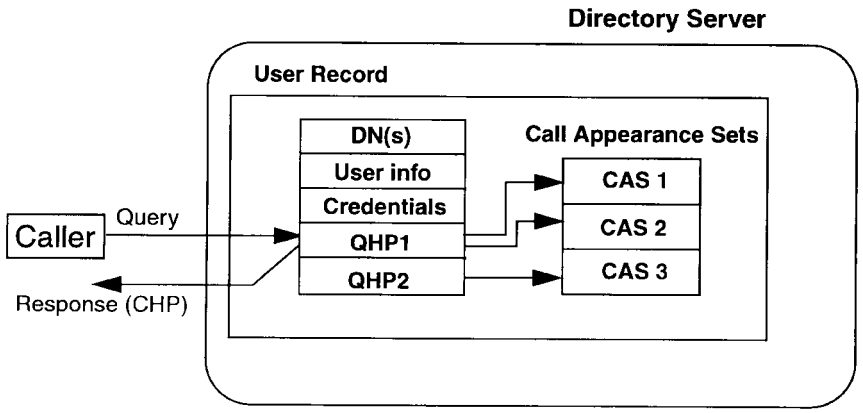


Fig. 2. Structure of a user record.

streams, richer signaling, complex processing, flexible user interfaces, and interactions with network servers. Telephony in packet networks will enable media communications to be easily integrated with other networked applications, much as is currently occurring with text and images on the Web. This integration offers the opportunity for a myriad of new services, many of which cannot even be imagined today. The TOPS architecture, described in this paper, illustrates a particular framework for realizing this vision.

III. TOPS Directory Service

Network directory services provide a convenient way to insulate users from the need to deal with network addresses. The domain name system [4] is a familiar example of a directory service that translates host names to Internet addresses. The PSTN also uses directory services to support features such as 800-numbers, which allow a dialed number to be mapped to many different telephone numbers based on certain rules or preferences. However, in order to reach the vast majority of PSTN subscribers, a caller needs to know the network address (telephone number) of the terminal closest to the subscriber's current location, e.g., his office phone, home phone, car phone, etc.

The goal of the TOPS directory service is to provide a simple dial-by-name capability: callers place calls using a *distinguishing name* (DN) that uniquely identifies the person they wish to reach. The directory maps a query for a distinguishing name to one or more *call appearances*, each of which typically represents a terminal at which the user can be reached. In this way, the directory can support both user mobility (users can be reached at any terminal by updating their directory entry with the address of their present location) and terminal mobility (a user can continue to receive calls at a mobile terminal even if its network address changes as it roams across the network). In the PSTN, the "single number reach" capability is achieved by a network switch attempting to set up the call based on the translation of the name to a unique telephone number. Call appearances may also refer to other ways of "reaching" a user, such as a voice mail server or secretary. Our approach allows the end-systems to use the information provided by the directory service directly, thus exploiting the enhanced capabilities of these terminals. The directory service also allows for the incorporation of query-handling profiles that allow users to control which call appearances are returned by name resolution queries and when they are returned.

The directory may be viewed as a distributed database. A hierarchical name space is used to derive the distinguishing names of users. TOPS allows multiple hierarchical name spaces to be defined over the database.

In this way, a user record can be accessed through an X.500 distinguishing name (e.g., CN=Graham Bell, O=ATT, C=US) [22], an e-mail address (e.g., graham.bell@att.com), or a regular telephone number. In addition, the directory may support traditional database search functions, in case users do not remember DN's directly. For example, it may be possible to look up users by last name, address, etc.

The naming hierarchy typically corresponds to administrative responsibilities for portions of the name space. Different network operators or large businesses own portions of the name space and operate their own servers for their part of the name space. The master copy of a user's record is always stored in one authoritative server (referred to as the user's home directory). Terminals access the directory service through a local directory server (this server is usually the one closest to the current location of the terminal and is administered by the network operator responsible for the domain to which the terminal is currently attached). An inter-directory server protocol is responsible for locating the home directory of the called user and fetching the appropriate record. TOPS does not allow user records to be cached by local servers; only the address of the home directory server can be cached. This ensures that the user's call appearance information received at the caller's terminal is always up-to-date.

A. Directory Contents

Fig. 2 illustrates the contents of a user record in the directory database. Each record is associated with one or more DN's. The record contains information about the user (e.g., full name, address, etc.), a number of *query-handling profiles* (QHP), one or more sets of call appearances where the user can be reached, and credentials to authenticate the user. The credentials are used by the directory to authorize a modification of the user record and by external servers to authenticate access to network services.

TABLE I
EXAMPLE OF A CALL APPEARANCE SET

Type	Identifier	NLA	TLA	Te minal Capabilities	Application Info	Priority	Timeout	Comment
Terminal	ESN1	NLA1	TLA1	audio G.711	none	1	30 sec	work phone
Terminal	ESN2	NLA2	TLA2	audio G.711 video H.261	none	2	20 sec	home phone
TTS	TTS_ID	NLA3	TLA3	audio	ESN3	2	30 sec	mobile phone
DN	my_secy@ att.com					3		secretary
PSTN				audio G.711	(973) -555- 1212	4	15 sec	parent's POTS phone
Server	VM_ID	NLA4	TLA4	audio	VM box #	5	10 sec	voice mail

The query-handling profile (QHP) is used to determine the response provided to incoming queries about this user. The QHP allows users to control access by specifying who can reach them, where, and when. The QHP consists of a table of matching rules that are executed sequentially. Every rule is associated with a *call appearance set* (CAS), a group of call appearances that are returned to the querying terminal representing the different ways in which the user can be reached. There is always a default CAS which is returned in case the QHP cannot match the incoming query to a particular rule. A call appearance set consists of a number of call appearances (see Table I). Call appearances are typically associated with a terminal device or server and contain the following set of fields.

- The *type* field identifies the type of the call appearance. In most cases, the type refers to a terminal or server, such as a voice mail server. In addition, a call appearance may be a pointer to another user record (DN) or indicate that the call appearance exists within another network domain, in which case a gateway must be employed.
- The *identifier* field contains optional information that can be used to identify the call appearance among others of the same type, if necessary. For example, call appearances for terminal devices may contain an electronic serial number (ESN). The ESN may be used to support terminal mobility, in order to update the relevant call appearances when a terminal changes its network address.
- The *NLA/TL4* fields contain the network-layer address of a terminal device or server, and the transport-layer address of the TOPS application. The format of the addresses stored in these fields may differ depending on the network to which this terminal is attached. For telephony gateways, the NLA/TLA of the gateway may be statically assigned, or may be dynamically generated by the local directory server using a specialized gateway location mechanism since the address of the gateway to be used may depend on the location of the caller. The NLA/TLA fields are typically omitted when the call appearance contains another DN.
- The *terminal capabilities* field contains information about the media processing capabilities of the terminal (e.g., video, voice, or fax), or the preferences of the user for a particular media type for communication.
- The *application information* field is used to convey additional information to the calling application. For example, for a call from a packet telephone to a call appearance on the PSTN, the phone number associated with the DN may be stored in the directory server, and passed to the gateway by the calling application.
- The *priority* field indicates the order in which call appearances should be attempted from the caller's terminal. Entries with the same priority are attempted at the same time (the equivalent of having several phones ring at once).
- The *timeout* field indicates the amount of time before trying the next call appearance if no reply is received.
- Finally a *comment* is used as a reminder for both the user and the caller of the nature and function of the call appearance.

Table I illustrates a call appearance set for a user who has indicated as his first preference the terminal at his office. The caller tries this terminal first, and if no reply is received within 30 s, tries simultaneously the next two call appearances: the terminal at the user's home and a cellular phone (accessed indirectly through a terminal tracking server (TTS)—see Section III-C). If these call appearances fail to provide a response, the terminal attempts to contact the person's secretary by initiating a second directory query. If this fails as well, the terminal attempts to reach a PSTN phone through a gateway. Finally, if everything else has failed, the caller is connected to the callee's voice mail server.

B. Client Interactions with the Directory Service

1) Service Subscription: Directory records are created when a user subscribes for service. At that time, he supplies the full name, billing address, and service preferences to a service representative, who then assigns a DN. To maintain compatibility with domains that use only numeric addressing, such as the PSTN, a unique telephone number may also be assigned as an alternative DN. Cryptographic credentials for user authentication and registration are also created at the time of service subscription.

A directory entry is required only for users that wish to receive calls. An "anonymous" user who does not have a directory entry with stored credentials can also place calls as long as he has provided some means for being charged for the service (e.g., a prepaid calling card, etc.).

2) User Registration and User Mobility: In order for a user to receive calls, one or more call appearances must be entered into the directory record. Call appearances can be configured in three different ways.

