Engineering Journal of Qatar University, Vol. 4, 1991, p. 51 - 69.

DESIGN OF AN E-MAIL SYSTEM FOR PERSONAL COMPUTERS

Mahmoud T. El-Hadidi*, Sultan A. Mohammad*
and
Mostafa S. Metwally**
* Electrical Engineering Department, Qatar University, Doha - Qatar
** Ajman College for Science and Technology, Ajman, United Arab Emirates

ABSTRACT

The design of an Electronic-Mail system that can operate in a personal computer (PC) environment is presented. The underlying philosophy is based on a centralized mailbox approach, wherein a dedicated SERVER stores the mail submitted by an originating USER, and then delivers it to the recipient USER upon request. The software components for each of the USER and the SERVER are described in some detail, and the communication protocol governing the interaction between them is also presented.

INTRODUCTION

Conventional mail systems have been among the earliest means of communication between individuals. As technology advanced, not only new forms of communications emerged — such as telegraphy and telephony — but also electronic versions of the conventional mail system were invented. These include: telex, facsimile, and computer-based messaging systems (which is commonly known as Electronic-Mail or E-Mail). Telex systems use specialized terminals — called teleprinters — to exchange information in the form of electronic pulses at the rate of 50-300 bits/s over a dedicated network. Likewise, facsimile systems use specialized terminals — called Fax machines — to transmit texts (and images) in a digital form, but they do so over the public switched telephone network, and at rates that reach 2,400 bits/s (or more). Meanwhile, computer-based messaging systems were initially designed around mainframe computers and minicomputers that are connected to their terminals via a dedicated data communication network or over the public telephone network. Recently, however, E-Mail systems have been configured using the increasingly popular microcomputers (or personal computers), and these are being interconnected through a high throughput multipurpose computer network (Burstyn, 1983).

From a technical viewpoint, E-Mail refers to a network service whereby every user on the network has a network name or address, as well as the capability to compose a message and send it to another user (or group of users), (Redell and White, 1983). Normally, the message is stored on the network server, and the user is notified that the message is waiting for retrieval. If a user is not logged on when a message arrives, the system stores the fact that a message is waiting and notifies the user the next time he logs on.

Clearly, an E-Mail system need not be used only to transmit information that would otherwise be sent through the mail. Typically, it is also used to replace short telephone calls, through the exchange of electronic information back and forth between users. The benefits of E-Mail include:

- Reducing the number of unsuccessful telephone calls;
- Increasing the speed of information dissemination;
- Lowering direct and indirect mailing costs (since no paper, stamps or mailman is needed);
- Enhancing human communication.

Several implementations of E-Mail systems have been proposed (Huffman, 1987). These are classified into two main categories, namely: private systems and public systems. Examples of private E-Mail systems are: DISOSS and PROFS by IBM; ALL-IN-ONE by DEC; and Mail Access PLUS by AT&T. Among the public mail systems are: AT&T Mail by AT&T; MCI Mail by MCI; and EasyLink by Western Union. Private systems are generally proprietary, and this makes their use limited to organizations in possession of the vendor's machines. Public systems, on the other hand, use public networks together with time sharing machines, and as a result they incur large billing costs.

In the present paper, we develop an E-Mail system that can run on personal computers, which are interconnected via local communication facilities. The design philosophy is based on a centralized mailbox approach, wherein a dedicated SERVER stores the mail submitted by an originating USER and then delivers it to the recipient USER upon request. The software components for each of the USER and the SERVER machines are described in some detail, and the communication protocol governing the interaction between them is also presented.

ARCHITECTURE OF THE E-MAIL SYSTEM

In an E-Mail system designed for personal computer (PC) environment, a USER would use his PC as the tool for sending or receiving his mail. By analogy with the conventional mail system, it turns out that there are two basic approaches for structuring the E-Mail system, namely; the distributed mailbox approach and the centralized mailbox approach.

In the distributed approach, a physical (or virtual) connection is established between the sending PC and the receiving PC, so that text files are exchanged between them directly. Figure 1 depicts the topology for such an approach, and shows the information flow pattern between the two PC's; USER A and USER B. Since the mail is delivered to and stored in the memory of the recipient PC, each PC acts as a mailbox in itself. Such an approach derives its name from the fact that these mailboxes are distributed in space.

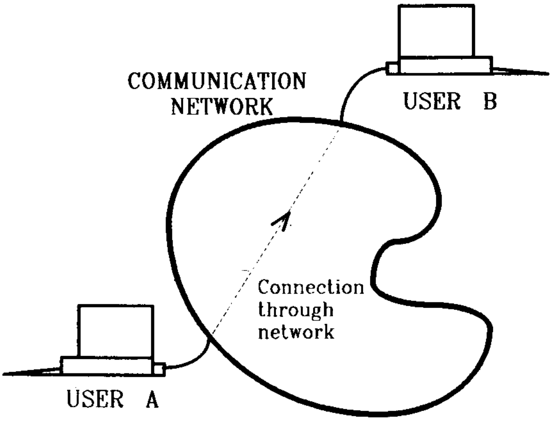


Fig. 1: Structure of the distributed mailbox system

The basic advantages of this approach are:

i - Capability for multiple E-Mail sessions among different pairs of USERs, at the same time.

•ii - High system reliability, due to the fact that failure of one machine will not affect the exchange of text files among other machines.

On the other hand, this approach has a number of disadvantages, including:

i - All PCs should be turned on all the time; if it is desired to send and receive text files at any time.

ii - All PCs should be dedicated to perform the single job of E-Mail activity. Thus, unless a sophisticated software program is designed with proper interrupts and is run in the background, each PC would be in a standby condition to send or receive text files.

iii - The storage capacity of each PC should be increased to allow for the additional space required by E-Mail text files.

