Carrying voice traffic through an Ethernet local network -- a general overview

by John F. Shoch

June, 1980; revised August 1980

Abstract: Local computer networks have primarily emerged for use in carrying data traffic among terminals, hosts, and network servers. The spoken word, however, remains as an important mode of communication -- in the form of two-way conversations, one-way broadcasts, or non-real time applications (dictation, voice message systems, etc.). Within a building it seems to make sense -- in the long run -- to carry both data and voice traffic on the same network.

In this paper we explore the straightforward ways in which this joint service can be provided on a multi-access bus with distributed control -- on an Ethernet local network. The Ethernet system has proven to be an attractive architecture for carrying data traffic, and can with ease support full telephone service and other voice-based applications. Using this kind of local computer network, one can build a fully-distributed voice system in which there is no need for a central controller or switch. There is, however, a great deal of flexibility in designing such a system, and we examine some of the important dimensions of the design space. Against that background, we then describe a prototype voice system which has been used to carry voice on an existing Ethernet installation -- supporting both real time telephone conversations and a voice recording facility. Finally, we briefly touch upon the question of capacity -- how many telephone users could be supported on one network. Depending upon many different design choices and assumptions, this number ranges from several hundred to several thousand users.

CR Categories: 3.81.

Key words and phrases: Ethernet local network, packet voice, PBX.

This paper is to be presented at the *IFIP WG 6.4 International Workshop on Local-Area Computer Networks*, Zurich, August 1980.

XEROX
PALO ALTO RESEARCH CENTER
3333 Coyote Hill Road / Palo Alto / California 94304

1. Introduction

In the last five years we have seen the rapid development of *local computer networks* -- systems spanning a building or a small campus, and operating with data rates on the order of 1-10 Mbps [Shoch, in press]. These networks generally support a wide range of applications: file transfer between computers, shared use of expensive peripherals, terminal access to time-sharing systems, distribution of electronic mail, and much more.

Most of these applications, however, involve the transmission of digitally encoded data of some form. It is evident, though, that the spoken word still represents a very effective mode of communication. Traditional telephone service is used to support two-way real-time voice conversations, and voice conferencing. In addition, more specialized equipment can be used to support non-real-time voice applications, such as dictation or the exchange of voice messages.

In many buildings we now find that each office has two network connections: one to the local computer network for data, and one to the telephone network for voice. In the long run it seems inevitable that one of those systems will be displaced. We've seen that the telephone system (e.g., a PBX) may not be well suited for handling the increasing need for high bandwidth, bursty data traffic; thus the alternative approach is to consider "voice" as a form of "data," and then support voice communication through the local computer network. This is both a feasible and practical solution.

Among the many alternative designs for a local network, one of the most attractive architectures is the multi-access bus with distributed control -- the Ethernet design [Metcalfe & Boggs, 1976]. Using the techniques of *carrier sense* and *collision detection* with a *dynamic control procedure*, Ethernet installations have for many years supported a wide range of data applications. With suitable techniques for digitizing voice, an Ethernet system can in a straightforward manner support a very rich set of voice applications [Eccles, 1978].

In the sections which follow, we review the Ethernet principles, outline the ways one can carry voice through the network, and explore some of the dimensions of the design space. We then provide a short description of a prototype installation built on an existing Ethernet network; finally, we briefly consider the number of users who might be supported by such a system.

2. Review of the Ethernet principles

The Ethernet packet switching technique makes use of a single, shared broadcast channel. There is no central controller, but rather a distributed control procedure which is used to manage access to the channel.

Any station wishing to send a packet first invokes a *carrier sense* procedure -- it listens to the channel, and *defers* to any other station which is already transmitting. When the channel is idle a station is free to transmit; but if two stations transmit at the same time there may be a *collision.* While transmitting, therefore, a station continues to monitor the channel; a *collision detection* procedure will indicate if the data on the channel do not match the data being sent -- thus showing that a collision is taking place. As soon as the collision is detected, each station immediately shuts down and schedules a retransmission at a later time. To prevent repeated collisions the retransmission time is drawn randomly from an appropriate retransmission interval. To prevent the system from becoming overloaded the retransmission procedure is also *controlled* with a suitable algorithm (such as the *binary exponential backoff* algorithm).

This basic Ethernet control procedure can be applied to almost any broadcast channel -- radio, coaxial cable, twisted pair, and others. It was first implemented, however, in the "experimental Ethernet system": a local network using coaxial cable, and running at 2.94 Mbps. Since then, many other Ethernet derivatives have been proposed or implemented [Shoch, 1979]. Actual performance measurements have indicated that this network performs very well -- total utilization, for example, can approach 98% of the channel capacity.

Note that the shared component of an Ethernet system consists of only a passive coaxial cable; there are no central controllers, no switches, no active components in the line, and no power supplies. (For more background on the Ethernet local network, see [Metcalfe & Boggs, 1976; Shoch & Hupp, 1979, in press; Crane & Taft, 1980; Shoch, in press].)

