"MESSAGE ROUTING SYSTEM FOR ICS (PHASE 2)"

WRITTEN BY:

ODEMBA L.O.R
P64/7297/92

SUPERVISED BY:

DR OKELO-ODONGO

SUBMITTED IN PARTIAL FULFILMENT FOR THE AWARD OF POSTGRADUATE DIPLOMA IN COMPUTER SCIENCE OF THE UNIVERSITY OF NAIROBI.

WE CANNOT SOW THISTLES AND
REAP CLOVER. NATURE SIMPLY
DOES NOT RUN THINGS THAT WAY
SHE GOES BY CAUSE AND EFFECT.

-Napoleon Hill

D E C L A R A T I O N

This project, as presented in this report, is my original work and has not been presented for any University award.

Signed: [illegible]
Odemba L.O.R

This project has been submitted for postgraduate diploma in Computer Science award of the University of Nairobi with my approval as a University Supervisor.

Signed: ______________________
Dr Okelo-Odongo

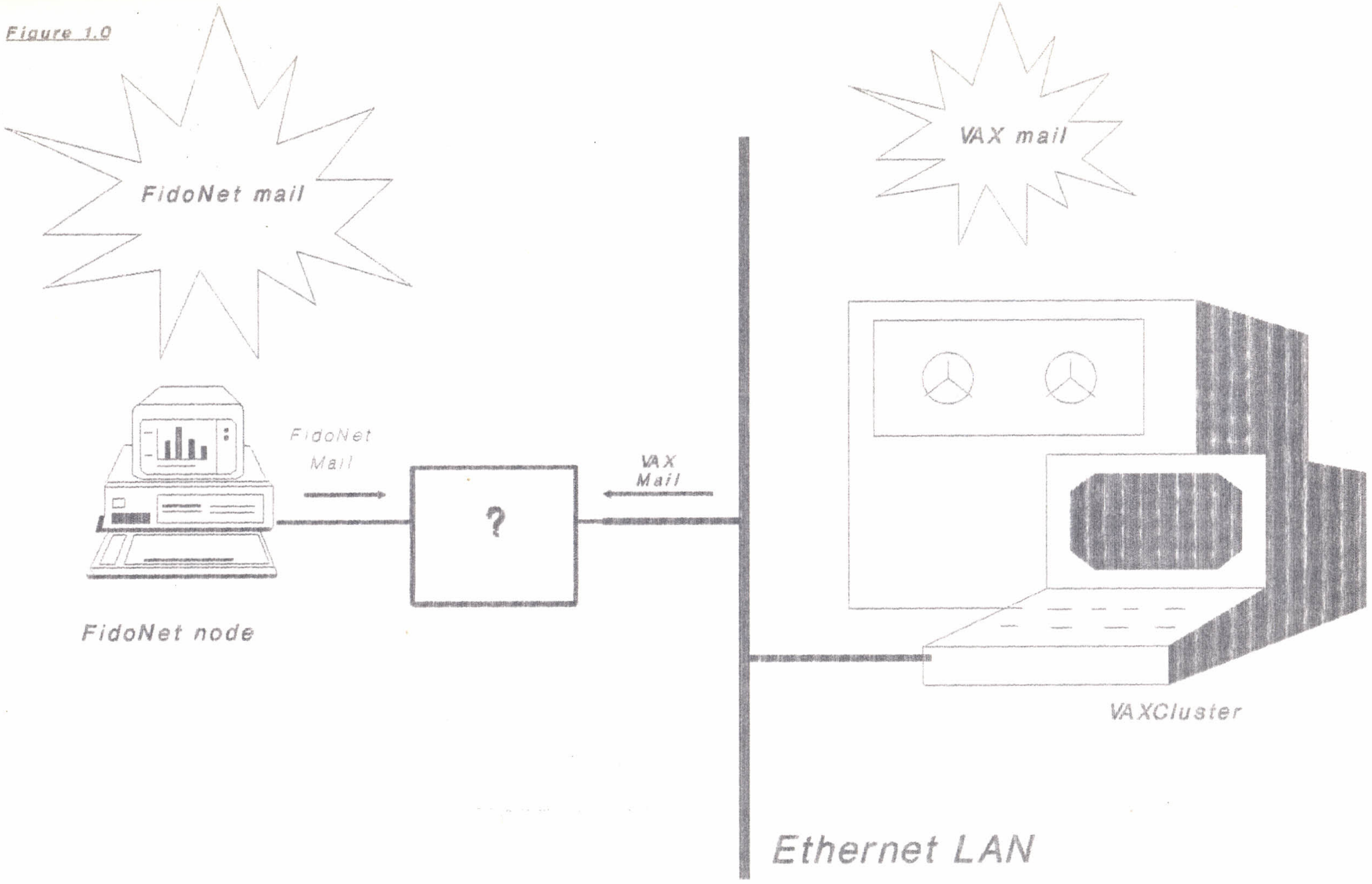
ACKNOWLEDGEMENT

I wish to acknowledge, with thanks, the following people without whom this work would not have been realised.

First and foremost I extend my appreciation to my supervisor Dr Okelo-Odongo for his guidance through the project work; Mr Solomon Odeny for assisting in connecting the required hardware and the PATHWORKS software and further for making all documents I required accessible at the shortest notice; Mr A. Laloui for assisting me in acquiring the necessary hardware.