On the other hand, the idea of the centralized mailbox approach is to setup a central facility — that resembles the post office — which houses the mailboxes for all USERs. In an E-Mail system, this centralized facility would be a dedicated machine which runs all the time and is equipped with sufficient memory to store the text files for all USERs. Such a machine is called a SERVER and, in a PC environment, can be a PC with a large hard disk.

Figure 2 depicts the configuration of the centralized mailbox approach. From the figure it is seen that USER A sends mail to USER B, by first sending the message to the SERVER. The SERVER would then store the received message in the mailbox (or folder) of USER B, so that it would be ready for retrieval by USER B at a later time.

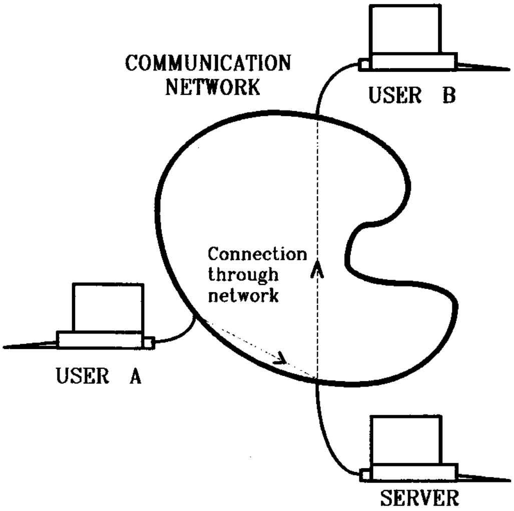


Fig. 2: Structure of the centralized mailbox system

Advantages of this approach include:

i - No need for having all PCs switched on all the time.

ii - PCs are free to perform ordinary tasks (computation, word processing, spread sheet, ..., etc), in addition to their capability to run E-Mail services.

iii - No need to increase the memory overhead of individual USER PC.

The disadvantages of this approach are:

i - Only one E-Mail session can be performed at any time, since any session should involve the SERVER.

ii - The system is less reliable, since it is highly dependent on the reliability of the SERVER.

It is to be observed, however, that both of the above disadvantages can be circumvented by introducing additional SERVER(s) into the system. Such modification can allow multiple E-Mail sessions to be conducted, while reducing the risk of system failure in the event that a certain SERVER should fail.

The foregoing considerations have led to the adoption of a centralized mailbox approach for the design of our E-Mail system.

Remark:

An international standard has been approved for the general E-Mail system which is known as X.400 (CCIT, 1983 and Caswell, 1988). In this standard, a model for the E-Mail system is proposed whose basic configuration consists of a number of User Agents interconnected by one or more Message Transfer Agents. The centralized mailbox structure conforms with the X.400 basic model, since each USER PC corresponds to a User Agent, while the SERVER PC represents the Message Transfer Agent.

COMPONENTS OF THE E-MAIL SYSTEM

1. General Description

According to Fig. 2, the centralized mailbox structure comprises three basic components. These are:

— the USER PC
— the SERVER PC
— the Communication Network

In addition to these physical components, the USER PC should be equipped with a suitable software that would enable the USER to create a text file, to transmit it over the input/output (I/O) port, and to retrieve a text file stored for that USER in the SERVER and receive it through the I/O port. Likewise, the SERVER PC should be provided with a software that enables it to receive text files sent to it by the originator USER and store it in the folder of the recipient USER, to read the contents of a file stored in a certain folder and transmit it to the requesting USER, and to prepare a list of the mail received for each USER. Finally, the communication network should be capable of establishing a physical (or virtual) circuit between the USER and the SERVER, as well as ensuring the error-free transmission of information.

Since the present market boasts with various products for local area networks (LANs), the basic task of designing the E-Mail system reduces to that of developing the software for each of the USER and the SERVER. These two aspects are discussed below.

2. Program Components of the USER Software

The designed E-Mail system has a USER software that consists of six separate modules as shown in Fig. 3. The UMAIN is the main module which is run when the USER software is loaded and executed. It takes the USER into another subprogram, UMENU, which allows him to select one of four actions:

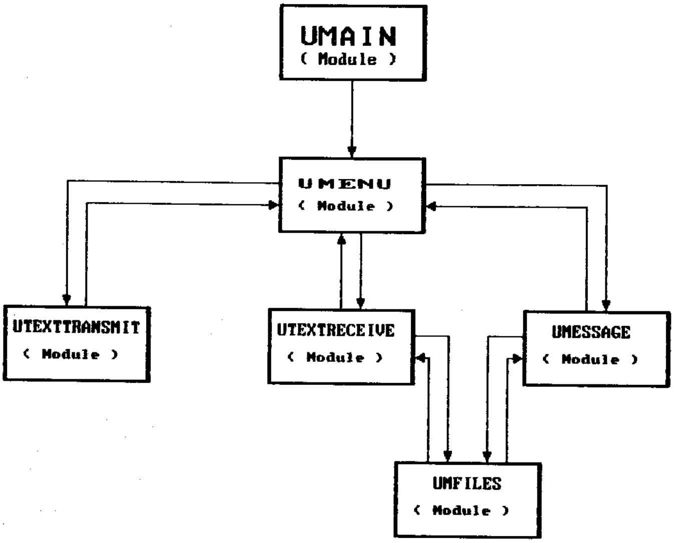


Fig. 3: Interrelationship among various modules of the USER part in the E-Mail software

a) Viewing the list of mail stored for the USER in the SERVER. This is accomplished using UMESSAGE and UMFILES modules.

b) Transmitting an existing text file to the SERVER. This is accomplished using UTEXTTRANSMIT module.

c) Receiving a text file which already exists in the SERVER. This is accomplished using UTEXTRECEIVE and UMFILES modules.

d) Exiting the E-Mail session.