- Statically, when the service representative creates the record.
- Off-line, when the service subscriber uses directory management software to configure his/her record from, say, a computer.
- Dynamically, when the user registers at a new terminal to receive calls, or moves one of his previously registered terminal devices to a new location. The user may register at a terminal that retains his information (his home phone) or to a terminal not associated with a user (e.g., a public phone).

In the last two cases, the directory must authenticate the user to prevent unauthorized access to the database by using the credential information stored in the user's record. Once authenticated, the user can update or delete existing call appearances, add new ones, or update his QHP to implement a new query resolution policy.

When a directory entry is configured off-line or dynamically, a challenge-response protocol may be used to authenticate the user. The terminal device may not have a keyboard, in which case the user may employ a smart card where the necessary credentials are stored. When the smart card is used at a public terminal, the directory can be updated automatically with a new call appearance entry for this terminal. Similarly, when the user's terminal moves to a new location, it may have been authorized beforehand to update the call appearance record in the directory with its new NLA/TLA for that user (possibly with a password). The user has the flexibility to define the desired level of security for updating the directory record. In addition, the introduction of a new call appearance in the user's record or the modification of an existing one to reflect the new address of a terminal can be specified to be valid only for a specific period of time. For example, user registration at a public phone need only be valid while the smart card is in place. Explicit user deregistration can also trigger updates of the relevant call appearance records. Deregistration from a public terminal can trigger a deletion of the corresponding call appearance.

3) Querying the Directory: In order to call another user, the caller queries the directory to obtain the call appearance(s) of the person he wishes to call. The only information that is required in a query is a DN of the called party. Optionally, callers may also supply their own DN's, the types of media to be included in the call, the capabilities of the calling terminal, and an indication of urgency.

The directory also supports the concept of a user *persona*. In this case, the caller supplies additional information that identifies the role in which a user is being called, for example professional, social, emergency, etc. Every user persona can be handled by a different QHP. The QHP contains the logic for handling a query related to the user/persona being called. The QHP may produce different responses indicating where a user can be reached based on who sent the query, the time-of-day, and the purpose of call. In addition to providing customizability, this allows the user greater control over the privacy of information compared to a traditional directory service.

The directory service does a small amount of processing to ensure that the caller's and callee's terminals are compatible. This optimization can minimize the amount of end-to-end capabilities negotiation during call establishment for a subset of well-defined forms of communication, such as simple telephony.

4) Query Response: The response to a query is a *call handling profile* (CHP). A CHP is a subset of a call appearance set stored in the user's record and is determined after executing a QHP. Call appearances in the CHP are derived from call appearance records in the directory by adding or filtering out some fields.

The terminal capabilities returned with the CHP may not be the full set of capabilities that the particular call appearance can support. This allows a callee to limit the terminal capabilities depending on the caller, the time of day, etc. For example, a user may wish to restrict incoming calls from certain people to be audio only.

Fields can be added to a call appearance to implement interesting services. For example, the directory could insert an estimate of the per-minute cost for communicating with a particular call appearance. The caller could then use this information to restrict "expensive" call appearances or keep track of the accumulated cost of a call.

When the calling application receives this information, it may use the CHP directly, taking into consideration application or user policy, or it might display the CHP to the caller (if the terminal is equipped with a display) awaiting further instructions. For example, the caller may want to skip call appearances that do not meet the current needs (e.g., a voice mailbox) or are expensive.

C. Support for Terminal Mobility

A directory record is updated either when a user moves between terminals or a terminal changes its point of attachment to the network and receives a new network-layer address (NLA). This works well as long as a user is registered at a terminal that does not move frequently; i.e., on the time scale of seconds or minutes. However, several performance problems manifest themselves with faster moving terminals. First, directory servers are usually optimized to support a high rate of lookups but a relatively low update rate. Second, it may be inefficient to update the home server every time a terminal changes its NLA, because the directory server may be far from the current terminal location.

To deal with the inefficiency of having to update the home directory service each time the terminal changes its NLA, we introduce an additional component in our architecture: a

"terminal tracking server" (TTS). A mobile terminal acquires an NLA and the address of the nearest TTS from the network via autoconfiguration. It then registers its electronic serial number and current NLA with the TTS. When a user registers at a terminal, the user record in the home directory is updated with a call appearance containing the network-layer address of the TTS, rather than the address of the terminal, in addition to a bit which identifies the terminal as a mobile terminal. In response to a query, the home directory server contacts the TTS to obtain the current NLA of the terminal and returns this value to the caller.

As long as a terminal moves within the domain of a particular TTS, there is no need to update any entry at the directory server. When a terminal moves to a new TTS, it is necessary to

1) update the entry at the previous TTS to point to the new TTS;
2) update the directory server entry (for each user registered at the terminal) to point to the new TTS.

Support for terminal mobility also requires the underlying network layer to reroute ongoing calls as terminals move. We assume this is supported at the network or link layer, for example using techniques such as ATM VC rerouting or mobile IP [13].

D. Directory Support for PSTN Interworking

Interworking with the PSTN is provided by telephony gateways that convert between the packet stream and the voice encoding, and the 8-bit mu-law format used by the PSTN. They also provide signaling interworking functionality as described in Section VII-B.

For a call from a packet terminal to the PSTN, the caller may have specified either a telephone number, or the DN of a user (such as an X.500 name), that has a PSTN telephone number as one of its associated call appearances. In either case, the directory service returns the address of a telephony gateway rather than the address of a terminal in the call appearance. When a query contains a telephone number, the local directory server may need to map from local dialing plan numbers to a standard E.164 address. Since a gateway is likely to handle a range of telephone numbers and since there may be multiple gateways serving the same range, the directory service may use specialized mechanisms for storing and searching for PSTN telephone numbers and for locating gateways. When a query contains a DN that is not a telephone number, the application information field in the gateway's call appearance contains the telephone number to be used by the gateway to place a call on the PSTN.

For calls from the PSTN to a packet terminal, the subscriber dials a regular telephone number that is routed through the PSTN to a gateway. The gateway then tries to resolve the telephone number to a user inside the TOPS domain. Since the information conveyed from the PSTN is in numeric format only, it is likely that users would be assigned an additional DN in the form of a (personal) telephone number at subscription time. This number can be used from both the PSTN and the TOPS domain as an alternative DN to reach that user. Once the user record has been located in the directory, a call-handling profile is returned to the gateway to complete the call. When the called number answers, the gateway bridges the connection between the two domains. Meanwhile, the gateway generates the appropriate call progress tones to the PSTN phone.

IV. Application-Layer Signaling

Signaling in the telephone network supports multiple functions, including routing, number translation, connection establishment, call control, terminal control, usage recording, and advanced feature support. In packet networks, routing, connectivity, and resource management are already supported by existing network-layer mechanisms. Moreover, intelligent end-systems reduce the need for network entities to be involved in terminal control. In our view, telephony applications communicate with each other directly using an application-layer signaling protocol for control, and media streams for data. In this section, we describe the application-layer signaling (ALS) protocol that is used in the TOPS architecture.

A. Protocol Fundamentals

The primary functions supported by ALS are establishing and maintaining call state between applications. ALS is used at an end-system to set up a call, establish and manage media channels, negotiate parameters for these channels, and to implement additional service features. The protocol has a relatively small and simple set of basic messages that are augmented by optional messages through the use of header extensions. ALS is intended to support call management for a variety of types of calls, ranging from simple, point-to-point voice calls, to multiparty, multimedia calls, to calls that might involve other application services, such as a game or storage server. We describe how ALS is used for point-to-point telephony and multimedia calls in this section.

All communication between the calling and called parties takes place over logical channels (LC) that are used to distinguish information flows within a call. Each call has a default LC for exchanging ALS messages, and one or more LC's for media data. Fig. 3 illustrates the protocol relationship between ALS and the logical channel layer; details of logical channel operation are provided in Section V. LC's may share a transport context while providing different semantics to the constituent flows. For example, the ALS protocol may use an LC that provides reliable and ordered message delivery, whereas an audio stream requires an LC that transports real-time data while meeting the performance constraints of that audio stream.

ALS is invoked after the calling terminal has queried the directory service to obtain one or more call appearances for the called party. The sequence of message exchanges are shown in Fig. 4. An INVITE message is sent to the default LC at the address returned in the call appearance field of the directory response. The message contains the distinguishing name of the called party, and optional information identifying the calling party, subject, and urgency. A key function of the invitation message is capability negotiation which is used by the two parties to agree upon the number of media channels

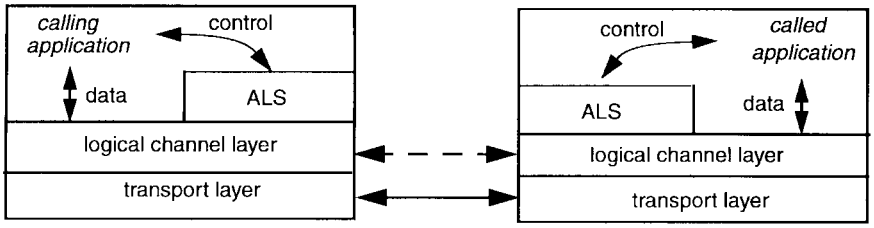


Fig. 3. ALS protocol layering.

