MESSAGE TECHNOLOGY IN THE ARPANET

T.H. Myer and John Vittal

Bolt Beranek and Newman Inc.
Cambridge, Massachusetts

I. INTRODUCTION

This paper presents an historical sketch of message technology in the ARPA Computer Network (ARPANET). The emphasis is on what's unique about the service that has emerged in the Network environment, and what it's development has to offer for the future.

Computer supported message service predates the development of the ARPANET by a considerable period. One can find origins for the medium in message switching applications such as TWX, TELEX, and private message switches, and in computer "mailbox" programs that allow the users of a time shared computer to exchange messages with one another.

The evolution of message service in the ARPANET, however, broke new ground in two important areas. First, as a communication medium, the network environment presented a new combination of opportunities and problems which have shaped the characteristics of the subsequent, evolving message service. Insofar as the ARPANET serves as a point of departure for future networks, the network message service has important lessons to offer for the future of message technology.

Secondly, from the very beginning, ARPANET message service was a success. It's effectiveness has made it attractive as a model for future communication systems within the Department of Defense, and so it has received considerable research attention. The result is that certain aspects of message technology, in particular the human interface to a message system, have received especially intensive development in the ARPANET environment.

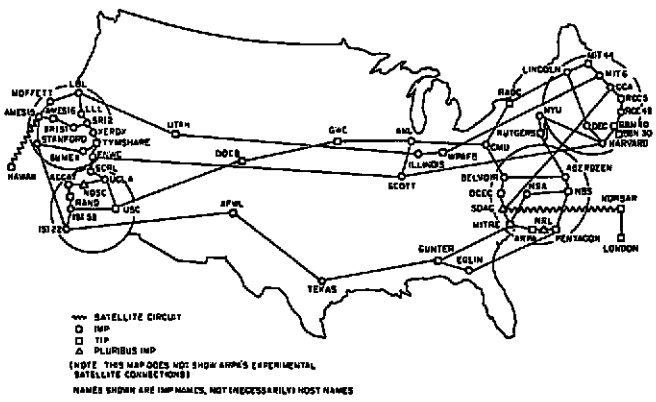
II. THE MESSAGE DELIVERY SYSTEM

A. The Foundation for Message Movement

The ARPANET consists of a "sub network" of specialized communication processors (Interface Message Processors, or IMPs) which are interconnected by communication links, and a collection of "host" computers. Each host is tied into the network by attachment to one of the communication processors, which, taken together, channel the flow of information through the network using the now familiar technique of packet switching. Direct terminal support is also provided by certain of the communication processors (terminal IMPs, or TIPS), making it possible to access host computers from remote sites through the medium of the network. This much is quite similar to the state of affairs in the commercial networks now emerging, where the major current application is to provide long distance terminal access to host computers.

It was a central mandate of the ARPANET, however, to support resource sharing, research on inter-computer communication techniques, and various aspects of distributed computation. Thus, it quickly became necessary to provide a means for various types of information exchange between the network's host computers. An important factor in providing for interhost communication was the polyglot nature of the host population. In July, 1976, for example, there were some 33 different host computer models from eight manufacturers, with a rich variety of operating systems [2]. The following figure provides a topological map of the ARPANET.

ARPANET GEOGRAPHIC MAP, JULY 1977



To support communication among these diverse machines, basic protocols were agreed on, and then it was left up to those in charge of the various hosts to create service programs that would meet the conditions laid out in the protocols.[3]*

The first and most basic of these protocols is the NCP (Network Control Program), which establishes a "virtual channel" between any two host computers, but leaves entirely open the question of what may pass through the channel, or how the transfer is to be effected. In time, various specialized

*These protocols at the host level, of course, made use of still lower levels of protocol and supporting software that provide for information transfer between host and their supporting IMPs, and ultimately for packet switched communication between IMPs.

protocols were established at a higher level that put the NCP to use in accomplishing specific communication tasks.