The flowchart representation for one of these modules — the UTEXTTRANSMIT module — is given in Appendix A as an example. Meanwhile, Fig. 4 shows the interface between the USER software and the outside world through the I/O serial communication port. As indicated in the figure, this port contains two buffers, a transmit buffer and a receive buffer, which are accessed, respectively, by the UTEXTRANSMIT and the UTEXTRECEIVE modules.

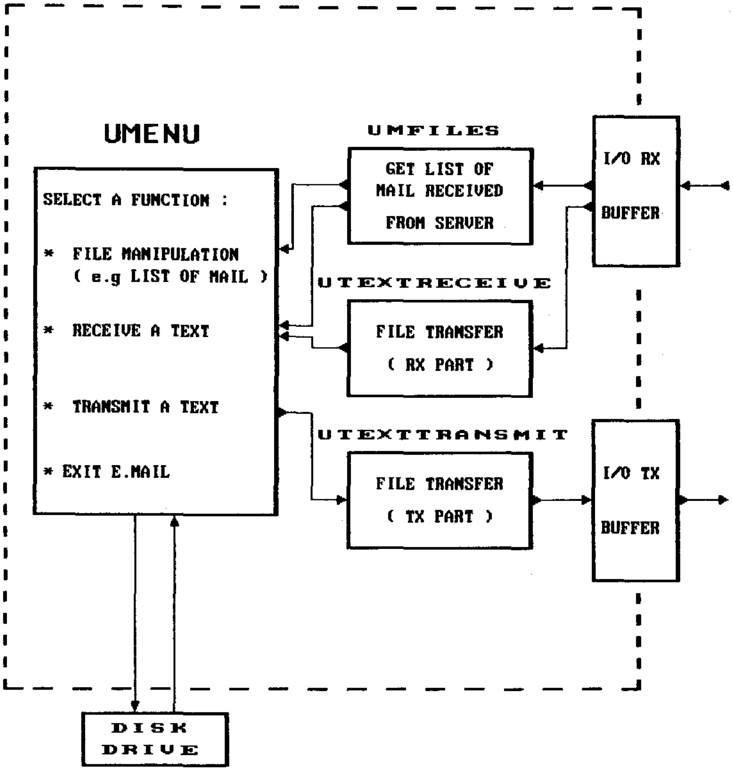


Fig. 4: Data flow pattern within the USER PC and through the serial port interfaace

A built-in feature of the USER software is a special scheme which provides adequate security for the E-Mail System. It consists of using a PASSWORD and an ID both of which should be entered when the USER loads the USER software and is running the UMAIN module. Figure 5(a) shows the sequence of events taking place after loading the E-Mail software and the verification steps which follow.

It is to be observed that the PASSWORD is used to control access to the E-Mail system, while the ID is used to control the use of the E-Mail facilities themselves. In particular:

- If a USER does not enter the correct PASSWORD, he will not be connected to the SERVER through the network. As a result, the SERVER is accessed only by serious USERs and, therefore, its time is utilized more efficiently.

- If a USER enters the correct PASSWORD, he still has to identify himself as an authorized USER. For that reason he enters an ID which consists of 8 characters as shown in Figure 5(b). The entire set of characters should be entered correctly, and recognized by the SERVER, if the USER is to be allowed to use the E-Mail facilities. Once inside the E-Mail menu, only the first three characters of the ID are utilized, as follows:

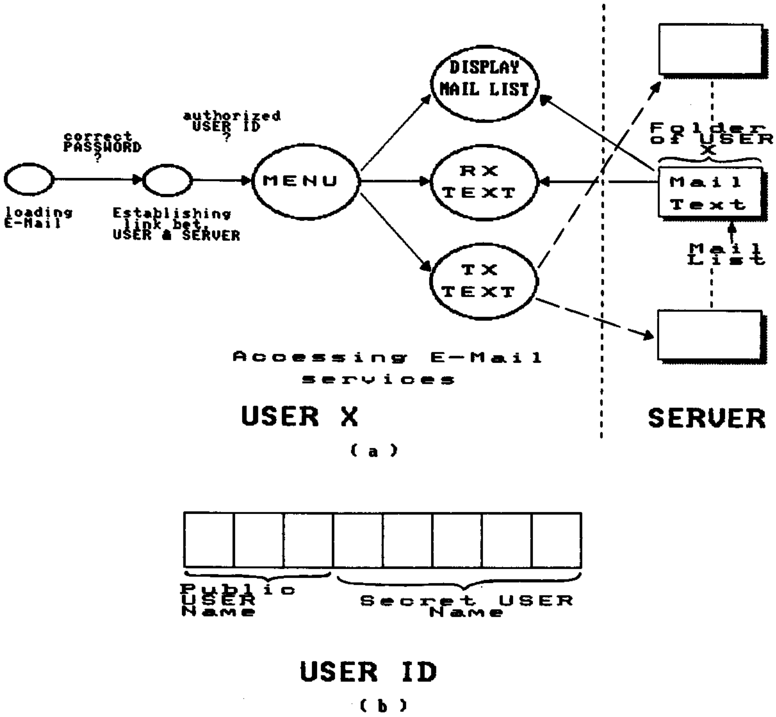


Fig. 5: a) Sequence of operations pertinent to the execution of the E-Mail system; b) Structure of the USER ID.

★ To transmit a text to the SERVER, the first 3 characters of the ID for the recipient are used to identify the destination USER.