Last but not least, my appreciation goes to the laboratory technicians at the ICS and my fellow classmates for their company and moral support that went along way in creating a conducive environment.

Figure 1.0



To answer the above question in terms of hardware and software can cost sleepless nights.Apparently it is the objective of this project.

T A B L E   O F   C O N T E N T S

ABSTRACT

The work presented in this report follows an earlier one that covered the overview, analysis, design and a prototype implementation of a system for interfacing the three e-mail networks at the Institute of computer science (ICS) ; namely VAXCluster system, FidoNet and PACSAT networks.

However, the elements of the PACSAT network is left out for the reasons given in the report. The current work therefore focuses on a system for VAXCluster-FidoNet mail transfer and routing. Although the analysis and design of such a system forms a component of the previous work, the author felt that the same should be done here to help clarify, re-design and explain certain areas that may not have been implicitly or explicitly covered.

C H A P T E R 1: INTRODUCTION

This chapter covers the following areas: introduction of the system, the objective and scope of the project and advantages that will accrue after the system is implemented and operational.

1.0 INTRODUCTION

At the institute of computer science (ICS hereafter) , University of Nairobi , it is desirable to interface the three mail networks namely FidoNet , PACSAT and VAXcluster so that mail received on one network can be transferred to another network.

The realisation of a system that can facilitate such a transfer is long overdue.

Such a system if operational, needless to say, will offer several advantages some of which are listed below:

(1) That each user maintains an account in each network will no longer be necessary because of mail routing capability of the system. This will, in effect, eliminate the undesirable redundancy of keeping multiple user accounts in each network for the sake of mail.

(2) Users will be saved the trouble of moving from one node to another logging in the networks merely for mail.

(3) Users logged in the network PC nodes can send mail to users working in the VAXcluster and vice-versa. This will be possible on line or otherwise.

(4) It will be a relief to the operators who currently have to use floppy disks in transferring mail from one network to another.

(5) It will be a portrayal of the ICS personnel readiness to acquire and use new technologies.

1.1 PROJECT REVIEW

The original idea of such a worthy system was conceived and perceived by Muiruri P.K as a project for the fulfilment of the postgraduate diploma in computer science award of the University of Nairobi. His work, titled "Message Routing system ", covered an overview of the three networks including modes of operation and mail transfers, analysis and design of the system and a prototype implementation of FIDOnet-PACSAT mailing system. In the context of further development of this routing system, his work will form the first phase.

Since the aspects (network configuration, technologies, mailing systems and file formats) of the three networks relevant for this system are well presented and documented in the first phase of the project, doing the same here will be like duplicating work already done. However, for the sake of clarity, understandability, continuity and independence some aspects will be revisited. These include those pertaining to the FidoNet and VAXCluster networks.

1.2 PROJECT OBJECTIVE AND SCOPE

The original objective of this project as presented in the proposal was to study and implement the system as outlined, analysed and designed in the first phase. Unfortunately, this was not possible and had to be changed for two reasons. First, the PC that is running the PACSAT software suite for the ICS station had its "card slots" full. Therefore it was not possible to hook the PACSAT network into the system and hence run PATHWORKS. Second, the VAX analysis and design issues are not as elaborate as to guarantee straight implementation. These two reasons leave no option but to outline, analyse, design and implement a system for transferring mail between the VAXcluster and the FidoNet networks.

The project report is divided into chapters. The first chapter covers the introduction . It discusses the aim and scope of the project. Chapter 2 reviews two networks with emphasis on the hardware components, configuration, mode of operation, mail address system and system users. Systems analysis is presented in chapter 3. Chapter 4 covers systems design which mainly dwells on how mail can be transferred between the networks. Weight is given on the need to keep as intact as possible the individual networks mailing system. Systems implementation is covered in chapter 5. Contained in this chapter is the description and purpose of various programs used in the system. Use of batch command files and function keys as a way of integrating the system is also described here. Areas that need further development are highlighted in chapter 6 together with summary reports under conclusion. Lastly, included in this report are Appendices, the systems user manual and program listings.

2.1 FidoNet System

2.1.1 What is FidoNet ?

Before going into the nitty-gritty of the mailing concepts (or technology) of the FidoNet, it would be in order to briefly look into its origin. Way back in 1984 Tom Jennings, the sysop (systems operator) of a private bulletin board system in the United States of America felt it would be a good idea if users of his system could send messages not only to each other, but also to users of a friend's BBS. He then started writing programs towards this goal and after a short while the first FidoNet mailer and BBS, " Fido ", was born.

" Fido " would pack all the messages destined for other systems, call them and deliver the mail. There, another ' Fido ' would accept the mail packets, unpack them and pass the message on to the individual users of that system.

The idea received massive feedback, and more and more sysops wanted to take part in big mail exchange (Wide area network). In just three months about 50 other system joined in, and in the beginning of 1985 there already were 150 " FidoNet nodes ", FidoNet was born.

2.1.1.1 FidoNet Structure and Technology

2.1.1.1.1 Structure

In the beginning, it was easy to know who operated what system, and what telephone number to call to reach a particular node. As the network expanded, it was becoming harder and harder to keep up-to-date. The routing of messages was getting more and more complicated as well. As a remedial measure, numbering scheme was developed, and therefore today's FidoNet address consists of four parts :

Zone, Net, Node and (optionally) point.