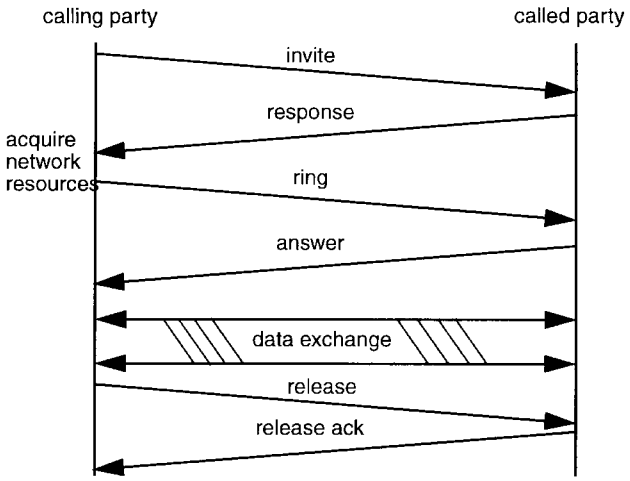


Fig. 4. Fundamental ALS message exchange.

TABLE II
KEY ALS MESSAGES

Type	Function
INVITE	invite called party to enter call; negotiate call state
RESPONSE	supply called party's reply; negotiate call state
RING	alert called party
ANSWER	notify caller the result of alerting the remote user
MODIFY	specify desired changes to the call state e.g. adding a media channel
MOD-ACK	reply, selecting acceptable changes from a modify request
RELEASE	notify intention to terminate the call
REL-ACK	acknowledge call termination; delete call state

for media data, as well as their attributes such as media type, framing, encoding, priority, and transport-related parameters. The ALS invitation contains the set of logical channels and their attributes as desired by the caller. These values may be derived by taking into account the called-party capability information provided in the directory reply.

The called party application replies with a RESPONSE message that indicates which of the desired media channels are available, as well as additional information that the called party application wishes to provide to the calling party application. If no channels are available, the ALS exchange terminates. If the response indicates that a set of channels that the calling party considers necessary are available, the calling party application then obtains the network resources required to support these channels. If it succeeds, it sends a RING message to the called party; otherwise, the ALS protocol terminates due to lack of resources. The RING message indicates the set of media channels that were actually allocated. When this message is received, the called user is alerted using some local mechanism. When the user accepts the call, an ANSWER message is sent to the calling party application. The ANSWER can contain the set of media that the user actually chose to accept. The called user can also indicate in the ANSWER message that he wishes to decline the call and a reason, e.g., user is busy. This exchange is all that is necessary to establish call state between the parties and allow communication to commence. We note that the invitation/response performs capability negotiation and provides additional information about the called party application in a single message exchange. For example, the RESPONSE may indicate that the called party does not wish to be disturbed. Once this initial handshake has been completed, the calling party application can allocate network resources if desired. This occurs prior to alerting the called party, as is usually the case in the PSTN.

Table II lists the key ALS messages. Once in an active call, either party can issue an ALS request to modify the call state. This can be to set up a new logical channel, to renegotiate parameters for an existing logical channel, or to tear down a logical channel. The called party's reply is used to accept, reject, or negotiate the proposed change. To enable this negotiation of the characteristics of the flow with the network, it is desirable that the network layer provide a convenient means for negotiation and renegotiation. An example of a lightweight network layer signaling protocol that allows end-systems to efficiently and conveniently negotiate quality of service characteristics is UNITE [21]. It supports both negotiation and renegotiation in a uniform manner.

To terminate a call, either the calling or called party must issue a RELEASE message to its peer, which in turn must send an acknowledgment. At that point the coders are stopped, logical channels are torn down, and related transport resources are freed up. Note that all ALS messages contain a calling-party-generated call identifier, allowing the context of each message to be easily determined.

B. Interworking Using ALS Proxies

An important feature of TOPS is the ability to interwork with the telephone network using gateways. Each gateway acts as an ALS proxy, processing messages on the packet network and taking appropriate action on the telephone network side. Fig. 5 shows both cases: calls originating on the packet network and calls originating in the telephone network.

Making a call from a packet terminal requires a directory query to obtain one or more call appearances, where each

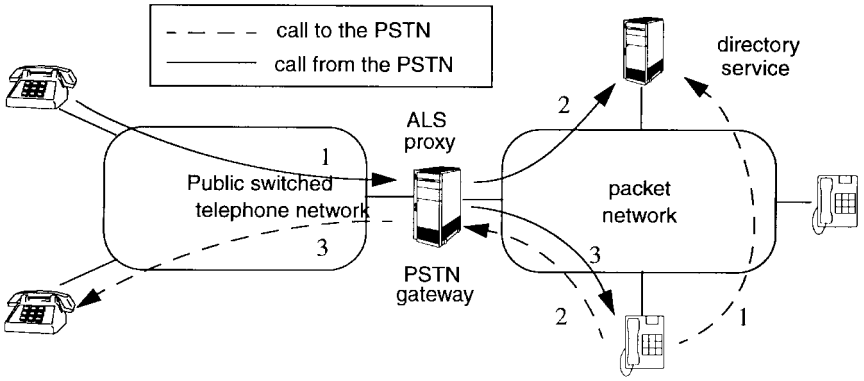


Fig. 5. Calling to/from the PSTN.

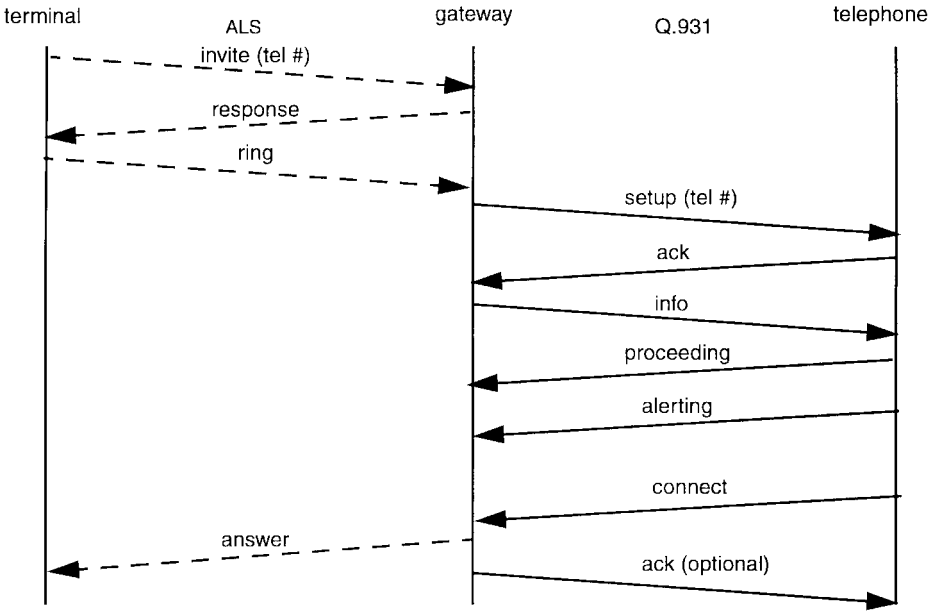


Fig. 6. Example of ALS proxy interworking.

call appearance contains transport and network-layer addresses identifying where to send the ALS invitation. When a call appearance is associated with a PSTN telephone, rather than a packet terminal, these addresses are used to identify a gateway running an ALS proxy. The PSTN telephone number is returned by the directory as part of the application-specific information in the call appearance. The packet terminal then initiates a regular ALS exchange with this gateway. The telephone number is included in the ALS invitation as a message extension. The gateway terminates the ALS protocol, translates messages to actions required on the telephone network side, responds appropriately to the called party, and maintains call state for each active call. This signaling interworking is illustrated in Fig. 6 for the case of a Q.931 ISDN gateway. A gateway can also provide media transcoding between the encodings used by the packet telephone and the PSTN.

From the telephone network's point of view, a gateway is a termination point for a set of telephone numbers. A call originating on the PSTN and destined for one of the TOPS terminals is routed to the gateway and terminated there. The gateway issues a directory query to map the telephone number (used here as a distinguishing name) to a call appearance, or to a set of call appearances if the telephone number is a personal number. The gateway then issues an ALS invitation, performs signaling translation, and maintains call state throughout the call.

C. Implementing Telephony Services in TOPS

Many of the standard POTS services can be supported in TOPS by appropriate customization of the query-handling profile for each distinguishing name, or the call-handling profile returned for each query. For example, POTS supports various flavors of call forwarding in which a call placed to a certain telephone number is forwarded to another number either by default, or if a user is busy, or if he does not answer. Supporting default call forwarding in TOPS is easy; a query to the directory service for a particular user name returns the address corresponding to the packet telephone (or perhaps a voice mailbox) to which the user would like the call forwarded.