★ To receive a stored text from the SERVER, the first 3 characters are used to select the folder for that particular USER, which contains his mail list and his mail text.

★ To identify the source of a text, the first 3 characters appear as part of the text heading to indicate the originating USER.

3. Program Components of the SERVER Software

The second part of the software for the E-Mail system is the SERVER software. Again, this consists of six separate modules which are interrelated as shown in Figure 6. SMAIN is the main module which is run when the SERVER software is loaded and executed. It provides the means of selecting one of several courses of action, the choice of which depends on the first record coming to the SERVER from the USER. There are four actions, namely:

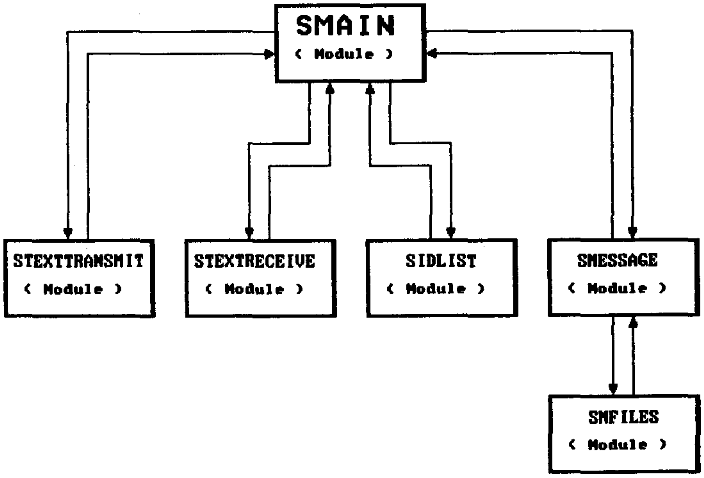


Fig. 6: Interrelationship among various modules of the SERVER part in the E-Mail software

a) Checking the USER ID. This is accomplished using the SIDLIST module.

b) Sending the mail information requested by a particular USER. This is accomplished using SMESSAGE and SMFILES modules.

c) Transmitting an existing file in the SERVER to a requesting USER. This is accomplished using the STEXTTRANSMIT module.

d) Receiving a text file which already exists in the PC of the USER. This is accomplished using the STEXTRECEIVE module.

As an illustrative example, the flowchart representation of the STEXTRECEIVE module is shown in Appendix B. Meanwhile, Figure 7 shows the interface between the SERVER software and the outside world through the I/O serial port. As mentioned before, the I/O port has a transmit buffer and a receive buffer. In the case of the SERVER, these are accessed, respectively, by the STEXTTRANSMIT and the STEXTRECEIVE modules.

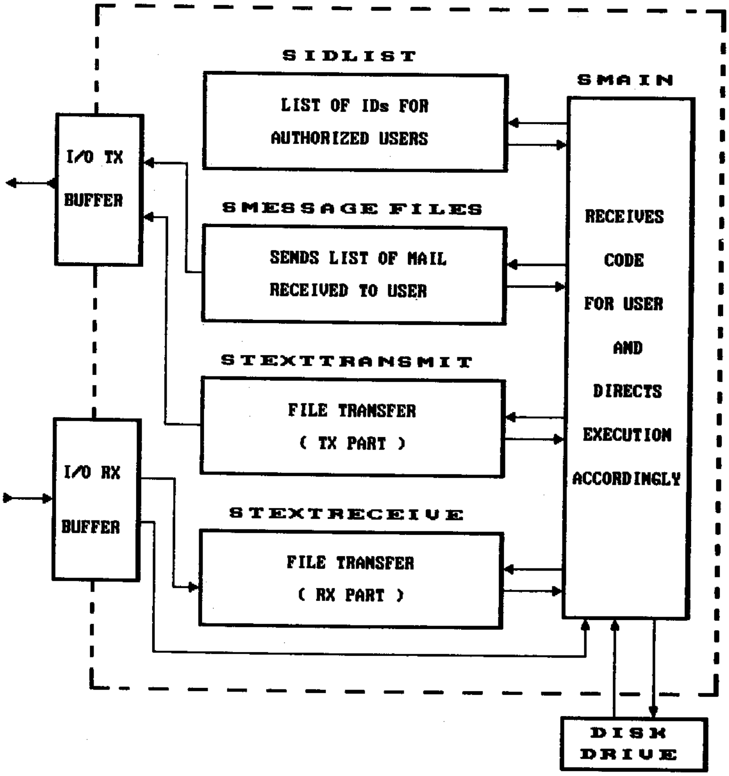

SIDLIST
LIST OF IDs FOR AUTHORIZED USERS
SMAIN
I/O TX BUFFER
SMESSAGE FILES
SENDS LIST OF MAIL RECEIVED TO USER
RECEIVES CODE FOR USER AND DIRECTS EXECUTION ACCORDINGLY
STEXTTRANSMIT
FILE TRANSFER (TX PART)
I/O RX BUFFER
STEXTRECEIVE
FILE TRANSFER (RX PART)
DISK DRIVE

Fig. 7: Data flow pattern within the SERVER PC and through the serial port interface

PROTOCOL FOR THE E-MAIL SOFTWARE

1. General Description

In order to ensure successful execution of the E-Mail activities, there should be certain coordination between the operation of the USER software and the SERVER software. Specifically, these two softwares should cooperate in order to accomplish the following functions:

a) Checking the existence of a communication link between the USER and the SERVER machines;

b) Checking the validity of the USER ID;

c) Selecting a particular activity within the E-Mail menu which is the same on both sides of the link;

d) Performing a record by record transfer between the USER and the SERVER — either for the purpose of retrieving a mail list or for the purpose of sending a file from one end of the link to the other.