This system can be viewed as a hierarchical structure with Zones forming the root and points ranked the lowest thus forming the leaves. Zones refer to continents, Regions represent groups of countries, Net designates countries. A Net is made up of one or more Nodes. A node is build up from points which are nothing but computers (IBM or IBM compatible PCs) which a part from being used (by individuals, enterprises etc) for normal operations are also used for mail services e.g sending or picking up messages.

There are six Zones namely Zone 1 through Zone 6 representing North America, Europe, Australia, Latin America, Africa and Asia respectively.

So a node address is given by ;

Zone:Net/Node.Point

Thus, Vienna (Net 310) which is found in Austria (Region

31) in Europe has the address 2:310/0.

To drive point home, an illustration is made in figure 2.0 using Kenyan situation.

2.1.1.1.2 <u>FidoNet Technology</u>

At the point level or more generally the node, FidoNet consists of a terminal (IBM or IBM compatible) PC connected to a modem. The points communicate by exchanging messages and data over a FidoNet subnet which is a normal switched telephone lines. i.e

TERMINAL<-->MODEM<-->PHONE CONNECTION<-->MODEM<-->TERMINAL

The subnet comprises nodes which store and forward messages along the line from source to destination points (nodes) just as in ordinary packet switching. In case of communication breakdown while data transfer is in progress, the transfer just resumes from where it was interrupted on a subsequent call.

FidoNet runs various communication softwares depending on computers and operating systems. The common ones include Confmail, Qmail, FrontDoor, TrapDoor etc. Most countries globally, including Kenya, use FrontDoor[1].

The data for all FidoNet systems is kept in a single database, the 'nodelist'. It lists all the details of every node, such as the BBS name, the sysop (system operator) name,

[1] see section 2.2.3.4

the telephone number, modem flags and more. It also lists the FidoNet address (the node-name) for each node. Every week, the nodelist is updated; closed systems are removed, new participants added, telephone numbers get updated. All this is done with more tools and utilities.

2.1.1.2 <u>Other FidoNet services</u>

At the moment FidoNet consists of over 10,000 nodes with uncountable number of users. There is private mail between users (Netmail), and also public conference (Echomail areas), some of which are distributed over the globe. There are conferences about cooking, politics, sports, computers, telecommunications, Programming and much more.

Programs and other files are also distributed via FidoNet, especially if they are public Domain, Freeware or Shareware. There are excellent distribution systems, where a programmer of a utility just has to pack it into a compressed archive (together with the documentation), send it to the next co-ordinator, and the file will be moved around the world within a few days.