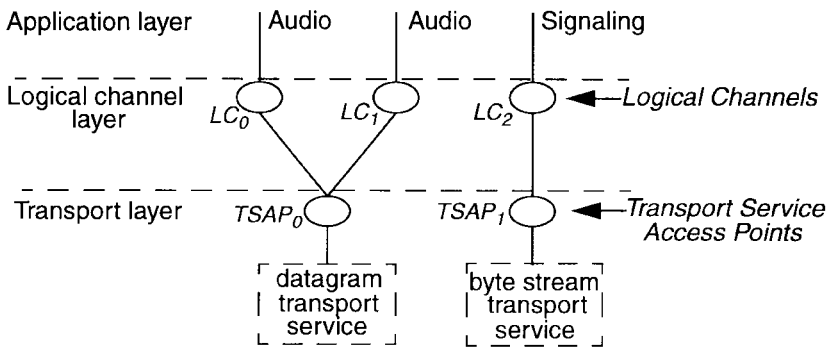


Fig. 7. Example of mapping logical channels to transport services.

Subsequently, the caller's terminal uses ALS to set up a call to this address.

Forwarding a call when a user does not answer or indicates that he is busy, is accomplished by having the directory service return two NLA's in response to the initial query with a call-handling profile that indicates which one to try first. If ALS fails to get a positive answer message to the RING sent to the first NLA, it tries the next NLA.

The simultaneous ring feature provided by PBX's is emulated by having the directory service return a call-handling profile that has more than one address at the same priority. ALS then attempts to establish a call to all of these addresses simultaneously. Service similar to the "advanced 800-number" service can also be emulated using the directory service. In the telephone network, this service supports sophisticated policies for translation of an "800" number to terminating telephone number. In TOPS, the query-handling profile can load-balance among call appearances by cycling through them in round-robin order or by following a more sophisticated algorithm. A similar mechanism can be used to do time-of-day-based routing. The QHP provides a flexible way of customizing these features by allowing individual users to specify the behavior and policies they need.

In addition to customizing the QHP, a user may control the service features he receives by suitably customizing the behavior of his telephony application. For example, when there is an incoming call to a user who is already involved in another call, ALS determines based on a user-service profile whether to provide the user with an indication of the incoming call, return a busy indicator to the caller, or to provide the caller access to a voice mailbox.

Finally, ALS is also used to create and manage conferences, and allow users to participate or be invited to join. This topic is discussed separately in Section VI.

V. Logical Channels and Encapsulation

A goal of the TOPS architecture is that packet telephony applications should be shielded from details of the underlying transport and network layers. The logical channel (LC) abstraction is defined to provide this independence by presenting a uniform communications interface to the application layer. LC's are used to carry application-layer signaling and media data, and also support multiplexing of related data streams onto a single transport connection. In an ordinary telephone call this can reduce the delay of call establishment in some networks by multiplexing signaling and data on a single transport connection. Multiplexing of LC's also allows temporal ordering to be preserved amongst streams, without the need for explicit interstream synchronization. Use of multiplexing has implications for meeting LC quality of service requirements. The LC layer allows applications to control which LC's are multiplexed, taking into account their different quality of service, reliability, and delivery order requirements.

Fig. 7 illustrates the mapping of logical channels to transport services for a packet telephony application comprising three LC's—two for audio and one for application-layer signaling. The audio LC's are multiplexed onto a transport service access point (TSAP) that uses best effort datagram transport service. The signaling LC is mapped to a reliable byte stream transport service. This illustrates the function of multiplexing application data over available transport services. LC's are mapped to TSAP's and multiplexing is achieved by allowing LC's to share a TSAP. On the receive side, the LC layer demultiplexes data arriving on a given TSAP. Implementation details are provided in Section VII.

Demultiplexing data requires that messages include a header identifying their intended destination LC. TOPS defines a header format which is used to encapsulate audio packets and provide LC identifiers along with other fields specific to packet telephony. The design is simple and does not make assumptions about the network layer used. Each packet carries a block of audio samples. An application reads a block of audio samples from an A/D converter, encodes them using a negotiated coding format, adds an encapsulation header, and arranges for transmission. The receiving application reassembles packets, performs delay adaptation, and writes the block of decoded audio samples to a D/A converter. Since the size of the audio blocks must be fairly small due to delay constraints, the TOPS encapsulation scheme is designed to contain only essential information in the header thus minimizing overhead. The encapsulation format is generic and may be used for control messages and other media data.

The format of the encapsulation header is shown in Fig. 8. An 8-bit logical channel ID (LCID) allows demultiplexing of LC's that share a transport connection. This is followed by a 7-bit sequence number which is used for reassembly and delay adaptation. Each packet contains an 8-bit coding field in the header which defines the compression and coding

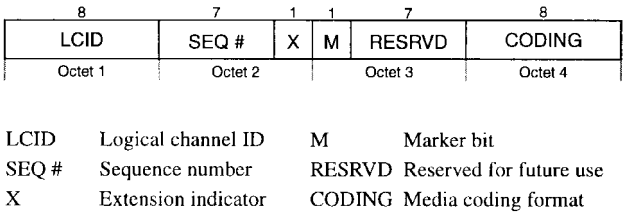


Fig. 8. Packet header format.

format used by the audio data in the packet. This field allows the coding format to be changed on-the-fly without the need for a separate control message exchange. This basic set of fields provides the minimum set of information needed for audio encapsulation. Flexibility is achieved through use of a header extension bit (X), allowing an application to provide additional header information, possibly using type-length-value fields. For example, it can provide timestamps for intermedia synchronization.

The encapsulation scheme also supports synchronization of generic control operations for a media LC by using a marker (M) bit in the header. As an example of the use of this bit, consider the synchronization of audio packets with related control packets, assuming that the control and audio LC's are not multiplexed together. This could be used for example to enable switching to a new set of Huffman tables for use with the audio decoder. Since the control messages may experience different delays than the audio data, a mechanism is needed for synchronization. This is achieved by setting the marker bit in an audio packet indicating to the receiver that a control packet was transmitted and that the action it specifies must be performed before the received audio data may be used. The receiver is required to acknowledge the receipt of the control packet. Once acknowledged, the sender clears the marker bit in subsequent audio packets. On detecting this transition the receiver sends an acknowledgment to indicate the end of synchronization. This handshake has the restriction that only a single control operation can be outstanding at any time. It also requires sequential delivery semantics for the messages on the control and data LC's.

VI. Conferencing for Packet Telephony

Conferencing of multiple call participants is an important service that must continue to be supported as telephony evolves to packet networks. This section explains the key issues for providing efficient conferencing for packet voice and explains how conferencing is supported in TOPS.

A. Issues

The principal processing operation in conferencing is mixing the voice signals. This can be performed in a decentralized fashion at each end-system or centrally at a packet voice bridge. Decentralized mixing is widely used for Internet conferencing [12], where each end-system sends (and receives) audio packets using a single IP multicast address for each data stream. Received voice packets are mixed with those from other sources that have the same playout time. Each end-system must have enough link capacity to receive streams from each source, and computing power sufficient for mixing. To reduce network traffic and the amount of computation required, the use of source silence suppression is advocated. However, even with silence suppression, it is possible that the network traffic generated can exceed the capacity of some receivers.

In centralized mixing, a voice bridge receives data from all sources, performs the mixing, and sends mixed stream(s) back to each end-system. This is beneficial in situations where end-systems have low access bandwidth or are incapable of performing mixing operations. It can also be used to access sources located behind a firewall, by mixing at an application-level gateway, e.g., in an RTP gateway [17]. Sources send voice packets to the bridge using unicast. The bridge generates a composite stream by summing samples from each source. It is desirable for the mixer to send the composite stream on a multicast address in order to reduce network traffic and the overhead on the bridge, but this has some problems with echoes as discussed below. Use of centralized mixing also introduces a single point of failure and a concentration of network traffic at the bridge.

Whether using a centralized or decentralized approach, the basic process of mixing is the same and involves three steps: decoding, thresholding, and summing of samples. These are performed after compensating for the delay jitter of packets arriving from each source. The task of summing involves assigning a weight to each voice source, adjusting the weighting to be zero for those not speaking in order to avoid an objectionable buildup of room background noise in the mixed signal. This is best performed by having each source suppress the generation of packets during silence periods.

Conferencing in packet networks can introduce problems with "echoes" of the voice signal that adversely affect the perceived voice quality. In general, a voice bridge must support echo cancellation. There are three sources of echo: electrical echo in the PSTN, echo produced by the mixing process, and acoustic echo at the end-systems. This section does not discuss acoustic echo cancellation, which must be performed at an end-system when audio output to a speaker is picked up by a microphone. It is assumed this is solved using a handset, headphones, or by canceling acoustic echo locally.