As a local network, the Ethernet system is just one small piece of a much larger *internetwork* architecture known as "Pup" [Boggs, *et al.*, 1980]. At the moment, this wide-ranging protocol design supports communication among over 1200 host computers, attached to over 30 networks of 5 different types; the system makes use of over 20 *internetwork gateways* which route packets among the different networks.

3. Basic strategy for carrying voice through an Ethernet local network

Until now, the Ethernet installations have primarily supported data communications; the issue here, however is the carrying of voice traffic. In describing a telephone system it is important to distinguish between the *voice transmission mechanism* and the *call processing* or *control* functions (ringing, off-hook, etc.). In the analog phone system it has been necessary to multiplex these functions on the phone line; this will not be necessary in a phone system built around a packet switched network.

In recent years, much of the work on *packet voice* has focused upon the use of long-haul store-and-forward packet switched networks, such as the Arpanet (see, for example, [Forgie, 1975; Cohen 1976, 1977; Gold, 1977; Dhadesugoor, *et al.*, 1980]). These systems are often characterized by medium data rates, as well as substantial (and highly variable) packet delay due to store-and-forward processing.

When outlining a design to carry voice on a local network, we have adopted much of the model used in the long-haul packet voice work. Some of the particular problems, however, are much simpler: local Ethernet systems, for example, have much higher bandwidth, much lower delay, and also lower variability in the delay.

Against this background, then, here is an outline of one strategy for handling voice transmission and the placement of telephone calls (see Figure 1):

---Voice input must be digitized before it is handled in the network -- the analog voice signal is sampled, producing a series of digitized samples. This operation can be done with an A/D and D/A converter, or a *codec*, located at the telephone. One might view each station as a computer which is augmented with a telephone handset, or as a telephone which is augmented with a bit of computing power.

---The station accumulates a series of voice samples into a packet for transmission. This may introduce some modest delay at the sender before the packet is sent; a smaller packet size will require the generation of more packets, but lower delay.

---When the station receives a packet it can play out the samples through the telephone handset.

---Control operations can be handled purely as data, managed by the processing capability associated with the telephone. When a user keys in the number of another telephone extension the regular packet protocols can be used to locate that station and open the connection. In particular, one telephone can use the Ethernet link to directly negotiate a call-setup with another station -- without using any central switch or controller. This style of operation has sometimes been called *distributed switching*, or a *distributed PBX*.

---Internetwork gateways can be used to interconnect the local network with other local systems, or with a long-haul packet switched system; these combinations can be used to directly support packet voice conversations between different networks.

---Telephone conversations with outside numbers will, of course, have to enter the regular phone system. For this purpose, one can use a special station which (like a PBX) terminates a number of trunk lines between the local Ethernet system and the central office; calls from user stations then get routed through this device, which will perform any necessary format or code conversions.

---The availability of a separate digital control unit in each phone provides a means to implement many of the enhanced PBX functions: speed dialing, automatic re-call, call forwarding, etc.

---The availability of a shared channel provides many opportunities for additional enhanced services, such as conference calls and other forms of broadcast voice communication. Support of teleconferencing is straightforward in a broadcast local network [Forgie, 1980; Grandy & Sargent, 1979]; it is also an application which can make good use of *multi-destination addresses* in a local net (also referred to as *group addresses*, or *logical addresses*). Suitable software capabilities will certainly be needed in order to set up a conference call, and manage the arrival and departure of additional participants.

This has only been a very brief summary of the overall architecture, but it serves to highlight some of the basic requirements and design issues in using a local network to carry voice. In addition to our own efforts [Shoch, 1978; Boggs, *et al.*, 1980], other researchers have been pursuing similar approaches. At the Mitre Corporation, for example, a general concept study has been done outlining the use of a shared TDM bus to support both data and voice communication in an aircraft [Grandy and Sargent, 1979]; particular emphasis here has been placed on the ease of installation, modification, and reconfiguration associated with bus systems. At the MIT Lincoln Laboratory in Lexington, analysis and simulation have been done of an Ethernet-style system carrying voice [Johnson & O'Leary, 1979]; a real system (sometimes called "Lexnet") is now being implemented.

4. Dimensions of the design space

There are many different design considerations in constructing a system for carrying voice through an Ethernet local network: one could vary the basic channel data rate, the voice encoding technique, and other factors. All together, these factors combine to produce a very rich design space; in this section we examine some of those factors, and the alternatives available in a system design.

Data rate

One of the obvious characteristics of a local network is the data rate. In an Ethernet-style system, selection of the data rate itself is influenced by the length of the cable, and the number of expected taps. The experimental Ethernet, for example, runs at 2.94 Mbps; the follow-on system runs at 10 Mbps.