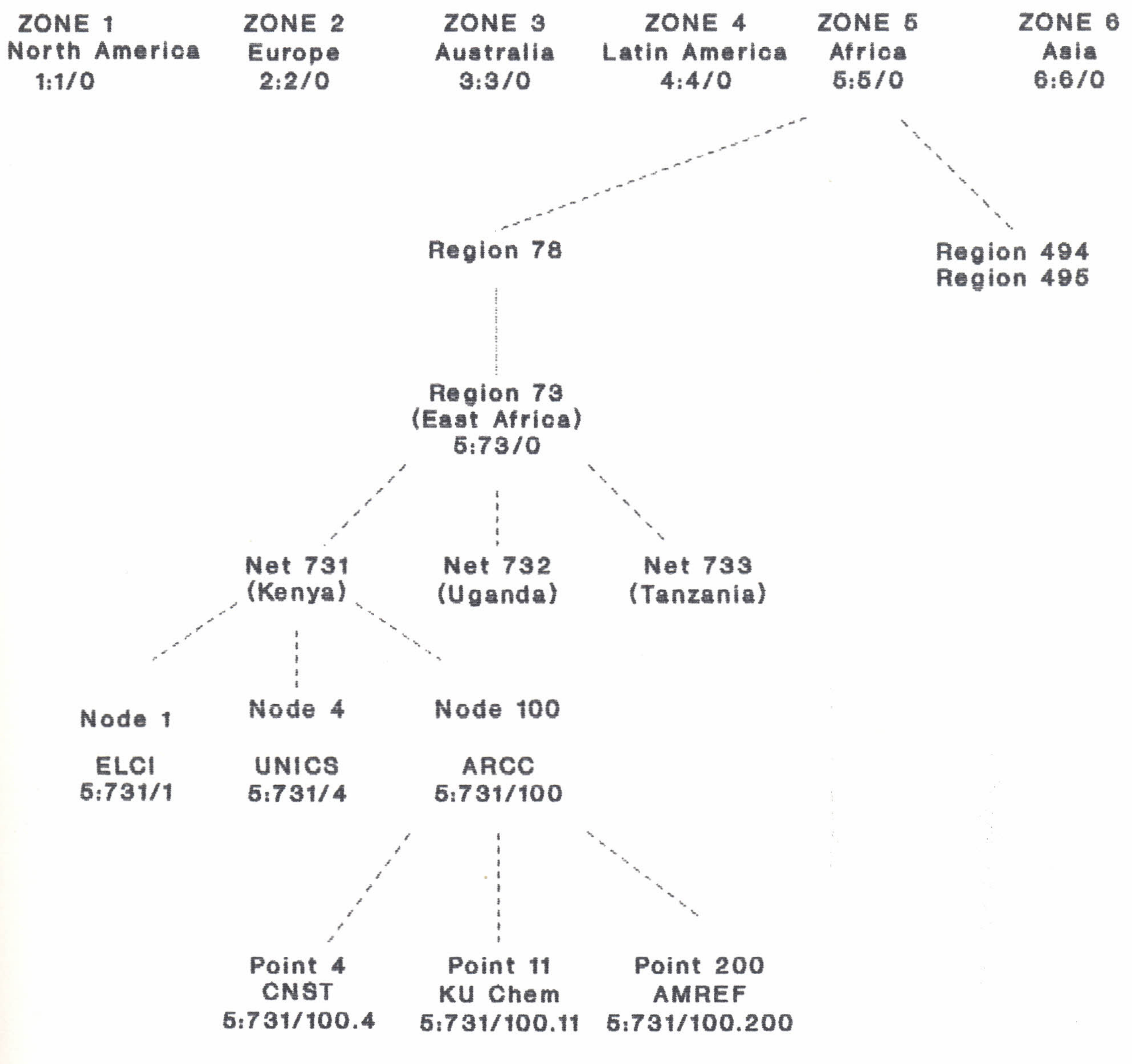

ZONE 1
North America
1:1/0
ZONE 2
Europe
2:2/0
ZONE 3
Australia
3:3/0
ZONE 4
Latin America
4:4/0
ZONE 5
Africa
5:5/0
ZONE 6
Asia
6:6/0
Region 78
Region 494
Region 495
Region 73
(East Africa)
5:73/0
Net 731
(Kenya)
Net 732
(Uganda)
Net 733
(Tanzania)
Node 1
ELCI
5:731/1
Node 4
UNICS
5:731/4
Node 100
ARCC
5:731/100
Point 4
CNST
5:731/100.4
Point 11
KU Chem
5:731/100.11
Point 200
AMREF
5:731/100.200

Figure 2.0

2.1.1.3 FrontDoor

FrontDoor is the complete e-mail package from Joaquim H. Homrighanson. It will run under MS or PC DOS versions 3.00 and above. It runs under most LAN software; including Novell and Lantastic. It also runs under PC-MOS/386, VM/386 and other multitasking environments such as DESQView.

FrontDoor incorporates ability to transfer data from one machine to another and guarantees error-free delivery by supporting the FidoNet protocols. It comprises the following components. **FD** : FrontDoor's Mailer or message module. It is responsible for sending, receiving and forwarding mail and files to other locations. **FM** : Is the FrontDoor's editor or message handler. **Set up:** which is FrontDoor's configuration utility; **FDNC:** the FrontDoor's compiler; **Terminal** : the FrontDoor's terminal Emulator; **FDCD** : the FrontDoor communication Driver.

2.1.1.4 FidoNet Mail Format

Although not very well documented, up to five parts of a FidoNet message can be distinguished. These are:

-Fixed header.

-Routing kludges.

-Mail text.

-in cases of a reply, a copy of the original message.

-and finally, route history kludges.

Fixed Header

The information content in this part is constant and comprises:

-Name of the mail sender.

-Destination name (or address).

-Subject of the message.

-Date and time of sending the message.

-Message status.

Routing Kludges

Mainly used by the mailer for message routing. A kludge is a set of characters preceded and superseded by ASCII characters ox1 and linefeed character ox10 respectively.

There are up to five kludges. The kludges are used to identify FidoNet mail, clarify a point system, originating node, address in which to reply to the message, e.t.c.

Message Text

The message comprises only a set of ASCII characters. If the message was a reply , the message being replied to is attached or appended at the end of the reply message.

Route History Kludges

These kludges give information on the FidoNet subnet nodes in which the mail travelled. They also contain information on the software running on the nodes.

2.1.1.5 FidoNet Mail Processing

The stages in which FidoNet mail is processed in presented in figure 2.1. The figure is explained as follows:

At the very bottom, the user sits in front of the message editor and read and write messages. On the left side, messages are created, entered in the message base, exported to the outbound directly, and sent to user's boss. On the right side, mail from user's boss is received into the inbound directly, then imported by the mail tosser, and finally read by the user.

2.2 The Vax System

2.2.1 What is VAX ?

VAX is a trademark of the Digital Equipment Corporation. It is a general-purpose computer system designed for growth and configured for many different applications. VAX computer come in many series including Vax 11, Vax 610, Vax 6300 series etc. The Vax computers support many users in a time sharing environment.

2.2.2 ICS VAXCluster System

The ICS VAXCluster system is a multipurpose system that provides the computer power, data availability and storage capabilities generally associated with mainframe system. The ICS VAXCluster system is configured using a method of interconnecting known as clustering. A **cluster** is a group