Unlike packet phones, which carry the transmitted and received signal via separate paths, PSTN phones introduce electrical echo due to the *hybrid*[2] circuit interfacing the telephone line to the network. The effect of the hybrid is to echo signals that are in transit toward the phone from the remote talker. Since it will not be known whether voice that travels from the PSTN to a packet voice bridge has had echo cancellation applied to it, the bridge must provide echo cancellation for the PSTN lines in the conference. This cancels the echo of the signal going toward the PSTN hybrid, ensuring that a packet phone talker does not hear his/her own voice echoed off a PSTN hybrid circuit. If the bridge is colocated with the PSTN gateway, it can take advantage of the echo cancellation that the gateway must perform to support 2-party voice calls.

[2] A hybrid is used to convert between a 2-wire and 4-wire telephone circuit. A 2-wire circuit sends voice in both directions over a single wire pair; a 4-wire circuit uses a separate pair for each direction.

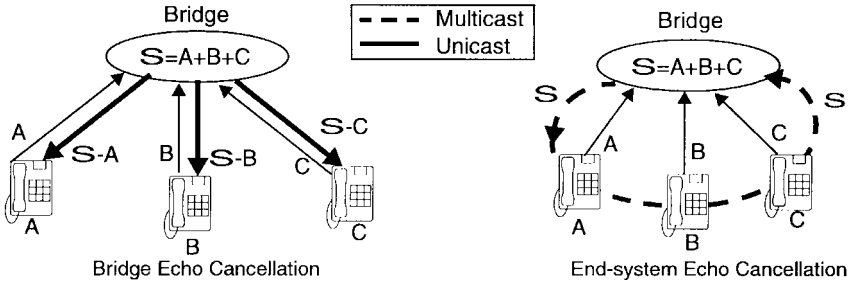


Fig. 9. Echo cancellation of mixer echo.

Echo due to mixing occurs when a mixer just sums the signal received from each participant and sends it back to everyone involved, causing each speaker to experience an echo caused by receiving a delayed version of his own voice (this is not present in the decentralized case since the local signal can be played out as soon as it is generated and then omitted from the mixing process). A mechanism known as echo suppression is commonly used in speaker phones, which cut off the speaker output when the microphone input is active. However, this prevents speakers from hearing other participants. A better solution is to use unicast from the bridge to send each speaker a mixed signal that has his own contribution subtracted, but this is less efficient in terms of network utilization and bridge operation. An alternative is to use multicast to distribute a signal containing all active speakers and rely on each speaker to subtract his own contribution. This demands extra buffering and significant processing overhead at each end-system in order to compare locally generated samples with those arriving in the mixed signal. These two approaches are illustrated in Fig. 9.

Conferencing also raises issues of setup and control that are more complex than point-to-point connections. For example, some users may desire secure conferences with explicit access control (tightly coupled), while in other circumstances less rigorous control over who is allowed to participate may be preferable (loosely coupled). A tightly coupled model maintains state about the conference participants in order to support centralized mixing or provide admission control for private conferences. A loosely coupled model does not maintain participant state, and might be used when the number of participants is large, or when it is infeasible or unnecessary to have control over who can participate in the conference.

B. Conferencing in TOPS

The TOPS architecture supports both centralized and decentralized mixing, along with tightly and loosely coupled control. Directory service and conference control servers are used to manage information about conferences. Conference control is accomplished using an expanded set of ALS signaling messages.

Section VI-A described the problem of mixer echo that occurs in conferences involving a voice bridge. This happens when a bridge just sums the signal received from each participant and sends it back to everyone involved, causing each speaker to experience an echo caused by receiving a delayed version of his own voice. TOPS solves this problem by having the bridge use a combination of multicast and unicast for sending downstream traffic. It takes advantage of the fact that only a few sources are typically active at a time, and all sources are expected to perform silence suppression. It works as follows. The voice bridge sends a composite mixed stream using multicast to all end-systems. In addition, for each active source, it cancels out the contribution of the source from the mixed samples as before, and sends the resulting stream to the source using unicast. An end-system that is also a source ignores samples received on the multicast address and plays out the stream received on its unicast address. The number of downstream channels from the bridge is therefore limited to one more than the number of simultaneous active sources (in some conferences such as lectures, there is only one active source). The number of streams received by an end-system is limited to at most two. This approach represents a "middle ground" between having a single stream distributed using multicast (placing significant additional demands on end-systems to subtract their signal), and using multiple unicast streams (which is less efficient in terms of network utilization and bridge operation).

Fig. 10 shows the various entities involved in conferencing.

Packet phones of varying processing capacity and access bandwidths may be involved. The directory service supports registration of conference names and provides associated addressing information, including multicast addresses and/or the address of a conference controller. A conference controller performs control functions for tightly controlled conferences, including support for capability negotiation and policy-based access control when this is desired. A packet voice bridge implements centralized conferencing on the packet network, while a conventional voice bridge may also be used in the telephone network. Finally, a PSTN gateway (optionally with a built-in voice bridge) can be used to support hybrid conferences involving both packet phones and PSTN phones.

Loosely coupled conferences are similar to current Internet conferences and do not use a centralized controller. End-systems obtain addressing information for conference media streams from the directory, avoiding the overhead of an additional message exchange with a conference controller. The directory service sends multiple transport-layer addresses per call appearance in the query response. Each address corresponds to a media channel, and is typically a multicast address. The directory response also includes an indication to the caller that the regular ALS invitation phase can be skipped.

Control functions for tightly coupled conferences include creating conferences, configuring conferences, participation in

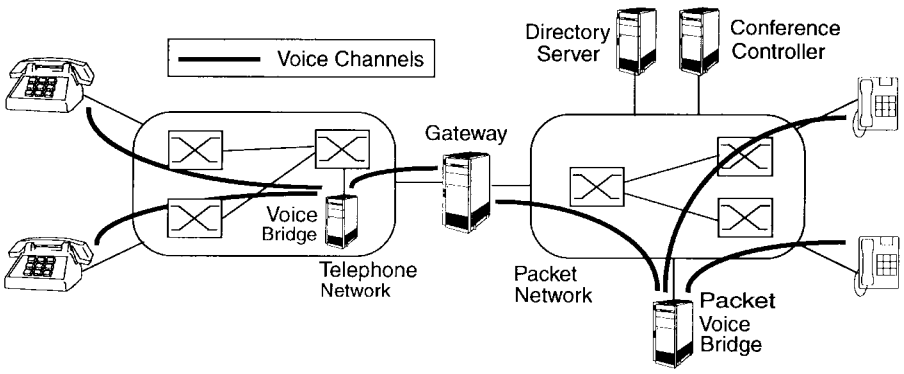


Fig. 10. Components of the conferencing architecture.

TABLE III
ADDITIONAL ALS MESSAGES FOR CONFERENCING

Type	Function
CONFIGURE	conference configuration request; specify parameters
CNF_ACK	reply, indicating conference state established
MODIFY	specify desired changes to the conference state, e.g. add party
MOD_ACK	reply, indicating changes from a modify request
END	notify intention to terminate conference
END_ACK	acknowledge conference termination; delete state

a conference, and being invited to join one. The relevant ALS message are summarized in Table III.

1) *Conference Creation:* This can be viewed as a provisioning step. A conference is created with a distinguishing name (DN), and the network and transport addresses of a conference controller are registered in the directory entry for the DN. It is possible to have multiple conference controllers that are registered, and returned as call appearances in the call-handling profile in response to a query. Some basic access control can be performed by the directory service using the query handling profile mechanism.
2) *Conference Configuration:* After a conference is created, a configure operation is required to set parameters for the conference in the controller. Configuration is typically done by the conference initiator who sends an ALS CONFIGURE message to the controller. The message contains the DN of the conference and of its initiator. The message may also include a key for the purpose of authentication. In the case when a voice bridge is used, the message contains its unicast address, and the multicast address on which the mixed voice packets are to be sent downstream from the bridge. For the purpose of access control, the initiator may specify a list of authorized participants. Information on the types of media, and the set of valid encodings for each are also specified.
3) *Conference Join:* A participant who wishes to join the conference sends an ALS INVITE message to the conference controller. This is exactly the same message that is sent to initiate a two-party voice call. Thus as far as the end-system is concerned, joining a conference is equivalent to calling another user. The invite message contains the caller DN and the set of logical channels and associated media that the caller is prepared to handle. The conference controller checks if the caller is an authorized participant. If so, it returns an ALS RESPONSE message containing the set of logical channels and associated addressing information required by the caller to set up the media and control channels. Typically, this includes a unicast LC for the upstream data and a multicast LC for the mixed downstream data. The response message to the caller also contains the media format and other control information.
4) *Add Party Operation:* This message allows a party to be called and added to an ongoing conference. An end-system sends an ALS MODIFY message to the conference controller indicating that a new party should be added to the conference state. The message includes the DN of the party to be added, and the DN of the conference. The controller calls the new party using the same mechanism as a two-party voice call. Once the call gets established, the called party becomes a participant in the conference.