Portion of the channel allocated for voice

As we've seen, an Ethernet local network might be used to carry both voice and data traffic. In engineering a system one must consider the balance between these two forms of traffic. If desired, one could reserve a portion of the total capacity exclusively for voice traffic -- by limiting the amount of data allowed, or by augmenting the design to include suitable priority mechanisms for voice.

Voice coding techniques

Voice is, of course, an analog phenomenon. To carry analog voice through a digital transmission system requires use of an A/D *coder* at the input, and a D/A *decoder* at the output; this process is usually implemented in a device known as a *codec.*

Broadly speaking, all of the coders work by sampling the input voice signal, and producing digital samples at regular intervals. One of the most common systems used in the voice transmission plant is *pulse code modulation* (PCM): the system samples at 8 Khz, producing an 8-bit quantity per sample. To provide greater dynamic range the input is not quantized in a strictly linear manner -- each sample represents the log of the input; thus, this is known as *log-PCM*. Note that the total data rate required for one voice channel is thus 64 Kbps (or 128 Kbps for a two-way conversation).

A great deal of work has been done exploring alternative coding techniques -- particularly aimed at reducing the bandwidth required to transmit the digital signal, without sacrificing much voice quality. Some of these techniques include adaptive delta modulation (ADM), linear predictive coding (LPC), continuously-variable slope delta modulation (CVSD), and many others; for a very readable and comprehensive discussion of voice coding, see [Flanagan, *et al.*, 1979], and also [Gold, 1977].

With current technology and coding algorithms of modest complexity, good quality voice can easily be transmitted with coding techniques that require bandwidth from 64 Kbps down to 16 Kbps; lower data rates are feasible with more sophisticated and complex coders, or with additional sacrifice of quality. In designing a voice system to be supported by an Ethernet, one would have to choose an appropriate encoding technique; this decision allows one to trade-off among complexity, data rate, and quality.

Silence detection and digital speech interpolation

The coding technique determines the data rate required when someone is actually speaking; in a circuit-switched system, such as the regular phone network, this bandwidth is permanently reserved in both directions during the lifetime of a call, even when someone is not speaking. Statistical analysis of conversations, however, indicates that this channel is consistently underutilized: most of the time there is only one speaker, and during smaller portions of the conversation there is either "double talking" or silence. Although two channels may be allocated, actual speaking only requires a total data rate of 70-90% of one channel [Brady, 1968].

It is this phenomenon which has allowed more efficient utilization of particularly expensive channels. *Time assigned speech interpolation* (TASI) is used to share underwater cables: a simplex circuit in the cable is only allocated to a conversation when someone is speaking [Bullington & Fraser, 1959]. When a person pauses, the circuit can then be allocated to another conversation; these speech interpolation techniques require some form of *silence detection* which indicates when the speaker is idle. This method, of course, exhibits statistical behavior: at some point, a person may begin speaking when no circuit is free, and the first part of the utterance may be lost due to this *cut-out* or *clipping.*

This technique is also known as *talk-spurt,* and can easily be applied to digital coding and transmission systems: when silence is detected, the digital samples are not transmitted. Silence detection with simple *digital speech interpolation* (DSI) may still clip off the beginning of an utterance, since it may take a moment to recognize that a user has begun to speak [Campanella, 1978]; furthermore, DSI does require a coding technique which can tolerate long bursts with no data. In a packet voice system, using a packet-switched communications network, the source merely stops generating packets during a silence interval. Note that this kind of silence detection eliminates any straightforward synchronization between the sender and the receiver, and additional mechanisms may be necessary to properly preserve the pauses in speech. Time-stamping of packets can be used for several functions in a voice system, such as finding duplicates, and can also be used to ensure that packets are played out at the proper time. Some packet voice systems intentionally introduce a small additional delay at the receiver, to absorb some of the potential variability in packet arrival (or *jitter*).

But the packet-based communications systems manifest an important property which is associated with all digital systems having memory: an initial utterance need not be clipped when someone first begins to speak. If it does take a moment to recognize that speaking has begun, the prior speech samples may still be available in memory, and can be included in the next packet of speech which is assembled. This may introduce a tiny delay before the speech is played to the other user, but it eliminates the annoying truncation of initial words.

If a designer chooses to use one of the silence detection techniques, it can eliminate over 1/2 of the total bandwidth requirement within the system (at the cost of a very small amount of computing at the user terminals).

Error rates

Speech signals are, in general, extremely forgiving: there is a lot of redundancy, and a human listener can tolerate a fair amount of noise or error, and still understand what is being said. After all, a bird flying through a microwave link can generate a "hit" or a "glitch" on the line, yet most users won't even notice. One of the advantages of digital transmission, of course, is the ability to regenerate the original digital input, filtering out the noise which would otherwise accumulate and be amplified in an analog link.