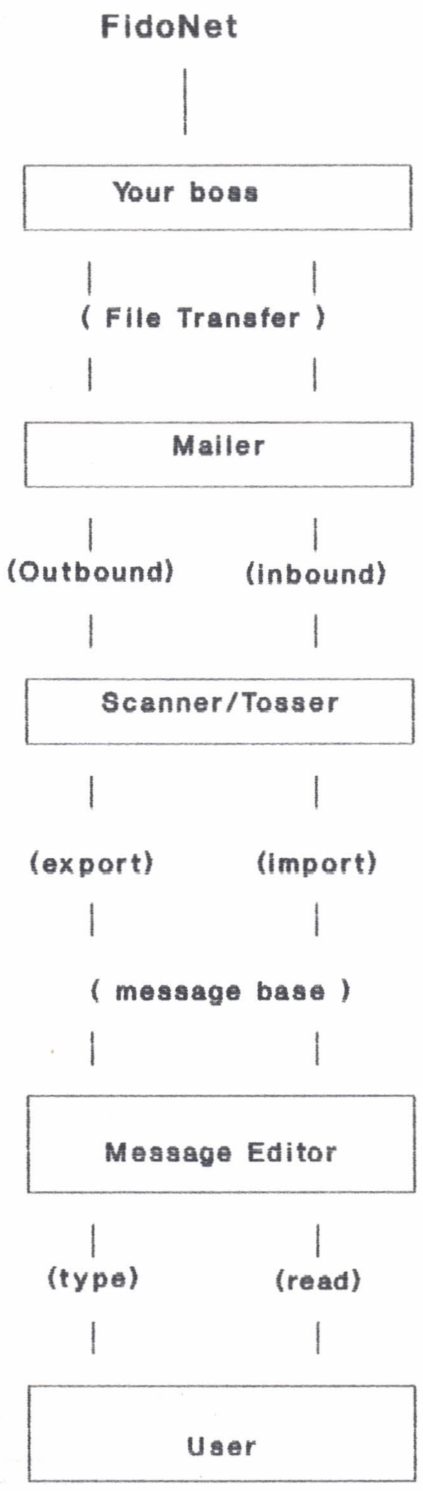


Figure 2.1: Mail transfer process

of two or more computers that share some or all of their resources. This provides increased capabilities for sharing data, distributing workload and providing higher system and data availability to University of Nairobi campuses and Kenyatta University.

The cluster consists of two VAX 6310 computers and VAXstation 3100 which although not part of the cluster is used to help manage the cluster. The names of the two computers, or nodes as they are normally referred to, which make up the cluster are AKILI and KAMILI. The name of the VAXstation 3100 is CHIEF.

CHIEF runs the VAXCluster console system (VCS) software package that provides a way to monitor and manage the cluster.

The VAXCluster has both Local Area Network (LAN) and Wide Area Network (WAN) configurations. Vax terminals and printers within Chiromo campus are connected to Ethernet LAN via DECsever. DECserver is Ethernet-based server that provides an interface between asynchronous serial data communication channels and Ethernet LAN. Other campus sites and Kenyatta University are connected in a WAN through leased lines via modem, MUXserver, and DELNI which finally connect to Ethernet LAN. MUXserver and DELNI are remote and local terminal servers for use on Ethernet LAN.

2.2.3 <u>VAXcluster Users</u>

VAX users comprise students, University academic and non-academic staff. Various system and application programs are run on VAX depending on users. These include students admission system, individual user programs, University account and payroll systems etc. Every user maintains a user account, which is assigned by the systems administrator, in the system disc. So to log in the system you must supply a username and password. This serves to inhibit unauthorised system or account access. Users are assigned different privileges depending on their needs.

2.2.4 Mail Facility

Other than running application and system programs, VAX users can send and receive messages from other users having accounts in the system. This is facilitated by mail utility called MAIL. To invoke the MAIL, type MAIL <enter> (type MAIL and press enter key) at the main prompt. The mail prompt (MAIL >) then appears. At the mail prompt users can send, read, extract or even store the mails as text files in their directories.

Since mail is sent and received by system users, all that happens is that the mail is transferred to the destination user directory in the mail folder after the sender presses ctrl/z (holding down ctrl key and pressing z key).

Messages received are stored in mail file, which have default file type of MAI. In this file, by default, MAIL provides three **folders** that store old, new and currently deleted messages. New messages are automatically placed in a folder called NEWMAIL; old messages are placed in a folder called MAIL. After you read a new message, the message is automatically moved from the NEWMAIL folder to the MAIL folder. Deleted messages collect in the WASTEBASKET folder until it is emptied. To empty the WASTEBASKET folder, enter either the EXIT or PURGE commands.

C H A P T E R 2: SYSTEMS ANALYSIS

Three issues are discussed in this chapter. The header information for each network mail, the mode of transferring the mail and the processes undergone by mail on transit (i.e what is referred to as external mail).

3.0 ANALYSIS OF THE SYSTEM

Although analysis of FidoNet-VAXCluster system was done in the previous phase of the project with a great level of satisfaction, the same will be done here to try and unearth cases of commissions or omissions resulting from sheer oversight or for any other reasons. Note that the terms mails and files will be used interchangeably.

As already stated in the project overview above, the current study will concentrate on mail transfers between the VAXCluster and the FidoNet networks for the reasons already given. Thus, the analysis will focus on mail transformation from the VAXCluster to the FidoNet and vice-versa. This will include:

(i) The mechanism for transferring mail between the networks (inter-network file transfer) .

(ii) Routing information headers appropriate to each network for transfers to the external domains in cases where this applies.

(iii) How en route mail is treated by each network with respect to changes in address and format.

3.1 Inter-network file transfers

Basically each network operates in its own unique way with a suite of program utilities including mail editor, the mailer, operating system, communication software etc. Each network therefore can be viewed as a "closed system". Despite the status quo, a way has to be found of moving files between the networks.

Three currently available methods of transferring files at ICS are discussed in the first phase, namely :

.Use of floppies

.Using KERMIT Utility

.Using PATHWORKS software

A merit-demerit analysis ensured thereafter which saw PATHWORKS, despite its software overheads, convincingly selected as the most optimal option under the current circumstances. This second phase now discusses PATHWORKS into greater details.