It is to be noticed that functions a) and b) above are performed at the beginning of an E-Mail session; that is at the time when a USER calls his E-Mail software. On the other hand, functions a), c) and d) are performed for each selection of the E-Mail menu and, therefore, may be performed several times per E-Mail session.

The signals exchanged between the USER and the SERVER, and the order of their occurrence as a function of time (until an E-Mail activity is successfully accomplished), denotes a PROTOCOL — i.e. the rules of conduct between the two sides. Recalling from the above section (Components of the E-Mail System) that there are three basic E-Mail activities, namely: (i) reading a mail list, (ii) sending a text file, and (iii) retrieving a text file, it is natural to examine the protocols associated with each of these activities.

2. Graphical Representation of a Protocol

In order to facilitate the description of a protocol, we shall represent it in a graphical form. The following guidelines are adopted for such representation:

- Two vertical lines are drawn in the graph; the left hand side line represents the time axis for the USER and the right hand side is the time axis for the SERVER.
- Alongside the USER time axis the information which is entered into the USER PC is indicated. Such information is entered either from the keyboard or else from the USER storage unit.
- The time instant at which either the USER PC or the SERVER PC runs a specific module is indicated by a thick stroke which crosses the relevant time axis. Alongside each stroke is written the name of the module being executed.
- Diagonal lines are drawn between the two time axis with arrows pointing either from the USER to the SERVER or vice versa. The characters written alongside the diagonal lines indicate the information being exchanged. Moreover, the slope of the diagonal line indicates the elapse of time between information transmission and information reception.

3. Protocols for USER-SERVER Communications

Using the above guidelines, the graphical representations shown in Figs. 8-10 can be deduced. Figure 8 depicts the protocol for the USER-SERVER communication in the case of the Mail Review mode. Meanwhile, Fig. 9 shows the protocol for the Mail Receive mode and Fig. 10 for the Mail Transmit mode.

It is to be noticed that the three protocols begin with checking the communication link, then checking the USER ID. The first function involves the transmission of the characters "STA1" and the reception of "OK" for acknowledgement. The second function consists of transmitting "I" followed by the USER ID and then waiting to receive DC2 (i.e. ASCII 18) for acknowledgement. After completing the above two operations, another link checking is performed whose purpose is to make sure that the link is still established. Finally, in all three protocols, the remaining part consists of record by record transmission, and the acknowledgement of the recipient by sending DC2 at the end of each record.