With the use of more sophisticated encoding techniques, however, much of the redundancy is taken out; this can make the digital signal much more vulnerable to errors -- one wrong bit in a tightly encoded signal may dramatically change the resulting output.

Thus, in an overall system design there will be some tradeoffs reflecting the error rates of the underlying channel, and perhaps impacting the choice of encoding technique, or the quality of service provided to the user.

Packet protocols used

Packet switched systems provide another important degree of freedom for the system designer: selection of an appropriate packet protocol for transporting voice packets [Cohen, 1978; Boggs, *et al.*, 1980]. If it is necessary to ensure very high reliability, one can use a sophisticated end-to-end stream protocol that will incorporate error detection and retransmission, and which that will make sure all packets arrive in order. This may, however, contribute to the delay encountered by voice samples before they are presented to the destination. But as we've seen, voice is usually quite forgiving of occasional errors, while inordinate delays do more damage to the quality of service.

Thus, many packet voice systems choose to use a "datagram" or "raw packet" grade of service: the network does its best to deliver each packet, but if one is lost there is no attempt to retransmit -- the samples are already "stale" at this point, and there is no point in playing back an overdue segment of speech. If a packet is damaged, lost, or inordinately delayed a receiver may just skip this packet, or perhaps attempt to sustain some signal from the previous segment.

Many aspects of protocol design for real-time voice were explored in the development of the Network Voice Protocol (NVP) used on the Arpanet [Cohen, 1978]. The real-time nature of voice communication impacts many areas of packet protocol design, particularly the development of suitable procedures for flow control and congestion control [Cohen, 1980].

5. Experience with a prototype system

The experimental Ethernet local network has been in use for more than five years, both inside and outside of Xerox. It uses standard coaxial cable, runs at 2.94 Mbps, and spans a nominal distance of up to 1 Km. Ethernet systems are in use at several dozen locations, supporting over 1200 host computers tied together with the Pup internetwork protocols.

We have now been able to use this system as a testbed for initial experiments in carrying voice traffic. To perform the experiments, several Alto computers [Thacker, *et al.*, in press] were each augmented with a connection to a telephone handset and a modest amount of hardware, along with appropriate microcode and software.

Since the machine has very powerful microcoding capabilities, only a small amount of special-purpose hardware was required: a 4-wire connection to the telephone instrument, a filter (Intel 2912), and a 12-bit A/D and D/A converter. Microcode was used to perform many of the device control functions, as well as the actual voice coding and decoding. The writable control store provides a great deal of flexibility when experimenting with coding algorithms; in this case, however, we chose to use standard PCM coding techniques. Although it could have been run at variable sampling rates, the basic voice digitization rate was the standard 64 Kbps. Furthermore, the microcode included the capability to do silence detection, but it was not enabled.

The software used in each machine had two functions: establishing calls to other stations on the network, and actually transporting voice data; the Pup hierarchy of internetwork protocols was used to support these functions (see Figure 2). Existing resource-location procedures can be used to specify and find the destination station, and a rendezvous protocol is used to set up a connection [Boggs, *et al.*, 1980]. For simplicity in handling real-time voice conversations, a simple voice protocol was built at level 2 of the Pup architecture, on top of the basic datagram layer.

These hardware and software facilities have now been used to support real-time, two-way telephone conversations; these conversations can take place between two stations on the same network, or connected to different Ethernets and interconnected with an internetwork gateway. There are no unusual characteristics to the system, and the overall quality is what one would expect from 64 Kbps PCM played through a telephone handset.

In addition, however, the voice apparatus has been used to experiment with the recording of voice messages on a network-based file server. To support voice dictation and recording we made use of several existing network files servers; compatibility with those existing servers required use of the standard File Transfer Protocol (FTP), layered on top of the Byte Stream Protocol (BSP) in the Pup hierarchy (see Figure 2). This is a fairly elaborate protocol, with full error control, retransmission, sequencing, and window-based flow control. For the voice application, selection of this protocol may be a bit of overkill; but experiments have indicated that the BSP can easily handle data transfer at hundreds of kilobits per second. The network file servers, though, are shared resources which may be simultaneously serving multiple users -- for either voice or data transfer. If special care is not taken, however, heavy load on the shared file system can degrade overall server performance, thus impacting the quality of speech being played back.

All in all, the experiments have gone very smoothly. In this case, though, the stations were really host computers supplemented with telephone terminals. In the future, however, it should be possible to construct a stand-alone telephone instrument which can directly connect to an Ethernet local network: this might be a handset combined with a codec, a suitable microprocessor, and a single-chip Ethernet controller.

6. Estimated channel utilization, delay, and line capacity

To date, experimental use of the Ethernet channel to carry voice has not placed a substantial load on the local network; one of the important questions is the estimated performance with large volumes of voice traffic. Our own measurements of the existing Ethernet installation have indicated that it does perform very well under very high load [Shoch & Hupp, 1979, in press]. With large amounts of artificially generated traffic, total utilization approaches 98% of channel capacity, and behavior is both stable and fair. Thus, the Ethernet access scheme -- carrier sense multiple access with collision detection (CSMA/CD) -- is a very efficient means to share a channel.