3.1.1 PATHWORKS

3.1.1.1 What is PATHWORKS ?

PATHWORKS is the framework that integrates different types of computers into a complete working environment. Although these computers use different operating systems, including DOS, OS/2, Macintosh, ULTRIX and VMS, users continue working as they do within their own environment.

With PATHWORKS, users can share data and files, take advantage of different printers, exchange mail and use network services. They can also run popular PC applications and send files to each other without thinking about the operating system.

3.1.1.2 How PATHWORKS operates

PATHWORKS is based on the server/client relationship model. Some computers on the network operate as **servers.** Servers make resources such as printers, applications such as Microsoft Word, and information such as data files available as services. When a PC is connected to a server, it is called a **client.** A client requests services.

Users can use resources directly connected to their PC (local) or the resources available as servers from a server (remote). The client request a service and the server responds to, or serves, the request. The client then uses the service transparently.

In this project the VAX computers running VMS operating system will act as the server while the FidoNet PC node, running DOS, will be the client.

3.1.1.3 Communication Transport Software

PATHWORKS requires a common transport on the client and the server. Therefore the same transport software must be installed on both the client and the server. For this project the client (FidoNet) will be connected to Ethernet-based local area network running DECnet communication transport software which will therefore provide the mechanism that allows computers to interact transparently.

Once PATHWORKS for DOS software has been installed on the FidoNet node PC (PATHWORKS for VMS has already been installed in the VAX computers), the following services will be offered to the PC:

.File services

.Disk services

.Print services

.Time and Date services

.Server manager and control

.Remote Boot services

.Broadcast utility

.Mail services

Of the services listed above, only file and mail services fall within the scope of this project and are consequently discussed in subsequent sub-sections.

3.1.1.4 <u>File Services</u>

File services allow DOS clients access to files stored on remote server. To the DOS user, file services look like local drives. Depending on the type of file service, users can control which of their files are accessible to others.

Similarly, DOS files become accessible to other types of users. For example, by storing files on a file server, VMS users can share files with DOS users.

The systems administrator configures each DOS client to meet that individual users needs. For example, one user may need access to Dbase IV and Wordstar file services while the other requires Rdb, Lotus-1-2-3 and WordPerfect. When these users display services connected to their PCs the service names are different because the users are configured only for the services they need.

3.1.1.5 Mail Services

PATHWORKS for DOS supports MAIL, a DOS-based front-end to VAX Mail, for sending and receiving messages.

DOS clients can send and receive mail from VMS server or even send data, text and binary files.

The Personal Computing Systems Architecture (PCSA) mail utility provides the tools needed to send and receive messages, and to store and perform operations on the messages in user mail folders.

There are two pieces of software needed to run PCSA mail.

.The server software, which runs on the VMS

.The client software, which runs on the PC

All users must have user accounts on the VAX. Before use of Mail can be allowed, a user must run MAILSETU program (a process called configuring mail) where nodename and username will be specified. If the run is successful a mail configuration file called MAIL.INI, is created. With this file, every time you start Mail, a mail screen called **Browser** appears. Users may choose menu options (**R**ead, **S**end, **F**older, **M**essage, **F**ilter, **G**roup or **O**ther) by use of a Mouse, Tab and the hot key, or Tab and arrow keys.

3.1.1.5.1 Configuring Mail

At the operating system prompt (DOS prompt), enter:

```
MAILSETU
```

A message will be displayed on the screen to the effect that

MAIL.INI is not found and that MAILSETU will create a new one. It further gives a prompt where to specify the name of directory where MAIL.INI should be written. If this directory is different from C:DECNET , you should add an environment variable to set MAIL to the name of the directory where MAIL.INI is kept. Press enter.

You are then prompted to a screen where you enter the following set-up options.

*User options

User name

Node name

*Keyboard and Mouse

Command mode = on/off

Mouse = on/off

*Callable Test editor options

Callable editor = on/off

*Screen options

Character set = none

A message then appears that tells you whether MAIL set-up procedure has been implemented or not.

3.1.1.6 <u>Interfacing Card</u>

In order to connect the FidoNet PC node to the Ethernet network (the VAXCluster local network) an interfacing card must be installed in the card slots at the back of the PC. From this card, a cable (BRAND REX LTD DIGITAL 1701320 20 AWG TYPE CL2) is used to connect to the Ethernet LAN. The card used is the DEC EtherWORKS Turbo.

3.2 Mail Formats

3.2.1 VAX mail format

The interactive VAX Mail utility (MAIL) allows users to send and receive messages, as well as to file, forward, delete, and reply to messages they receive. The messages received are stored in files called mail files. The default mail file, called MAIL.MAI, is created in the user directory the first time they receive a mail message. Users can create other mail files to accompany MAIL.MAI by using the COPY, FILE, or MOVE commands.

To effectively send fidoNet messages the following header information is required:

To: <nodename::username> ; The node and the user account of the message destination must be supplied. In cases where the sender and receiver are on the same node, only username need be supplied.

Subj: <subject of the message> ; The subject of the message is optional but it is a good practice to supply it for better communication.

The message text follows after this. It is terminated by **ctrl/z** for sending or **ctrl/c** to quit.