The most significant of these, from the point of view of this paper, is the File Transfer Protocol (FTP). The FTP was established to support interhost transfer of files be they data or program, binary, ASCII, EBCDIC, etc. To the early message experimenters, the FTP provided a ready-made tool for moving messages through the network. The technique was very simple: establish the outbound message as a file on the host of origin and then use FTP to move that message to the destination host. FTP provided for physical message movement between hosts, but left unanswered the question of specifying, or effecting delivery, to the ultimate human recipient. This question was addressed and partly answered by a specialized "MAIL" extension to the FTP that allows one to send to another host the address of a recipient along with the text that is expected to contain the message.

This, then, is the foundation for all message delivery in the ARPANET. A number of comments are in order. First, the whole thing has a deceptively simple appearance, which arises from the layered nature of communication protocols in the ARPANET. The fact is that underlying the FTP mechanism is a rich structure of communication software that performs a myriad of supporting functions. None of this software had to be considered in developing the message service because it already existed.

Second, the delivery protocol contains very few restrictions. All that's necessary to play the game is that one provide an FTP server process which implements the protocol, and adhere to the very simple requirements of the FTP MAIL extension. The "host" computer is left entirely free to handle in its own way such issues as how messages are actually delivered to a recipient; how outbound messages are "handed over" to the FTP server; and even the format by which a recipient is to be identified.*

As we shall see later, this "laissez faire" attitude is an important and ubiquitous characteristic of the entire Network Mail service.

The third comment, to which we shall also return later, is simply that this delivery mechanism is far from the ultimate that could be wanted.

B. Standardizing the Form of Messages

The FTP delivery scheme provided simple ground rules for moving messages, but left open the question of message form. This is an important question if one wishes to provide software that helps the user read, process, and create messages, for then there must be a convention for the forms of data that can be included in a message. Messages should adhere to formal syntax so that message processors can understand the contents of messages, and perform meaningful functions on those contents. One such function might be to automate the addressing of a reply to a message by copying information from the message being answered.

To this end, an ad hoc committee was formed to standardize the form and content of the "headers" (i.e. that part of the message which adhered to a formal syntax) of messages transmitted between hosts using the FTP. The result was a proposed standard entitled Request For Comments #561 (RFC 561)[8].* This standard met the need perceived at the time, and even though it was unofficial, it was generally adopted. Several points about RFC 561 should be noted:

1. It defined the concept of a message header, and specified the syntax which delimited that header from the actual text of the message. The header is that part of the message (the initial lines of text in the message) which roughly correspond the the header on an inter-office memo, but are readable by humans and understandable by programs.

2. It proposed a standard format for some of the more obvious header items (fields): "From" (who sent the message), "Date" (when it was sent), and "Subject" (what it was about).

3. It proposed a general syntax for all other header items.

4. Its syntax was designed to allow humans to read the headers and text easily and without the aid of special message processing systems.

As message creation, sending and reading services grew in sophistication, the need arose for additional header items which were in RFC 561's "miscellaneous" category. "To" (the primary addresses for the message) and "cc" ("carbon copies" -- i.e. secondary recipients of the message), especially, were generated and recognized by several different message systems. However, there was no specific standard for the syntax of these items.

Recognizing the need for additional standards, a small group was chartered to develop a revised version of RFC 561. The result, entitled "Message Transmission Protocol" and labeled RFC 680 [3], attempted to specify the syntax for some of the additional fields. However, even this specification met the needs of the community for only a short time. As a result many hosts have deviated from the

- - - - - - - - - -

*According to ARPANET standards [3] the recipient may be specified by a NIC (Network Information Center) ident, which is a unique, abbreviated handle assigned to each human user of the network, or a "system ident" which is any identifier that meets the standards of a given host site.

- - - - - - - - - -

*RFC's are the mechanism for publishing or proposing protocols and standards for the ARPANET community.

standard, and those hosts that support message reading programs must cope with all the abberent forms as well as the formal syntax itself, but only if the reading program includes functions which require understanding the contents of messages.

To remedy this situation, yet another group was created. Its product, labeled RFC 724 and titled "Proposed Official Standard for the Format of ARPA Network Messages" [4], should be published in October, 1977. RFC 724 attempts to make the message header fields more legible to human recipients, and in addition, proposes mechanisms such that:

- a user of a mail system can have multiple mailboxes, and mail can be sent to other than a default mailbox
- mail can be sent to named groups of individuals
- mail can be sent to computer mailboxes or people whose only access to the ARPANET is through the U.S. Postal Service
- parenthetical remarks can be included
- messages are easily understood by both people and programs.