At these very extreme loads, however, delay in getting access to the channel can become a significant factor. Measurements indicate, however, that up to a total offered load of about 60%, almost all packet transmissions make it out on their first attempt. Even with total offered load at a sustained level of 90%, 75-80% of all packets make it out on the first or second transmission attempt -- after a very brief total delay.

Capacity of the system for voice usage, in terms of telephones lines supported, depends upon many considerations: base data rate, voice digitization rate, desired voice quality, use of silence detection, possible class-of-service and priority schemes, fraction of total capacity allocated for voice, and models of user telephone behavior during the busy period. Using a wide range of assumptions, our initial studies indicate that a single Ethernet could support anywhere from several hundred telephone lines (with conservative assumptions), up to several thousand users; this is an area of continuing research, which we expect to report in the future.

In addition to this work with actual measurements and analytic assessments, Johnson and O'Leary [1979] at Lincoln Labs have attempted to explore some of the performance issues using simulation. They modelled a system which was fundamentally similar to the Ethernet design, although a bit different in detail from the experimental Ethernet described above (e.g., running at only 1 Mbps). In simulating voice, they were particularly interested in the average queuing delay encountered by a host, and the number of voice samples which had to be discarded when they could not be transmitted in time. They found that "...the average delay is less than 1 ms for values of channel activity up to about .9. This is small compared to a typical vocoder frame time and negligible compared to what is acceptable in a system." Furthermore, "No packets at all were lost with values of A [channel activity] less than .75." A later paper observes [Weinstein, *et al.*, 1980]:

> "The key conclusion was that the CSMA strategy was quite feasible for voice and that a substantial number of off-hook voice terminals could be accommodated without incurring significant transmission delays. The bandwidth utilization efficiency of the CSMA scheme was shown to be equal to or better than the efficiency obtained using a fixed-TDMA technique."

From these measurement and simulation results it should be evident that an Ethernet can easily use a large fraction of its channel capacity to support packet voice, without incurring intolerable delay.

7. Conclusions

The Ethernet local network -- a multi-access bus with distributed control -- is an attractive architecture for local data communication, and can also be extended to support packet voice applications. Starting from work done on other packet-switched systems, one can design a packet voice communication facility for local use, with distributed control. The voice application can take advantage of the Ethernet facilities to provide a very rich range of enhanced PBX-style functions; in addition, the local network can use internetwork gateways to reach packet voice terminals on other networks, or can provide interfaces to the regular switched telephone system.

The basic elements of the design have been validated using the experimental Ethernet, supporting two-way conversations, and also recording/playback applications with a network file server. The availability of a telephone integrated with a microprocessor and a direct Ethernet connection could provide an interesting testbed for further experimentation (see Figure 3).

Finally, we should note that this approach has provided for the often-quoted "integration" of voice and data -- but this has merely integrated the *transmission* of voice and data. Many of the most interesting research questions will only emerge when we strive for a more complete integration of voice and data services: common techniques for creating and editing voice and text, and -- most importantly -- techniques for applying the full range of computational abilities to managing all aspects of voice communication.

8. Acknowledgements

Much of the material in this paper has evolved from conversations with Dan Swinehart, Larry Stewart, and Tim Eccles. The experimental packet voice system using the Ethernet local network was built by Dan Swinehart, Paul Rovner, Larry Stewart, and Jay Israel.

9. Bibliography

[Boggs, *et al.*, 1980]

D. R. Boggs, J. F. Shoch, E. A. Taft, and R. M. Metcalfe, "Pup: An internetwork architecture," *IEEE Transactions on Communications*, com-28:4, April 1980, pp. 612-624.

[Brady, 1968]

P. T. Brady, "A statistical analysis of on-off patterns in 16 conversations," *Bell System Tech. Journal*, 47:1, January 1968, pp. 73-91.

[Bullington & Fraser, 1959]

K. Bullington and J. M. Fraser, "Engineering aspects of TASI," *Bell System Tech. Journal*, 38:3, March 1959, pp. 353-364.

[Campanella, 1978]

S. J. Campanella, "Digital speech interpolation techniques," *National Telecommunications Conference (NTC '78)*, Birmingham, December 1978, paper 14.1.

[Cohen, 1976]

D. Cohen, "On network protocols for speech communication," *Proc. of the 9th Hawaii International Conference on Systems Sciences*, Honolulu, January 1976, pp. 83-86.

[Cohen, 1977]

D. Cohen, "Issues in transnet packetized voice communication," *Proc. of the Fifth Data Comm. Symposium*, Snowbird, Utah, September 1977, pp. 6-10 - 6-13.