3.2.1.1 Sending and Receiving Mail

Sending Mail

To send a mail to Emisiko, username is PGD92001, on the VAX the following entries will be made:

```
To: UMOJA::PGD92001
Subj: Wallowing on 24-9-1993.
Hi Emisiko,
          Man, I'm glad we're approaching the end <RET>
of this course. You admit it wasn't a child play.<RET>
          Lets do a post-mortem of it on 24th, Sept.<RET>
          Please, confirm.
                                   So long,
                                      Odemba.
```

Press CTRL/Z to send or CTRL/C to cancel. You can also send mails to more than one user.

Receiving Mail

When you log in your account, the MAIL notifies you in case you have a mail. The message "You have n new mail messages" appears (where n is the number of mails i.e 1,2,3..). To read the mail, press return key at the MAIL prompt (MAIL>). The message number, date, time, the source address and the message subject will be displayed above the message text.

3.2.2 FidoNet mail Format

The FidoNet message header information to be supplied with the message text include:

From: <Name of message sender> ; This is the name of the person from which the message is originating. The default name is that of the "Superuser". This can changed by pressing **"ALT G"** and selecting other user names from the system account.

TO:<Destination address> ; The address of the node to which the message is destined. The format is ZONE:NET/NODE.[POINT] .

Subj: <Message subject> ;Refers to the topic of the message.

St: <Message Status> ; Depending on the urgency of the mail, it can be assigned the following status: CRASH, DIRECT, HOLD, IMMEDIATE e.t.c.

```
{ ------------------------------------------------------
           message text is entered
  ------------------------------------------------------
           in this area
  ------------------------------------------------------}
```

The **host address**, appears at the top left corner of the computer screen and need not be supplied. On the other hand the destination address can either be supplied interactively through the FRONTDOOR editor (FM) together with the message text or by the program AUTOMAIL.

3.2.3 En route Mail

Generally mail will be processed in stages more or less in the store-and-forward fashion. This is necessitated by the fact that mail format (the header information) is unique for each network and the appropriate changes must be made as the mail moves from one network environment to the other.

The mail will be categorised into two. The internal and the external mails. The idea here is to make the system more versatile so that a VAX user can send and receive mails locally from a user with an account on PC node at no cost other than through the gateway at some cost and delay. Internal mails are those originating from the PC running the FidoNet node and destined for VAX users and vice-versa. External mails are those from the VAX system or the PC node but are destined for external FidoNet domain and vice-versa.

3.2.3.1 Internal Mail

VAX system users may communicate locally with other users logged on the PC node running the FidoNet node. This will be made possible by the use of the PATHWORKS MAIL (mail utility). The only requirement is that their accounts (usernames) must be configured using the program MAILSETU. See section 2.4.1.1.6 above.

3.2.3.2 External Mail

External mail will be comparatively difficult to process. Mail from the external FidoNet domains and destined for VAX system and those sent from the VAX system to external domains will undergo a transformation. The appropriate routing information will have to be attached. The way to go round this is to have a point within the system network (which is accessible to both networks) where the header information can be changed accordingly. VAX server file services can be deployed with the PC node being the client. Program routines can then be developed that can carry forward the mails to the intended destinations.

A formal representation of the system is presented in figure 3.0.

3.3 Data Dictionary

In order to make the processes and entities in the Data Flow Diagram (figure 3.0) more understandable, a brief description is given, for each, in table 2.1.