C. Joining the Club

Taken together, the FTP (with MAIL extension) and the still evolving rules for message format constitute the total set of standards that govern message delivery throughout the ARPANET. As we noted earlier, the FTP mechanism places few restrictions on a host site. As we have just noted, there is considerable disagreement about the transmission form of messages, and the message service seems to have developed considerable tolerance for those sites that have deviated from such standards as exist.

What this suggests is a rather loosely coupled distribution system. This characteristic is all the more apparent when one considers that there is no central administration for message service. Each host (or cluster of hosts) is free to adopt its own policies and practices. This state of affairs has some interesting consequences.

First, the simple rules for message exchange make it relatively easy for new hosts to "join the club". All that's necessary is to implement the layered series of communication protocols leading up to the FTP with its MAIL extension.

Second, the message standards provide for physical message movement and indicate the contents of messages transmitted over the ARPANET, but not how they are created by or delivered to their human recipients. Specifically, different hosts deliver mail in different ways. As we've suggested previously, the "host" has considerable freedom in designing these functions. Incoming messages can be transferred to their recipients within the host system or simply dumped to a line printer for manual delivery in paper form. Messages delivered electronically can be analysed, indexed and sorted before reaching the user or placed in a simple queue for subsequent processing by programs at the application level. Programs can be provided that assist in outbound message creation, or that function can be left up to the user. As far as we know, each of these approaches can be found at one or another ARPANET site.

D. The TENEX Example

Message sending and delivery within PDP-10 computers running the TENEX operating system [5] is described here as reasonably typical of ARPANET practice. The situation is similar on hosts running the DEC TOPS-20 monitor. Under TENEX, messages are delivered to and picked up from files assigned to each user account, and indexed in a file directory associated with that account.

Each user has one file (given special status by the operating system) that serves as a "mailbox" for that user. The TENEX FTP server is extended to append incoming messages to that file, and this can also be done by other programs accomplishing local message delivery from users of the same host.

Outbound messages are placed, one per file, into the user's file space, and linked into what amounts to a pickup queue by the use of special naming conventions in the file directory. TENEX provides a special daemon program, "MAILER", that provides a bridge between the file space and the FTP server. On a routine, scheduled basis MAILER scans all user directories for queued outbound messages. For each such message, the FTP server is activated to accomplish the physical transfer of the message to its destination, and the message is then deleted from the user's file directory.

The TENEX Operating System provides a minimal level of user service. The task of reading, managing, and creating messages is left up to application level software external to the operating system.* Thus, there is a clear split in the TENEX environment between the delivery service and those programs that make it accessible to the user. This separation has had the advantage of making it possible to experiment with many different message handling programs, without risk of disrupting the delivery mechanisms that must provide reliable service to all users. The ease of creating such programs is enhanced by the simple file system interface between application and system levels of the message service.

- - - - - - - - - - -

*Strictly speaking, such software is unnecessary. The user can create and access messages through general purpose Executive functions, and utility programs. In practice, however, message handling can be done far more effectively through software designed for that purpose.

E. Failure Management

As a distributed system, the ARPANET embodies an approach to reliability that accepts component failure as inevitable, but seeks to minimize its impact or spread through the system. This is accomplished by the loose coupling between component machines, and by providing for failure management as an integral part of all critical software. The result is a system all of whose component parts (Hosts, IMPS, TIPS, Communication Links) can and do fail, but whose overall reliability remains extremely high.

This same approach to reliability has also fared well in the message system: failures of various kinds are expected and tolerated, but for each failure condition there is a well established management procedure. Failure control is all the more important in the message delivery service because of the informality of it's management. Hosts can be added to or removed from the network and user's can come and go without any need to inform all message service users of the changes.

We have enumeratd some principal failure conditions below along with procedures for their management.