[Cohen, 1978]

D. Cohen, "A protocol for packet-switching voice communication," *Computer Networks*, 2:4/5, September/October 1978, pp. 320-331.

[Cohen, 1980]

D. Cohen, "Flow control for real-time communication," *Computer Communication Review*, 10:1-2, January/April 1980, pp. 41-47.

[Crane & Taft, 1980]

R. C. Crane and E. A. Taft, "Practical considerations in Ethernet local network design," *Proc. of the 13th Hawaii International Conference on Systems Sciences*, Honolulu, January 1980, pp. 166-174.

[Dhadesugoor, *et al.* 1980]

V. R. Dhadesugoor, C. Ziegler, and D. L. Schilling, "Delta modulators in packet voice networks," *IEEE Transactions on Communications*, com-28:1, January 1980, pp. 33-51.

[Eccles, 1978]

T. Eccles, *Application of Ethernet principles to telephone voice switching*, unpublished paper, June 1978.

[Flanagan, *et al.*, 1979]

J. L. Flanagan, M. R. Schroeder, B. S. Atal, R. E. Crochiere, N. S. Jayant, and J. M. Tribolet, "Speech coding," *IEEE Transactions on Communications*, com-27:4, April 1979, pp. 710-737.

[Forgie, 1975]

J. W. Forgie, "Speech transmission in packet-switched store and forward networks," *AFIPS Conference Proceedings (NCC '75)*, 44, May 1975, pp. 137-142.

[Forgie, 1980]

J. W. Forgie, "Voice conferencing in packet networks," *International Conference on Communications*, Seattle, June 1980.

[Gold, 1977]

B. Gold, "Digital speech networks," *Proc. of the IEEE*, 65:12, December 1977, pp. 1636-1658.

[Grandy & Sargent, 1979]

R. C. Grandy and C. F. Sargent, Jr., "TDM solutions for internal aircraft communications: C^3 concept for E-4 evolution," *Proc. of the Local Area Communications Networks Symposium*, Boston, May 1979, pp. 251-261.

[Johnson & O'Leary, 1979]

D. H. Johnson and G. C. O'Leary, "A local access network for packetized digital voice communication," *National Telecommunications Conference*, Washington, December 1979, paper 13.4.

[Metcalfe & Boggs, 1976]

R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks, *Communications of the ACM*, 19:7, July 1976.

[Shoch, 1978]

J. F. Shoch, "From local computer networks to integrated communications systems," *Proc. of the ACM Annual Conference*, Washington DC, December 1978, p. 479.

[Shoch, 1979]

J. F. Shoch, *An annotated bibliography on local computer networks*, Xerox Parc Technical Report SSL-79-5 and IFIP WG 6.4 Working Paper 79-1, October 1979.

[Shoch, in press]

J. F. Shoch, *Local Computer Networks*, McGraw-Hill, in press.

[Shoch & Hupp, 1979]

J. F. Shoch and J. A. Hupp, "Performance of an Ethernet local network -- a preliminary report," *Local Area Communications Network Symposium*, Boston, May 1979, pp. 113-125.

[Shoch & Hupp, in press]

J. F. Shoch and J. A. Hupp, "Measured performance of an Ethernet local network," to appear in the *Communications of the ACM*.

[Thacker, *et al.*, in press]

C. P. Thacker, E. M. McCreight, B. W. Lampson, R. F. Sproull, and D. R. Boggs, "Alto: A personal computer," *Computer Structures: Readings and Examples (2nd edition)*, Siewiorek, Bell, and Newell, Eds., in press.

[Weinstein, *et al.*, 1980]

C. J. Weinstein, A. J. McLaughlin, and T. Bially, "Efficient multiplexing of voice and data in integrated digital networks," *International Conference on Communications (ICC '80)*, June 1980.