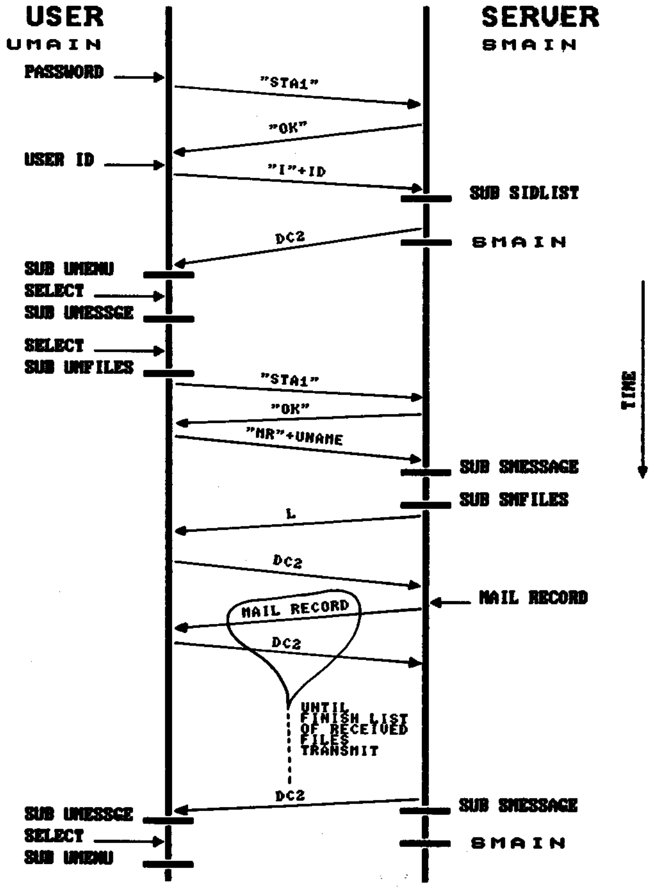


Fig. 8: Graphical representation of the protocol for the Mail Review mode

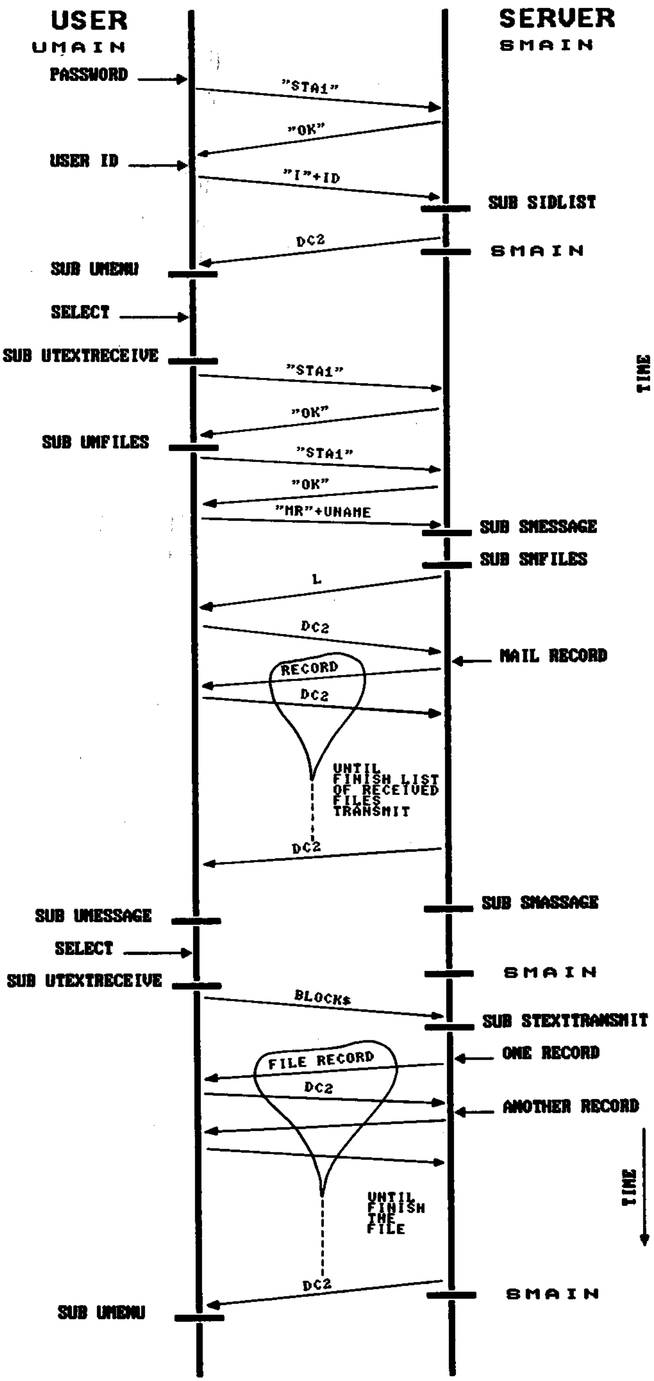


Fig. 9: Graphical representation of the protocol for the Mail Receive mode

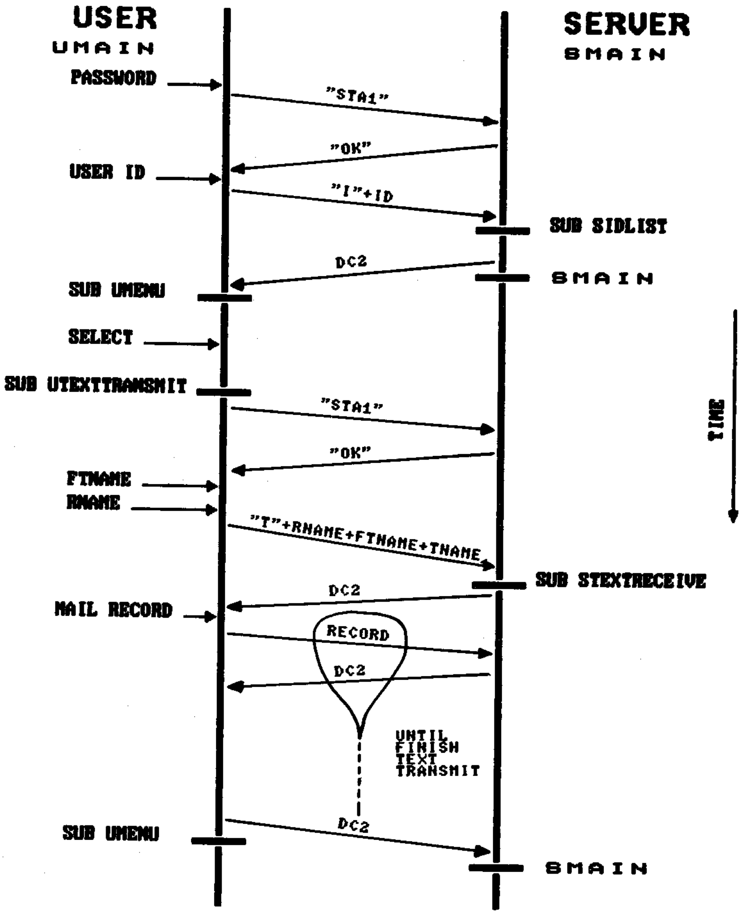


Fig. 10: Graphical representation of the protocol for the Mail Transmit mode

4. Alternative Representation for the E-Mail Protocol

Researchers have developed a number of other representation methods for the purpose of protocol specification and verification. These include the finite state machine model (Bochmann and Sunshine, 1980) and the Petri net model (Danthine, 1977). With the help of these methods, it is possible to investigate the proper operation of complex communication protocols, and check for deadlocks as well as other undesirable conditions. As an illustration, we show in Appendix C the Petri net model for the Mail Transmit mode of the E-Mail protocol depicted graphically in Fig. 10. It is clear that the Petri net model displays some of the details that appear in Appendices A and B, but are not explicit in Fig. 10.