- Component failure in the network could cause garbled or aborted message transmission. Component failure is tolerated but such damaging results are prevented by extremely reliable communication techniques.*

- Delivery may be made impossible by a failure in the receiving host, or an addressing error. These failure conditions are treated very much as in the postal service. A reasonable attempt is made to deliver, failing which the message is returned to its sender along with notification of the failure.

  In TENEX, message "return" is accomplished by shifting the blocked message from the outbound pickup queue to a second, "undeliverable mail" queue.** User notification is accomplished by automatic generation and delivery of a short message.

  With address errors, the failure is established immediately. A host outage, however, may be short lived; and so repeated delivery attempts are made until a pre-established time period has elapsed.*

- Failure of the user's own host may occur before delivery can be accomplished. In TENEX, the treatment of outbound messages as files minimizes the impact of such failure, since great pains are taken in the operating system to guard against disk file destruction.

- Messages contain only text (their internal structure is represented by the format and content of the text). This promotes reliability by making it possible to list and manually interpret messages that arrive garbled either through an unmanaged failure condition or the actions of a maverick host.

F. Shortcomings

The mechanisms described here and their predecessors have provided five years of uninterrupted message service on the ARPANET. However, that is not to say that the system is without shortcomings.

- The text treatment of messaes may enhance reliability, but it inhibits the effective transmission of other than textual data and information structures other than the very simplest. In an advanced message system one might well wish to transmit non-text information such as graphic, facsimile or compressed speech data, and constructs such as the linked repeating field groups that are supported by many data management systems. It will take a more efficient, binary encoding technique to meet these needs on a practical basis.

- Although addressee designation is left up to the individual site, the common network practice is a two level structure that specifies the host, and the user or account name at that host. Thus, "JONES@ISIA" would indicate user Jones at the ISIA host.

  This addressing scheme is simple in that it reflects the network structure, and it adopts easily to changes in that structure. However, there is no way right now to determine within the system that a valid network address corresponds to someone other than the user's intended recipient. Users can't really express their intentions about the person to whom they are sending mail. RFC 724 tries to allow for this level of specification, and user programs can interface to data bases which associate network addresses

- - - - - - - - - -

*For example, information is dynamically rerouted around failed communication links, check summing techniuqes are employed throughout to ensure accurate transmission, and a copy of each information packet is retained pending its successful receipt at the next node. See references for a general discussion of failure management in the network.

**Like the pickup queue, this one is also supported by the TENEX file name structure.

- - - - - - - - - -

*This is done by the simple expedient of leaving the blocked message in the pickup queue. Each cycle of Mailer through the file space will then yield a fresh delivery attempt.

with the actual names of people. However, this level of development has not yet been completed in the Network. Several research projects are currently under way and are addressing just these issues.

- Even if one accepts the present address syntax, there are improvements that could be made in the area of address verification. Certain hosts and host clusters support data base functions that supply up-to-date address lists for local users. In these cases, local addresses can be verified prior to transmission. At most sites, up-to-date host tables are accesible to the message service so that at least this component of a network address can also be verified. In an advanced message system it would be highly desirable to provide full address lookup and checking on a network-wide basis.

- In many cases adequate privacy protection is accomplished through the general privacy safeguards of the host computer. This is the case in TENEX, where privacy mechanisms built into the file system protect messages before transmission and after receipt.

  As much cannot be said, however, for sender authentication. The entire message service is riddled with loopholes that make it possible to create anonymous messages, or falsify the identity of a message sender.

- The present delivery service suffers from inefficiencies that should be eliminated in a next generation system. For example, when a message is destined for multiple recipients on a distant host, multiple copies travel through the network, even though all are identical.

III. USER LEVEL MESSAGE PROCESSING

As alluded to previously, supporting the message delivery service at a system level has made it possible and attractive to experiment with user-interface programs to that service. How users specify elements of messages such as the body of the subject, to whom the message should be delivered, and so on, may be completely separated from the mechanism that delivers the message. Many interface programs have been written; several are still under development.

A. Early Informal Efforts

Ray Tomlinson created for TENEX what we believe to be the first two programs to send and read mail in the network: SNDMSG which created and sent mail to other hosts, and READMAIL which made it possible read the mail one had received.