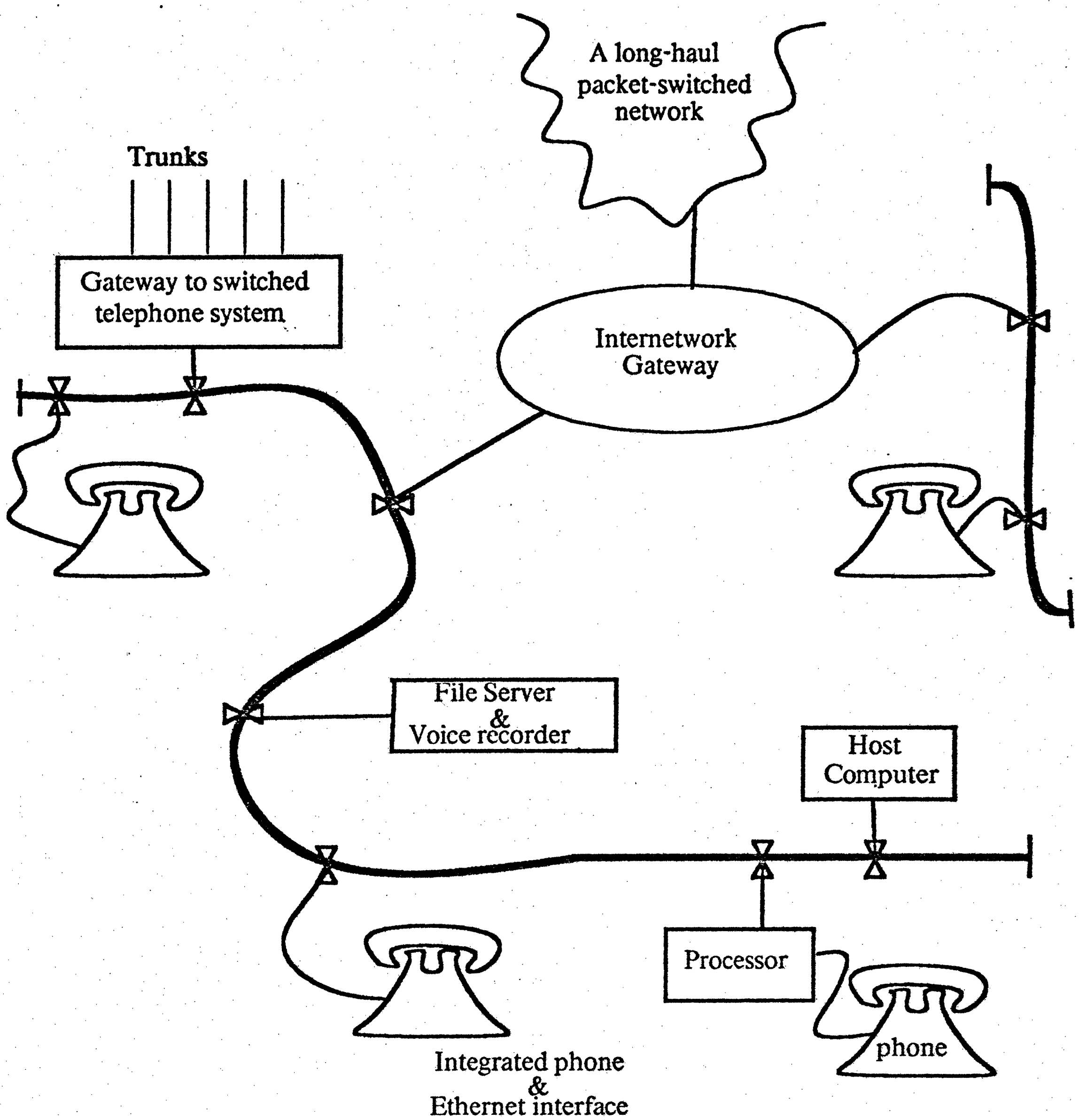

Figure 1: An Ethernet system supporting voice traffic.