VII. IMPLEMENTATION

A packet telephony system has been implemented to gain experience with our architecture. The initial focus for voice transport is native ATM, since it provides support for quality of service and offers good delay characteristics. Control information, such as directory access and ALS, is exchanged over native ATM or classical IP-over-ATM.

The LC layer is implemented as a user-level library and exports a synchronous receive and send API to the application. An LC *send* call results in a corresponding call to the associated TSAP. An LC *receive* call returns immediately if a packet is present in the corresponding LC receive queue. Otherwise, it blocks reading the LC receive queue. Packets that arrive on TSAP's are demultiplexed to an LC receive queue based on a logical channel ID (LCID) in an LC header. For applications that do not wish to block, a UNIX-like select call is provided for LC's. We are currently exploring an API that uses buffer pointers to eliminate unnecessary copy operations.

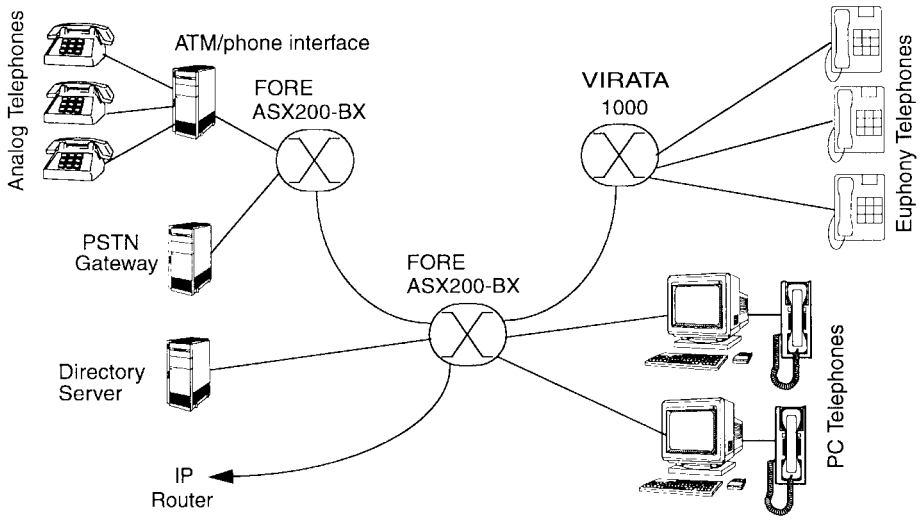


Fig. 11. Current packet telephony testbed.

A. TOPS Terminals

The prototype system consists of a set of packet telephones and servers interconnected by an ATM network infrastructure, as shown in Fig. 11. There are three types of telephones: PC's running a telephone application, custom-built ATM telephone appliances, and analog phones which are controlled and interfaced to ATM by a PC. A server provides a gateway to the PBX (and hence PSTN), and a directory server is used to resolve queries to ATM addresses.

The PC telephone is implemented as an application over Linux. The hardware configuration consists of a Pentium-based PC with full-duplex Gravis Ultrasound and SoundBlaster audio cards sound cards and a Zeit-Net 155-mb/s ATM card. The application uses the open sound system (OSS) API [16]. It supports a variety of audio coders including G.711, G.723.1, G.726, G.728, and G.729. Both UDP as well as native ATM AAL5 are used as audio transport protocols. In addition, the software provides a graphical X Windows user interface for call control, including invitation, release, call transfer, etc. A user-level LC library and ALS signaling have been implemented over the Linux ATM API [1].

A custom-built ATM telephone appliance has been designed and is being used experimentally. It is based on the Euphony processor [5], [6] developed by AT&T Labs and LSI Logic. Euphony is a highly integrated RISC processor that incorporates signal processing functions, a 25-mb/s ATM interface, a digital audio interface, and all the system logic required to build a complete system (e.g., DRAM/SRAM controller, five DMA channels, counter timer, etc.) into a single VLSI device. The ATM telephone hardware consists of a Euphony, SRAM, Flash, A/D and D/A, VART, and a 25-mb/s ATM physical-layer device. Due to its high level of integration, the entire system is packaged in a standard POTS-like telephone case. The software running on the ATM telephone consists of the telephony application, the VxWorks real-time operating system, and ATM signaling code. The ATM telephone augments the traditional POTS telephone user interface, which consists of a handset, an off-hook switch, and a 12-button keypad, with two additional buttons and three LED's. Features such as LCD screen, mouse, or keyboard may be connected to the base unit through a serial port on the back of the ATM telephone. For example, it is used for a prototype 10-in color touch screen with a keyboard and mouse, allowing access to advanced services. Fig. 12 shows a photograph of the telephone circuit board and its plastic casing.

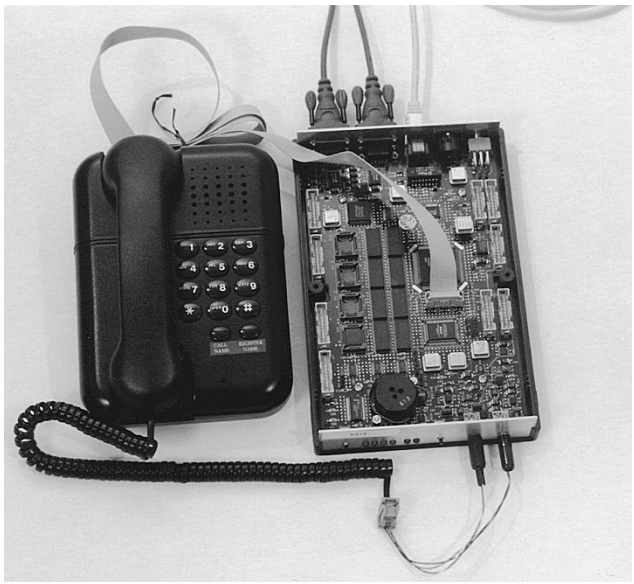


Fig. 12. The Euphony packet telephone.

The third type of device is an off-the-shelf analog telephone connected to a PC-based ATM/telephone interface card [9]. In this arrangement audio data is handled entirely within the interface card, which provides audio A/D and D/A, ATM access, and AAL functions. The card forward off/on-hook and key press events to a control program on the PC which is responsible for generating appropriate dial tones, directory access, and call signaling. Users dial using either unique user identifiers or telephone numbers which are passed to the directory server. Special key press sequences are provided for user registration and deregistration, to take advantage of user mobility and caller-ID-based directory features.

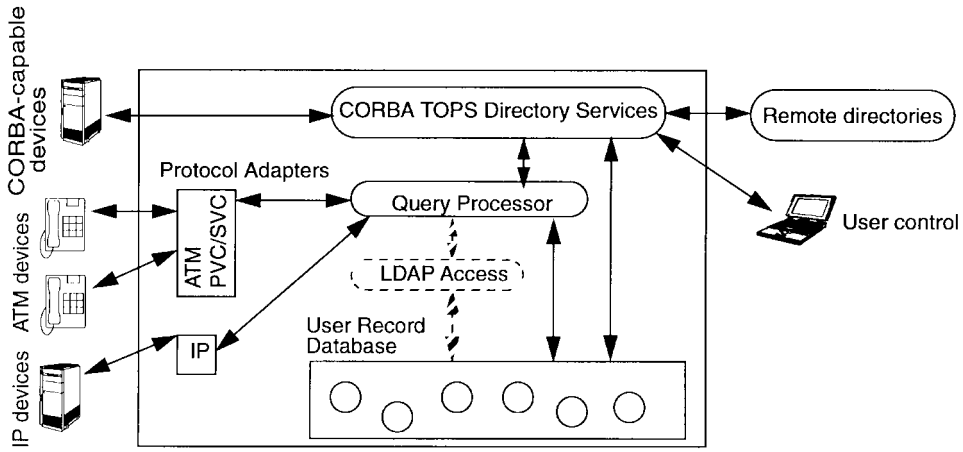


Fig. 13. Directory software architecture.

TABLE IV
DIRECTORY COMMAND INTERFACE

Command	Supplied Attributes	Description
Query	Callee, Caller, Persona, Terminal Caps.	Submits a query to the directory and returns a set of call appearances that can be used by the terminal to locate the user.
Create Update	ESN, Address, Priority, TImeout, Description	Creates or updates a call appearance record when the user or a terminal moves.
Find	First/Last Name, Email, etc.	Returns a list of DNs from the local database that match the user supplied predicates (white pages service)

B. TOPS Gateways

A PSTN gateway has been constructed using telephone interface cards [3]. Multiple PSTN telephone numbers are assigned to the gateway. Calls from the PSTN are completed automatically to a TOPS terminal when translation entries exist in the directory server for the called number. These entries map dialed PSTN telephone numbers (as a DN) to ATM addresses. The gateway itself is also assigned a PSTN telephone number. When this number is called, the gateway provides a secondary dial tone from which the caller may enter the destination name via the telephone keypad.