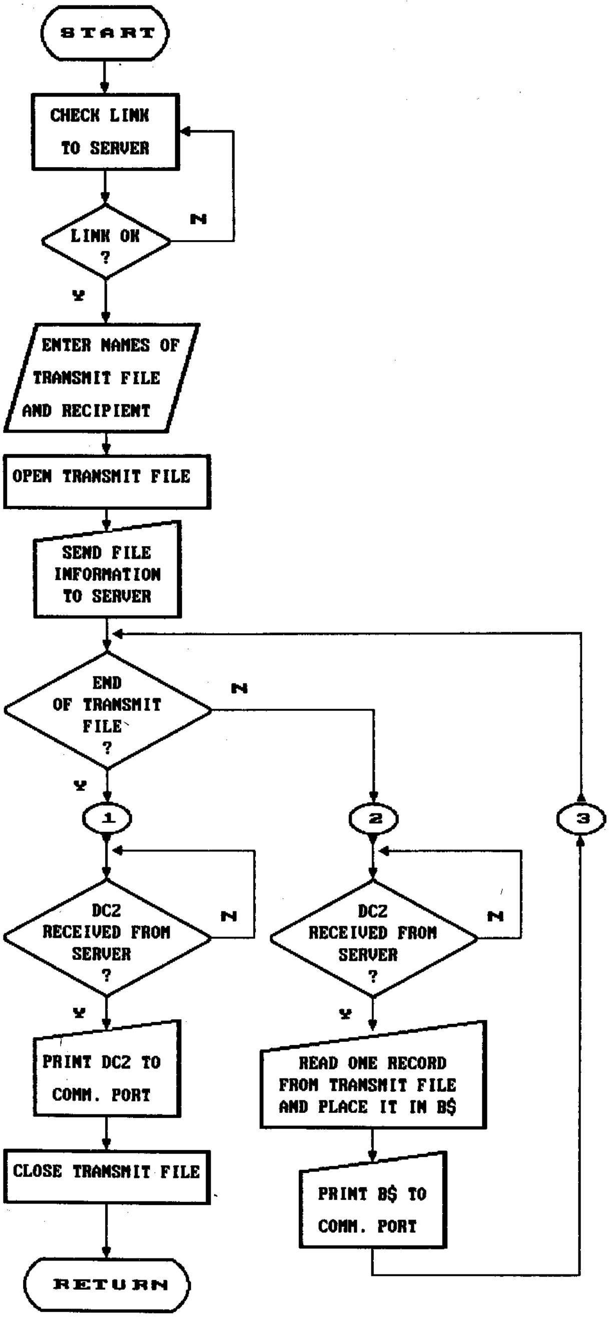
CONCLUDING REMARKS

The E-Mail system described in this paper has been designed to run on IBM compatible PC's using the MS-DOS environment. It has been implemented with an IBM XT as the SERVER and two EPSON PCs as USER's A and B. The communication link between these machines was established using a local area network (LAN) called PLANET (Racal-Milgo, 1983). This LAN has a ring topology, utilizes the token technique for network access, and has a twin coaxial cable as the transmission medium. The USER and the SERVER portions of the E-Mail software have been developed using the QUICK BASIC (Version 4.0) programming language (Microsoft, 1987). The memory requirements for the USER software is 45 Kbytes and for the SERVER software 42.5 Kbytes.

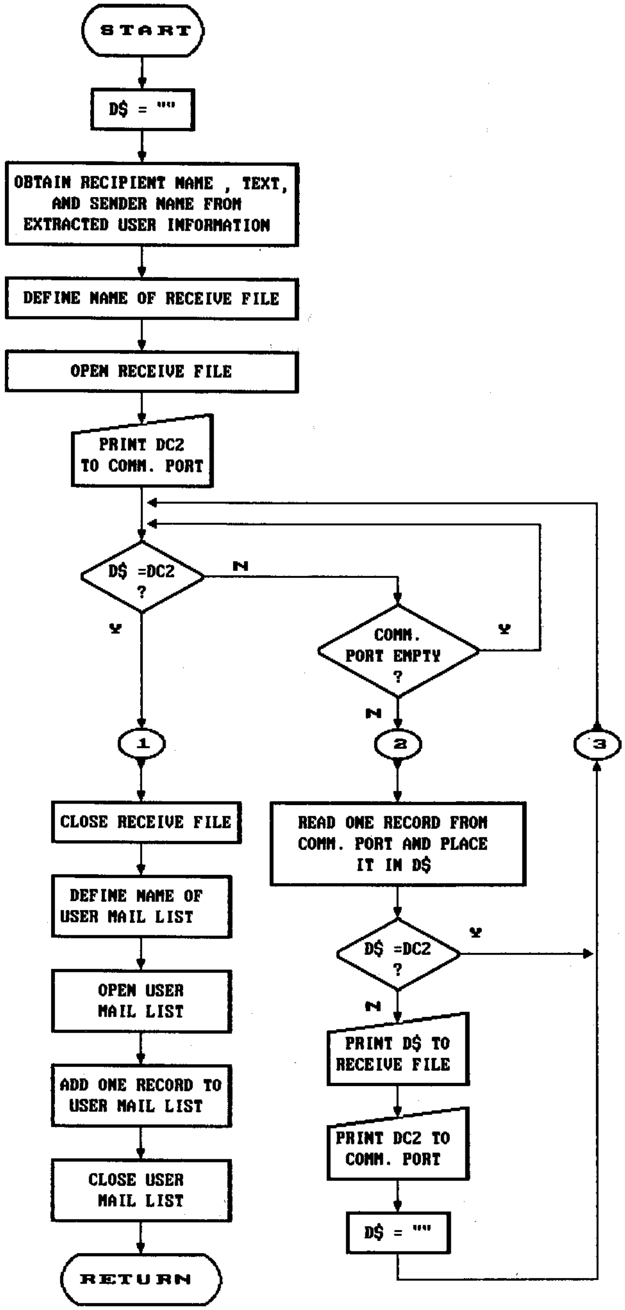
Using the above set-up and software, English text has been successfully exchanged between USER A and USER B. In particular, configuring the serial ports of all personal computers to have a speed of 1200 baud, an ASCII file of size 4214 bytes was transmitted from USER A to the SERVER in 85.15 seconds, while USER B was able to retrieve this file from the SERVER in 94.47 seconds. Meanwhile, for an ASCII file of size 9727 bytes, the transmission and reception times were, respectively, 170.33 seconds and 191.52 seconds. By further increasing the port speed, one can clearly achieve a much faster operation.