The first SNDMSG was a simple program that prompted the user for a set of addresses and the body of a message. The message was automatically sent. Since then, many enhancements have been made to SNDMSG; however, its basic operation remains effectively the same. Currently, the program is still in widespread use throughout the ARPANET on systems running TENEX or TOPS-20.

Conversely, READMAIL, also in use today, has not been modified to any significant degree for several years. READMAIL lists on the user's terminal all messages which have arrived since a specific date, or since the last time the mailbox was accessed. This was fine while the volume of inbound mail remained fairly low. However, that did not remain the case for long. People in positions which required a large amount of communication with others recognized the limitations of READMAIL. Two such people were Larry Roberts, then the director of the Information Processing Techniques Office of ARPA, and Steve Crocker, an ARPA IPTO Program Manager. They felt that a system was needed which:

1. allowed one to see (survey) one's mail without having to see all of each message, and

2. allowed one to examine selected messages.

What resulted was a program called RD which did just those two things. This was the first step in a series of programs which "understood" the structure of messages. For each message in the mailbox, RD typed out a handle for later reference to the message (the sequential index of the message in the mailbox), the date the message was sent, its originator and it's subject. After a period of time, RD was expanded to include additional functions. This was the first program to provide message surveys and selective output as an ARPANET utility.

Barry Wessler (then also of ARPA) wrote a program in SAIL named NRD (New RD) as a successor to RD. It provided the same functionality as RD, but added the ability to manage additional files of messages by providing generalized filing and retrieval functions, and also allowed users to delete messages and generally "clean up" their mailboxes. However, this program was never publicly released, and did not see widespread use.

Somewhat later, Marty Yonke (at the time associated with ISI) obtained a version of NRD, added functionality, provided a uniform interface to all commands, and coupled in the message creation program, SNDMSG. The result was first named WRD, and then renamed to BANANARD. It was the first program to provide a friendly user interface which included a command driver on-line help facility for each command, and integrated the message reader and sender all into a single program.

Shortly thereafter one of the authors (Vittal), desiring a different user-interface

and additional functionality, created a program called MSG, taking a version of BANANARD as his point of departure. The primary functions that MSG provided over BANANARD were the ability to forward messages to other people, to automatically set up the address fields in message replies, and to provide a user profile that allowed some actions of the system to be determined by the user. Both programs, although they provided both mail reading and creation functions, used SNDMSG for all mail sending functions.

Similar programs have been written for several other hosts. Ken Pogran created a program called "read_mail" at MIT (on Multics) which is functionally equivalent to BANANARD. A version of MSG was written for PDP-11's running the UNIX operating system. Several programs have been written for DEC-System-10's. And others were written for TENEX: for example, Jim Calvin wrote a program called HG which has become functionally similar to MSG but whose primary goal was speed.

All these efforts provided a basis. All are still in widespread use today in the ARPANET. So, even though research efforts are continuing into topics such as user-interfaces, functionality of message systems, integrating message systems into the larger organizational environment, and so on, because there are strong proponents of each mail reading and sending program available today in the network, the ability to change the standards and protocols is greatly diminished. All researchers who want to provide a service for a set of users desiring to interface to other individuals and mail systems on the ARPANET must adhere to the conventions. However, some can and have departed significantly from the original paradigm of just reading and sending mail.

B. More Recent Work

The Hermes group at BBN is responsible for one such effort.[9],[10] Their original aim was to provide an integrated system of mail processing functions for the set of TENEX computers on the ARPANET. The Hermes system, still under active development, allows users to cope with messages, but also provides additional functions for general office support. The general functions it provides are:

- Message creation and editing tools that are tightly coupled into the system. Users can create messages, add notes and retrieval keys to the messages in their files, create special-format messages for special functions, and so on. The message creation tools have been extended to provide general office support. For example they can be put to use in defining and generating form letters for transmission through the Postal Service.

- The ability to generate formatted output, with the selection of message parts to be displayed and their layout under the control of "print templates" that can be defined by the user.