When a packet telephone receives a CHP that includes a call appearance on the PSTN network, this call appearance contains the address of a PSTN gateway. The initiating terminal then connects to the specified gateway and uses ALS to convey the target PSTN telephone number. The gateway itself also has a distinguishing name in the directory server. When this name is called, the gateway provides a secondary dial tone and the caller is allowed to dial directly.

C. Directory Implementation

The TOPS directory server is shown in Fig. 13. It is organized around a database of user records. Clients can access the directory service using different protocols (the current implementation supports both IP and native ATM). A protocol adapter module produces a directory query from a message (e.g., a UDP datagram or an AAL5 frame) and similarly constructs a response with the results of the query.

Every query is passed to a query processor which identifies a user record to fetch from the database. It uses the persona information in the query (if present) to select the appropriate QHP to execute. The query processing interface is also accessible as a CORBA service [14] for clients that have this capability.

In order to submit a query, the client uses a simple message protocol (shown in Table IV). Each message is of the form: *command_name attrname1=attrval1 attrname2=attrval2*$\cdots$, where *command_name* identifies the action to be taken by the directory. Parameters are specified as attribute name/value pairs.

User profiles can be created and modified through a separate CORBA service. For convenience, a user can modify his profile using a Web browser, which loads a Java applet from the server that supports an interface for profile modification. Another set of interfaces is used to access remote directories in order to fetch user records, update call appearances, etc.

In the future we plan to integrate the TOPS user database within an X.500 directory service. The main advantage is that user records for TOPS can be kept together with user information for other services. In this case, the TOPS directory server will require to access the user record through an X.500 access protocol such as LDAP [10] rather than from local disk.

VIII. RELATED WORK

There are a number of efforts under way to design various pieces of a packet telephony service. These include the session initiation protocol (SIP) [18]–[20] being standardized by the IETF, the ITU H.323 [8] family of protocols; and the call management agent (CMA) architecture specified by the voice over IP consortium [2].

A. SIP

The IETF is developing a signaling protocol for internet telephony known as the session initiation protocol (SIP). SIP supports call management, capability negotiation, and name translation/user location services. SIP is used to exchange messages between *user agents*, which are applications that reside in terminals, or in proxies for user agents. A client invitation may be directed to a user agent or to a proxy server that can interact with a location server before forwarding the invitation to a user agent or redirecting the caller to a different agent. TOPS, on the other hand, integrates all of the location management and terminal mobility functions in a directory service. This allows clients to directly interact with the directory service, rather than going through an intermediary. The TOPS directory service provides more information to the caller about options for reaching a user and the services supported by call appearances, which the caller can use to determine which call(s) to initiate.

SIP combines user location and application-layer signaling functions in a single protocol. By separating these functions, TOPS allows them to be engineered and to evolve separately. For encapsulation, Internet telephony applications based on SIP would be likely to use the real-time transport protocol [17], in conjunction with UDP and IP. TOPS defines an encapsulation that is efficient for transporting real-time data, while retaining flexibility. The TOPS encapsulation influenced the design of the real-time AAL5 protocol being standardized in the ATM Forum [7] and ITU.

B. H.323

The ITU H.323 series of recommendations define protocols for multimedia communications including voice telephony. Within the H.323 framework, H.245 is used for logical channel management and capability negotiation; H.225.0 is used for call control as well as registration, admission control, and status (RAS) functions; H.246 is used for interactions with circuit-switched services, etc. H.323 was initially designed for managing multimedia communications in a LAN environment, and is now being proposed for Internet telephony. In the *gatekeeper routed signaling* mode, H.323 terminals exchange signaling messages with a gatekeeper to set up a call. The gatekeeper supports admission control on a call-by-call basis through RAS exchanges and supports call control through the subset of ISDN Q.931 signaling that is implemented by H.225.0. Once call state has been established via the gatekeeper with the remote terminal, H.245 is used for capability negotiation and logical channel establishment.

H.323 is complicated because it involves a number of different protocol components, a range of options, and a large number of messages to support basic telephony signaling and features. Even a relatively simple call establishment can involve many messages. In the gatekeeper routed signaling mode, the gatekeeper participates in signaling and maintains call state for every terminal registered with it. The primary advantage of this model is that the gatekeeper, since it maintains call state, can be tailored by Internet service providers to support call features, billing, etc. However, gatekeepers introduce concerns about cost and scalability. H.323 supports relatively limited address translation capabilities as compared with the flexibility provided by the TOPS directory service.

C. CMA

The call management agent system (CMA) is an agent-based architecture for supporting multimedia calls proposed at the Voice over IP Consortium. In this system, a call management agent represents an entity that can send or receive calls: a user, an application, a gateway, a server, etc. A client that wishes to place a call uses a *CMA name* of the user he wishes to call. This is used to locate the appropriate CMA server. The client sends a resolve request to this server. The response is a set of *communication terminal specifications* (CTSpecs) that describe the communication terminals that can be used to reach the user.

The CMA architecture is intended to provide services that are very similar to the functions of the TOPS directory service, including the ability for a user to customize the logic in the CMA server that responds to a query. However, it does not address user or terminal mobility, or the use of user personas. The CMA architecture does not address application-layer signaling or encapsulation.

IX. Summary

TOPS enhances existing telephone services. It leverages the combination of packet networking and more sophisticated end-systems to enable a variety of creative new services. TOPS extends traditional directory services to support powerful location management functions. The directory service maps from a user's distinguishing name to one or more terminals, servers, or other entities to which a call should be routed. Users have the ability to customize the translation using a profile which allows them to be associated with multiple terminals, and to screen and route calls addressed to them. By using a name as the basis for reaching a user, the TOPS architecture directly supports both user and terminal mobility.

TOPS incorporates an application-layer signaling (ALS) protocol that supports call establishment, terminal capability negotiation, and session management for the duration of the call. ALS supports multiple simultaneous sessions, which allows traditional services, such as three-way calling, to be easily implemented. ALS supports multiple media streams to be associated with a call. TOPS uses an encapsulation format that is simple and efficient for real-time transport. A logical channel mechanism allows control and media streams to be multiplexed over an single transport association, and to be managed as a single, application-layer context. For example, one or more control channels may be used to manage the constituent media streams of a single ALS context. This allows the application to be isolated from the characteristics of the underlying transport layer (such as a datagram or connection-oriented transport) or network layer. TOPS only assumes that the network supports establishment of end-to-end connectivity between terminals, with the appropriate quality of service to carry the real-time media streams.

TOPS supports both loosely and tightly coupled conference control. It also allows either distributed or centralized mixing of the media streams. In distributed mixing, each end-system is responsible for audio mixing. This requires adequate bandwidth and processing capacity at each terminal. With a centralized mixer, the architecture can accommodate a wide range of terminal capabilities and available bandwidths. We have proposed an efficient means for distributing mixed audio streams to the participants through a combination of unicast and multicast. The tightly coupled control model is based on a conference controller that is notified when participants join and leave a conference. The loosely coupled model builds on network-layer multicast mechanisms where an end-terminal may not be aware of changes in the list of participants. Both models take advantage of the directory service for conference name resolution.

TOPS interfaces packet telephony services with the public switched telephone network through a telephone gateway. The gateway performs conversion from packet to circuit, and acts as a signaling gateway between ALS and traditional telephone signaling.

We believe that mechanisms for authorization and usage recording must be developed that allow service providers to charge users for telephony services. We are presently exploring these issues.

REFERENCES

[1] W. Almesberger, "ATM over linux documentation," [Online]. Available WWW: http://lrcwww.epfl.ch/linux-atm/doc.html.

[2] O. Kahane and S. Petrack, "Call management agent system: Requirements, function, architecture and protocol," IMTC Voice over IP Forum Submission VoIP97-010, 1997.

[3] Dialogics Corp., "More about computer telephony," [Online]. Available WWW: http://www.dialogic.com/company/3602web.htm.

[4] P. Mockapetris, "Domain names: Concepts and facilities," IETF RFC 882, 1983.

[5] P. Z. Onufryk, "Euphony: A signal processor for ATM," *Elec. Eng. Times*, pp. 54–80, Jan. 20, 1997.

[6] ——, "Euphony: An embedded RISC processor for low cost ATM networking and signal processing," in *Design SuperCon97: Digital Communications Design Conf.*, 1997, pp. C112-1–112-16.

[7] A. G. Fraser, P. Z. Onufryk, and K. K. Ram Krishnan, "Encapsulation of real-time data including RTP streams over ATM," ATM Forum Contribution SAA-98-0139, Feb. 1998. [Online]. Available WWW: http://www.research.att.com/~kkrama/papers/atm_98-0139.pdf.