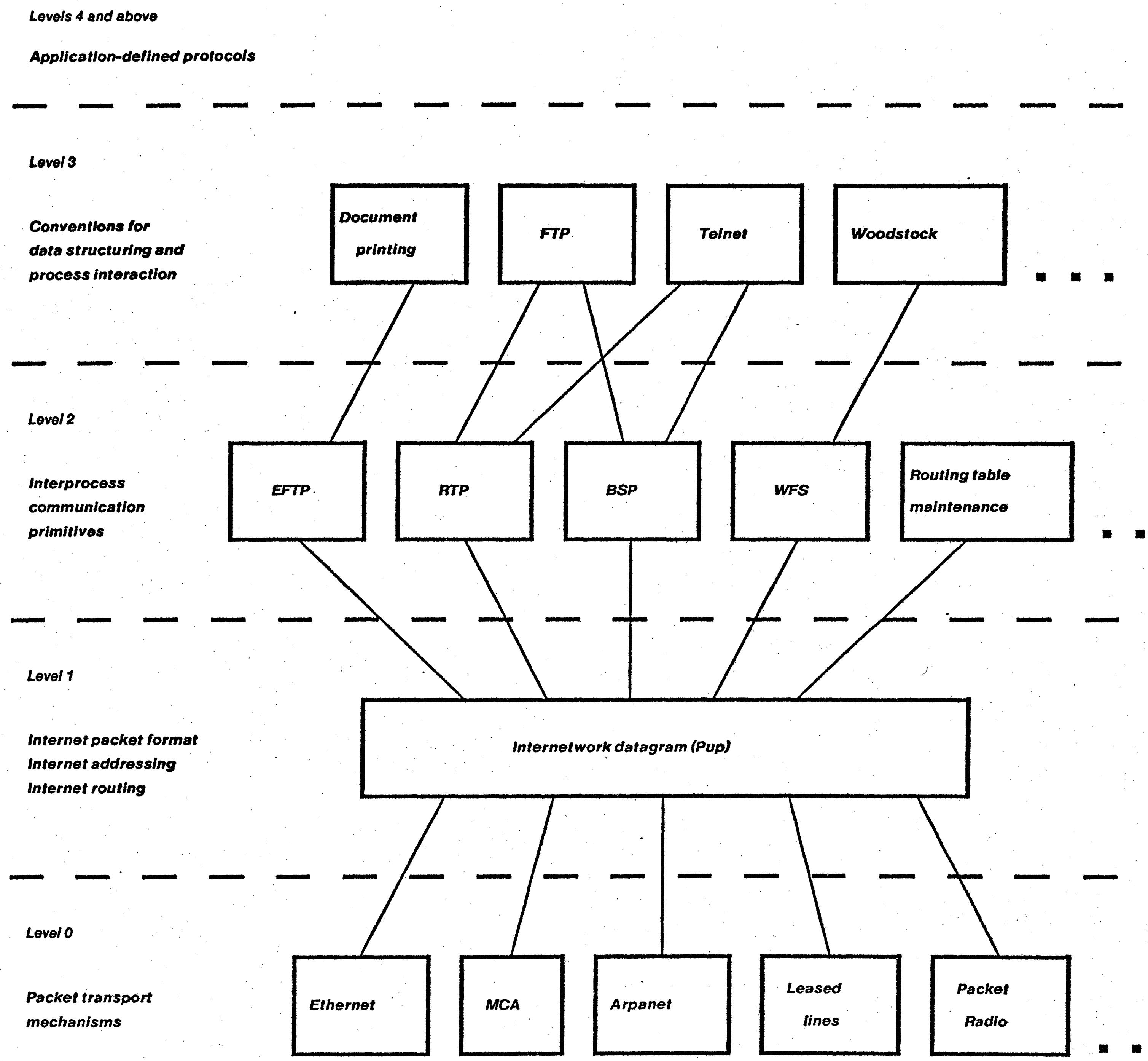

Figure 2: The Pup protocol hierarchy.
(Adapted from [Boggs, *et al.*, 1980].)

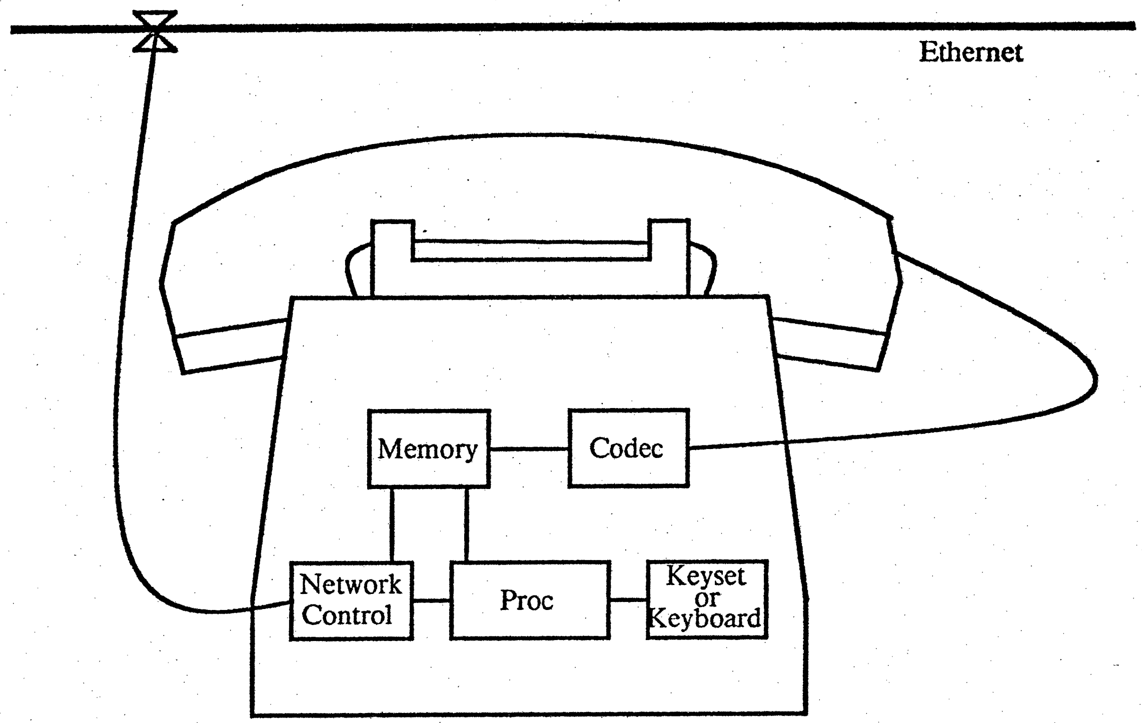

Figure 3: An integrated phone and Ethernet interface.