- Tools for storing and retrieving messages. There are two notions here. The message file paradigm remains. However, index structures can be overlayed on a file to support overlapping, named groups of messages. As an example, one might group together all messages on a particular topic. The filing tools have also been extended to support the storage of general records. Hermes has been put to use in managing files of address records, bibliographic data, Navy ship records and product characteristics.

- General text editors applicable to the text contained in messages and other information records.

- Extension of the profile concept considerably beyond the point reached in MSG. The user's profile not only provides for the conditioning of various system functions, but also supplies a storage mechanism for objects, such as the "templates" alluded to above that assist the user in various ways.

MS (pronounced "mizz"), is a message system under development by Dave Crocker of The Rand Corporation for use on PDP-11's running the UNIX operating system.[16] MS is the most recent of the "traditional" message systems, taking MSG and HERMES as points of departure. It also uses a "parallel file" structure in a way similar to Hermes.

C. Exploring New Territory

The systems described above accept the TENEX notion of "mailbox" (simple linear file of messages) as the fundamental starting mechanism. The following two systems are "non-traditional" in the sense that they have modified the mailbox paradigm.

MSGDMS [6] is being developed under the direction of Al Vezza at MIT. It was first developed for the Dynamic Modeling System PDP-10 at MIT, and has since been modified to run under TENEX. It provides all the the standard functions associated with message systems. In addition, it is assumed to be "data-base intensive". Rather than each user having a set of "message files", all messages are incorporated into a single data base, with a file (folder) being ordered sets of pointers into this data base -- only one instance of a message exists in the data base at any time. Also, background processes can be defined which notice events in the data base, communicate with other processes, and so on. MSGDMS has recently been modified to take advantage of a modified Hewlett Packard terminal developed by Information Sciences Institute (ISI) which provides local editing and scrolling functions.

ISI has created a system, named SIGMA [7, 14, 15] for use in testing message processing techniques in a segment of the Navy. As

such, SIGMA is completely independent of the ARPANET delivery service, and reflects independent decisions as to the direction to pursue. There are several features which deserve note:

- The system handles just one kind of terminal, the modified HP terminal mentioned above in the description of the MSGDMS system. It provides local editing, scrolling, and windowing so that response to the user for perceived simple tasks (like deleting a character) is fast.

- All messages are stored in a cache. There is exactly one copy of every message. Users can modify or annotate the message at will, making these annotations public or private.

- Users can access either personal or organizational folders of messages. These folders are really summaries of the messages in the cache, and point to the cache for the actual message.

- Mechanisms are provided for coordinated creation of mail. That is, several people can be working on (editing) the same message simultaneously; mechanisms are provided for coalescing all the comments and modifications into one final version of the message prior its being officially sent.

IV. CONCLUSION

Five years of message service in the ARPANET have demonstrated the workability of a loosely organized system. This leads us to wonder whether the same approach might serve as a model for the broader development of message services in the world outside.

Digital networks are springing up throughout the world with collections of attached host computers. Many of these machines already support some form of "mailbox" program. Right now, so far as we know, there is no way to exchange messages between separate hosts on other than the ARPA network. However, we suspect that there will be strong forces to overcome this limit as user groups expand beyond the capacity of a single host, and as it becomes desirable to exchange messages between separate user groups. We expect that before long interhost and eventually inter-network message exchange capabilities will begin to emerge. The possibility exists, therefore, that message service within and even between these networks could evolve as it has done in the ARPANET -- with minimum control beyond the basic standards and protocols required for orderly message exchange.

One might argue that this is an outlandish proposition, hardly the appropriate scenario for creating an effective, large scale communication service. But, is it really so outlandish as all that? For those who believe that free enterprise yields the best ultimate service to the consumer, this approach would offer free enterprise with a vengeance. For the price of a network connection, any computer proprietor could tap into the global message flow, offering his own brand of end user service at his own price. Competition would encourage the development of cost effective services and foster a lively exploration of new and imaginative kinds of service.

Rather than attempt an artificial separation between communication and data processing, this approach would freely acknowledge their close relationship. Indeed, our research suggests that the most effective services of all might be those in which communication and data processing functions were tightly bound together.