[8] ITU-T, "Recommendation H.323: Visual telephone systems and equipment for local area networks which provide a nonguaranteed quality of service," 1996.

[9] Inno-Media Logic Ltd., "Building telephony switches and systems from 1000 to 100 000+ ports with computer telephony chassis linked over an ATM infrastructure." [Online]. Available WWW: http://www.iml-cti.com/index_doc.htm.

[10] M. Wahl, T. Howes, and S. Kille, "Lightweight directory access protocol (v3)," IETF Draft <draft-ietf-asid-ldapv3-protocol-09>, Nov. 1997.

[11] R. L. Rivest, "RFC 1321: The MD5 message-digest algorithm," Internet Activities Board, Apr. 1992.

[12] M. Handley, J. Crowcroft, C. Bormann, and J. Ott, "The internet multimedia conferencing architecture," IETF Draft <draft-ietf-mmusic-confarch-00>, July 1997.

[13] P. Bhagwat, S. K. Tripathi, and C. Perkins, "Network layer mobility: An architecture and survey," *IEEE Personal Commun.*, vol. 3, June 1996.

[14] Object Management Group, "The common object request broker architecture and specification," rev. 1.2, Dec. 1993.

[15] T. Richardson, F. Bennett, G. Mapp, and A. Hopper, "Teleporting in an X window system environment," *IEEE Personal Commun. Mag.*, vol. 1, pp. 6–12, Sept. 1994.

[16] Open Sound System (OSS/Linux). [Online]. Available WWW: http://www.4front-tech.com/linux.html.

[17] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," IETF RFC 1889, Jan. 1996.

[18] H. Schulzrinne, "A comprehensive multimedia control architecture for the Internet," in *7th Int. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 97)*, May 1997.

[19] H. Schulzrinne, "Personal mobility for multimedia services in the Internet," presented at the European Workshop on Interactive Distributed Multimedia Systems and Services (IDMS'96), Mar. 1996.

[20] M. Handley, H. Schulzrinne, and E. Schooler, "SIP: Session initiation protocol," IETF draft <draft-ietf-mmusic-sip-03>, July 1997.

[21] G. Hjalmtysson and K. K. Ramakrishnan, "UNITE: An architecture for lightweight signaling in ATM networks," in *Proc. IEEE Infocom'98*, San Francisco, CA, Apr. 1998.

[22] ITU-T, "The directory: Overview of concepts, models and service," Rec. X.500, 1993.

Nikolaos Anerousis (S'87–M'96) received the Dipl. Eng. degree from the National Technical University of Athens, Athens, Greece, and the M.S., M.Phil., and Ph.D. degrees from Columbia University, New York, NY, in 1990, 1991, 1994, and 1995, respectively, all in electrical engineering.

He is a Senior Technical Staff Member at AT&T Labs–Research (formerly AT&T Bell Laboratories), Florham Park, NJ. His research interests include network dimensioning and optimization, service pricing, software architectures for management and control of broadband networks and multimedia services. In 1998, he was also an adjunct Assistant Professor of electrical engineering at Columbia University. Most of his research work has been in the area of broadband network management. In 1992, he designed and implemented the first operational performance management system using TMN standards on the AT&T Xunet gigabit platform. He is currently investigating scalable management architectures using mobile agent technology and packet telephony systems.

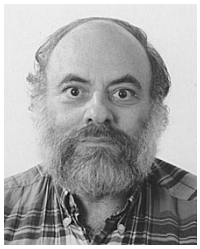
Dr. Anerousis is a member of the Technical Chamber of Greece.

R. Gopalakrishnan received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, India, in 1987, the M.Tech. degree in computer science from the Indian Institute of Technology, Delhi, India, in 1990, and the Ph.D. degree in computer science from Washington University, St. Louis, MO, in 1996.

From 1987 to 1989, he worked as a Research Engineer in the ERNET project at IIT Kanpur where he wrote drivers to interface a variety of network adaptors to UNIX TCP/IP. From 1990 to 1991, he worked as Senior Design Engineer at Wipro Infotech, Bangalore, India, where he was involved in the development of an i860 based workstation focusing on the network subsystem under UNIX SVR4. He is currently the Senior Member of the Technical Staff at AT&T Labs–Research, Florham Park, NJ. His research interests are packet telephony, service differentiation in protocol stacks, and I/O subsystem optimizations for multimedia applications.

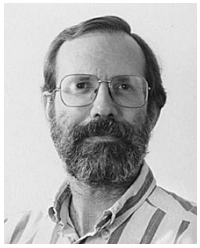
Charles R. Kalmanek (M'93) received the B.S. degree in applied physics from Cornell University, Ithaca, NY, in 1980, the M.S. degree in electrical engineering from Columbia University, New York, NY, in 1981, and the M.S. degree in computer science from New York University, New York, NY, in 1988.

He is currently Division Manager of Networking Research in AT&T Labs–Research, Florham Park, NJ. His research interests include network architecture, protocol design, quality of service, and multimedia systems. Recently, he has been working on broadband access, packet telephony, and IP routing over ATM networks.



Alan E. Kaplan received the B.S. degree in engineering science (physics) in 1968 from the University of Rhode Island, Kingston, and the M.S. degree in mathematics from the University of Michigan, Ann Arbor, in 1969.

He is a Principal Member of the Technical Staff in the Networking Research Department at AT&T Labs–Research, Florham Park, NJ. He was with Bell Laboratories from 1968 to 1997 and has been with AT&T Laboratories since then. His research interests are in the areas of computer networking and telecommunications.



William T. Marshall received the B.S. degree in mathematics from Carnegie-Mellon University, Pittsburgh, PA, in 1974 and the Ph.D degree in computer engineering from Case Western Reserve University (CWRU), Cleveland, OH, in 1979.

From 1978 to 1979, he was Assistant Professor of Computer Engineering at CWRU. He joined AT&T Bell Laboratories in 1979 and is now a Principal Researcher in the Networking and Distributed Systems Research Center of AT&T Laboratories, Florham Park, NJ. His activities and research interests include all aspects of data communication networks, analytic modeling and performance measurement, and operating system design.

Partho P. Mishra received the B.Tech. degree from the Indian Institute of Technology, Kharagpur, India, in 1988, and the M.S. and Ph.D degrees from the University of Maryland, College Park, in 1991 and 1993, respectively, all in computer science.

He is currently a Principal Member of the Technical Staff in the Networking and Distributed System Research Center at AT&T Laboratories Research, Florham Park, NJ. His research interests include traffic management, mobile networking and packet telephony, and video services.

Peter Z. Onufryk (S'87–M'89) received the B.S. degree in electrical engineering from Rutgers University, New Brunswick, NJ, in 1987, the M.S. degree in electrical engineering from Purdue University, West Lafayette, IN, in 1989, and the Ph.D. degree in electrical and computer engineering from Rutgers University in 1998.

He is currently a Technology Consultant in the Networking and Distributed Systems Laboratory at AT&T Labs–Research, Florham Park, NJ. He has been with AT&T Bell Labs and now AT&T Labs–Research since 1987. His research interests include computer architecture, parallel processing, and networking.

K. K. Ramakrishnan (S'78–M'83) received the B.S. degree in electrical engineering from Bangalore University, Bangalore, India, 1976, the M.S. degree in automation from the Indian Institute of Science, 1978, and the Ph.D. degree in computer science from the University of Maryland, College Park, 1983.

He is a Technology Leader at AT&T Labs–Research, Florham Park, NJ. He was with Digital Equipment Corporation until 1994 as a Consulting Engineer. His research interests are in the design and performance of algorithms for computer networks and distributed systems. He is a member of the End-End Research Group, as part of the Internet Research Task Force, and participates in the IETF and the ATM Forum. He has worked and published papers in the areas of congestion control and avoidance, algorithms for Ethernet, FDDI and ATM, load balancing, distributed systems performance, packet telephony, and issues relating to network I/O. He has several patents issued in these areas.

Dr. Ramakrishnan is an Editor for the IEEE/ACM TRANSACTIONS ON NETWORKING and IEEE NETWORK MAGAZINE and is on the editorial board for *Computer Communications Journal*.

Cormac J. Sreenan (M'93) received the B.Sc. and M.Sc. degrees in computer science at University College Cork, Ireland. He received the Ph.D. degree in computer science at Cambridge University, Cambridge, U.K., in 1993, for work on multimedia systems.

He is a Senior Technical Staff Member with the Distributed Systems Research Department at AT&T Labs–Research, Florham Park, NJ. Previously, he was a researcher at Lucent (formerly AT&T) Bell Labs in Murray Hill, NJ. His current research interests include networked multimedia systems, mobile computing, and packet telephony; his general research background is in distributed computing and operating systems.