It is worth mentioning that the developed E-Mail software would be applicable in a workstation environment without much difficulty. This is because most current workstations run the UNIX operating system and the latter has a DOS shell that enables the execution of DOS applications. It should be pointed out, however, that the UNIX operating system already has a mail utility — called *usermail* — which makes use of the multiuser/multitasking features of the UNIX operating system. The proposed software, therefore, has the advantage of its being implementable in a purely PC environment without the need for the extra power of the workstation.

APPENDIX A
Flow chart of the UTEXTTRANSMIT Module of the User Software

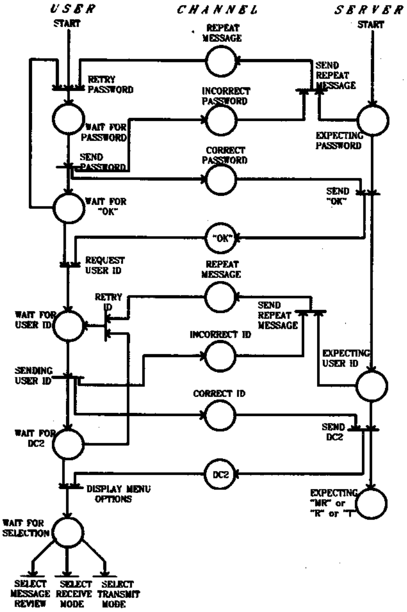

START
CHECK LINK TO SERVER
LINK OK ?
N
Y
ENTER NAMES OF TRANSMIT FILE AND RECIPIENT
OPEN TRANSMIT FILE
SEND FILE INFORMATION TO SERVER
END OF TRANSMIT FILE ?
N
Y
1
2
3
DC2 RECEIVED FROM SERVER ?
N
Y
PRINT DC2 TO COMM. PORT
CLOSE TRANSMIT FILE
RETURN
DC2 RECEIVED FROM SERVER ?
N
Y
READ ONE RECORD FROM TRANSMIT FILE AND PLACE IT IN B$
PRINT B$ TO COMM. PORT

APPENDIX B
Flow chart of the STEXTRECEIVE Module of the Server Software

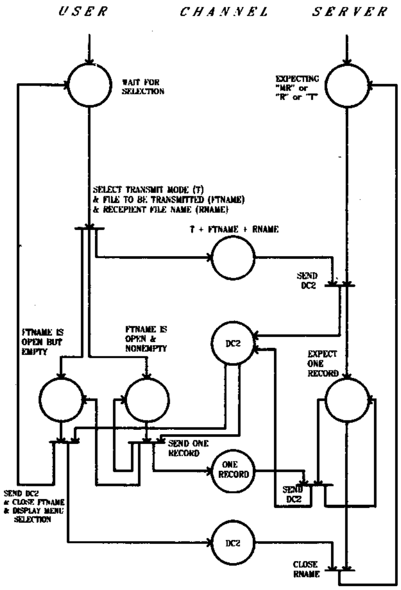

START
D$ = ""
OBTAIN RECIPIENT NAME , TEXT, AND SENDER NAME FROM EXTRACTED USER INFORMATION
DEFINE NAME OF RECEIVE FILE
OPEN RECEIVE FILE
PRINT DC2 TO COMM. PORT
D$ =DC2 ?
N
Y
COMM. PORT EMPTY ?
Y
N
1
2
3
CLOSE RECEIVE FILE
DEFINE NAME OF USER MAIL LIST
OPEN USER MAIL LIST
ADD ONE RECORD TO USER MAIL LIST
CLOSE USER MAIL LIST
RETURN
READ ONE RECORD FROM COMM. PORT AND PLACE IT IN D$
D$ =DC2 ?
Y
N
PRINT D$ TO RECEIVE FILE
PRINT DC2 TO COMM. PORT
D$ = ""

APPENDIX C
Petri Net Model for the Transmit Mail Mode of the E-Mail Protocol

1. Petri Net Model for the Initial Part of the Transmit Mail Mode



2. Petri Net Model for the File Transfer Part of the Transmit Mail Mode



REFERENCES

1. **Bochmann, G.** and **Sunshine, C., 1980,** "Formal Methods in Communication Protocol Design", IEEE Trans. Commun., Vol. COM-28, pp. 624-631.

2. **Burstyn, H.P., 1983,** "Electronic Mail: Evolving from intracompany to intercompany", AFIPS Conference Proceedings, Volume 52, National Computer Conference, pp. 379-383.

3. **Caswell, S.A., 1988,** E-Mail, Artech House, Boston.

4. **CCITT, 1983,** Recommendation X.400 (X.MHS1, REV-19)-Message Handling Systems: System Model-Service Elements.

5. **Danthine, A.A.S, 1977,** "Petri Nets for Protocol Modelling and Verification", Proc. Computer Networks and Teleprocess. Symp., pp. 663-685.

6. **Huffman, A.J., 1987,** "E-Mail — The Glue to Office Automation", IEEE Network Magazine, Vol. 1, No. 4, pp. 4-10.

7. **Microsoft Corp., 1987,** Microsoft QuickBasic 4.0, Vol. 1: Programming in BASIC, Vol. 2: BASIC Language Reference, Vol. 3: Learning and Using QuickBasic.

8. **Racal-Milgo, 1983,** PLANET System Manual TP5200.1.

9. **Redell, D.D.,** and **White, J.E., 1983,** "Interconnecting Electronic Mail Systems", IEEE Computer, pp. 55-63.