This does not mean to imply, however, that the approach is without problems. The closed community of the ARPANET and its research orientation yield a situation different from what could be expected outside. There is a degree of information sharing and cooperation among host proprietors that has helped the message service develop, but which might be hard to achieve in a commercial setting. The cost incentives and sensitivities in the ARPANET are different from those of the business world.

Beyond these social and economic issues there are technical problems of privacy, authentication, and address structure that must be solved and which may be more difficult to solve in a loose social structure than a single monolithic organization.

Difficulties notwithstanding, however, it is our feeling that the ARPANET experience should be carefully considered as a model for further development. We believe that a laissez faire approach should always be favored until it can be clearly demonstrated that some other approach is necessary.

REFERENCES

1. "Origin, Development and Curent Status of the ARPA Network", Peggy M. Karp, Stanford University. Reproduced with author's and publisher's permission from Digest of Papers, COMPCON 73, the Seventh Annual IEEE Computer Society International Conference, San Francisco, California, February - March 1973.

2. "ARPANET Directory", NIC 36437, Produced and Published by, Network Information Center, Stanford Research Institute, Menlo Park, CA., July 1976.

3. "ARPANET Protocol Handbook", E.J. Feinler and J.B. Postel, Network Information Center No. 7104; Augmentation Research Center, Stanford Research Institute, Menlo Park, April 1976. (NTIS AD A003890).

4. "Proposed Official Standard for the Format of ARPA Network Messages", (working draft) Ken Pogran, John Vittal, Dave Crocker and Austin Henderson, RFC 724, NIC 37435, Augementation Research Center, Standord Research Institute, Menlo Park, May 12, 1977.

5. "TENEX, A Paged Time Sharing System for the PDP-10", (with D.G. Bobrow, J.D. Burchfiel, D.L. Murphy), Third Symposium on Operating System Principles, pp. 1-10, (1971).

6. "An Electronic Mesage System: Where Does It Fit?", A. Vezza, and M.S. Broos. Published in Proceedings of IEEE Symposium on Trends and Applications 1976: comuter Networks, November 17, 1976.

7. "SIGMA Message Service Reference Manual", John F. Heafner, Lawrence H. Miller, Bonnie Arter Zogby. Working Paper ISI/SP-5, University of Southern California Information Sciences Institute, February, 1977.

8. "Standrdizing Network Mail Headers", A.K. Bhushan, K.T. Pogran, R.S. Tomlinson and J.E. White. ARPANET Request for Comments, No. 561, Network Information Center No. 18516; Augmentation Research Center, Stanford Research Institute: Menlo Park, September 1973.

9. Hermes Users' Guide, Theodore H. Myer and Charlotte D. Mooers, Bolt Beranek and Newman Inc., June 3, 1976.

10. "Notes on the Development of Message Technology", by T.H. Myer and D.W. Dodds in Proceedings of the Berkeley Workshop on Distributed Data Management and Computer Networks (LBL-5315). Lawrence Berkeley Laboratory, University of California and United States Energy Research and Development Administration, Washington, D.C., May 1976.

11. "Issues in Message Technology", D. Austin Henderson, Jr., and Theodore H. Myer, Bolt Beranek and Newman Inc., Cambridge, Massachusetts. Presented at the Fifth Data Communications Symposium, Snowbird, Utah, September 27, 1977.

12. "The Role of Informal Communications in Computer Networks", R.P. Uhlig, S.M. Martin and E.S. von Gehren, Pacific Area Computer Network Symposium, Sendai, Japan, August 21, 1975.

13. "Human Factors in Computer Message Systems", Ronald P. Uhlig, Datamation, May, 1977.

14. "Basic Functional Capabilities for a Military Message Processing Service", Tugender, Ronald and Oestreicher, Donald R. ISI/RR-74-23, University of Southern California Information Sciences Institute, May [1975.

15. "An Editor to Support Military Message Processing Personnel," Rothenberg, Jeff. ISI/RR-74-27, University of Southern California Information Sciences Institute, June 1975.

16. "User-Level Functins in MS: A Network-oriented Message System for Personal Computing", David H. Crocker R-2134-ARPA (DRAFT), Information Sciences Dept., The Rand Corporation, Santa Monica, California, December 25, 1976.