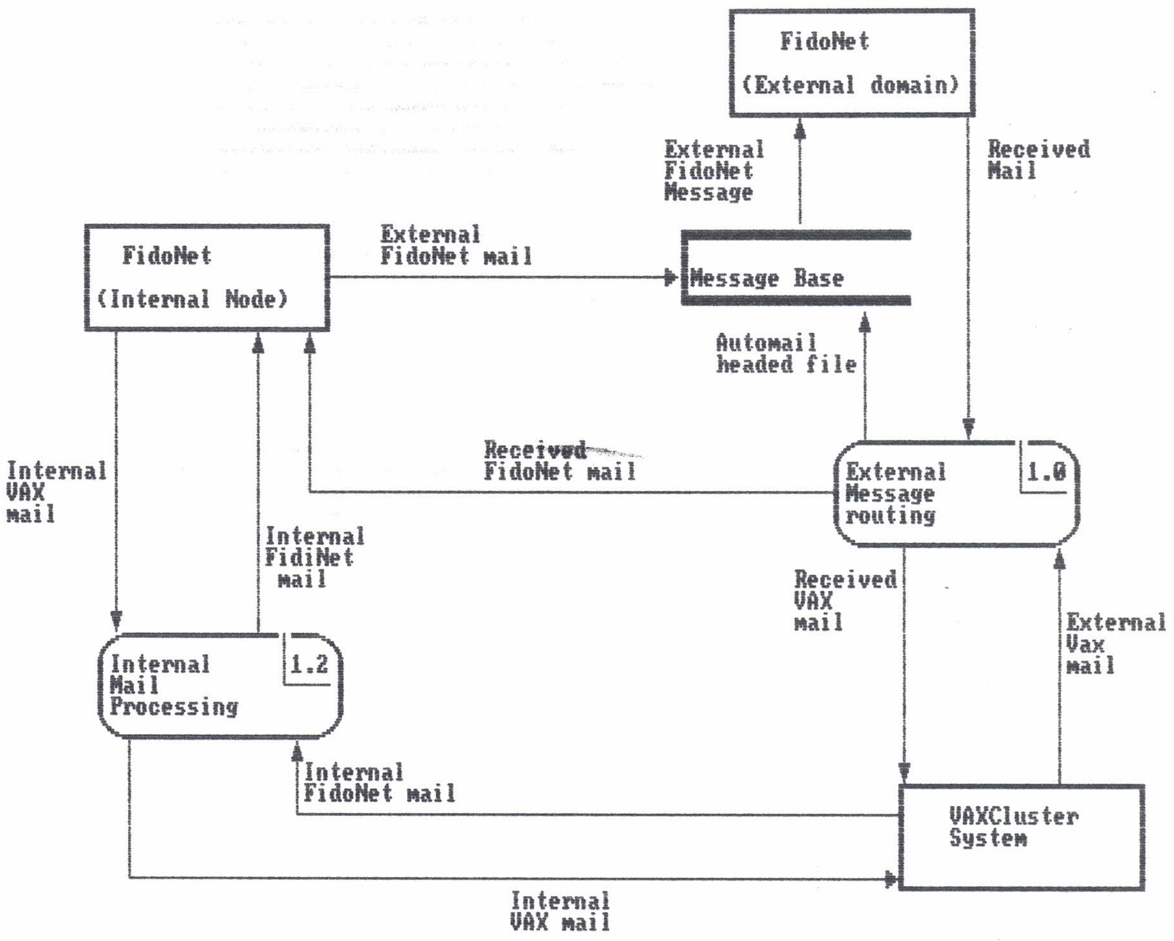


Fig. 3.0 : System Data Flow Diagram.

Table 2.1: Description of System entities and Processes

2.1 (a)

Entity	Description	Aliases
FidoNet nternal node	The PC at ICS running FidoNet node with address 5:731/4	FidoNet node
FidoNet External domain	Other FidoNet nodes all the world where FidoNet mails can be sent to	Remote nodes
VAXCluster System	The ICS VAX System	VAX Network

2.1 (b)

Process	Description	Aliases
Internal mail Processing	The system transferring mails from the VAX to PC node and vice-versa	Internal mailer
External message routing	The system that sorts the VAX and FidoNet mails received from the remote nodes and also assembles,at the messsage base, mails to be sent to the remote nodes.	External mailer

Table 2.1 (c)

Data structure/Element	Description	Aliases	Data Type
External FidoNet mail	Mail originating from local node and destined for remote nodes (header information + message text)	sent message/mail	ASCII text
Internal FidoNet mail	Mail from the VAX to PC node	-	ASCII text
Received FidoNet mail	FidoNet mails from the remote mails	new messages	ASCII text
Internal VAX mail	mail from the FidoNet node to VAX system	-	ASCII text
External VAX mail	Message from VAX System to remote nodes (header information + message text)		
Received VAX mail	VAX mail from remote nodes	-	ASCII text
Header information	The neccessary codes attached to a message text for routing purposes	routing information	ASCII text codes

C H A P T E R 4: SYSTEMS DESIGN

The system design is presented as a sequence of steps that mail undergo between networks. Emphasis is made on the desire to retain each networks mailing system as much as possible.

C H A P T E R 4: SYSTEMS DESIGN

The system design is presented as a sequence of steps that mail undergo between networks. Emphasis is made on the desire to retain each networks mailing system as much as possible.

4.0 SYSTEMS DESIGN

The idea in this systems design is to maximise on the interfacing capabilities of the two networks while minimising making changes on their current modes of operations (sending and receiving mail) such that very little or no alterations are made in the way each system handles its mail.

As a first step, the PC running FidoNet suite is to be hooked to the Ethernet-based LAN of the VAXCluster system via a cable (BRAND REX LTD DIGITAL 1701320 20 AWG TYPE CL2) which connects to the DEC EtherWORKS Turbo card at the back of the PC. The Ethernet LAN is running DECnet communication transport software. This will facilitate the operations of PATHWORKS software as the media for integrating the DOS and VMS operating system running on the PC and the VAX system respectively. Figure 4.0 shows the system configuration. The current study focuses on the area enclosed with the dotted lines.

Generally, mail will undergo different processes depending on the source and the destination. The systems design is therefore divided into subsections where each describes a component of the system mail transfer.

The design tool chosen is **structured charts.** This has solely been determined by the nature of the project.

4.1 Internal Mail

The PCSA mail will be used to send, receive, store and to perform operations on the messages in the mail folders.

As a prerequisite two pieces of software needed to run the PCSA mail must be installed.

. The server software, which runs on VAX server.

. The client software, which runs on the PC nodes.

The intended users must be assigned accounts on the VAX system. The accounts (containing node and user names) are specified when the mail configuring file, MAILSETU, is run to create a MAIL.INI file.

For mail transfer from VAXCluster to the PC node, specify the node name, username and the message subject. Thus:

```
TO: <NODENAME::USERNAME>
Subj:<subject of the mail> .
```

For mail transfer from the FidoNet PC node to the VAX the Browser menu is used i.e by choosing the appropriate menu options **SEND, READ, FILTER, FORWARD** etc followed by the prompt requests.

4.2 External Mail

Unlike the internal mail, the external mail traverses the networks boundaries. As such the necessary codes that specify the source and the destination must be attached at the appropriate fields as part of the header information.

The fields to contain this information will be determined by the source network and the destination network.

4.2.1 Mail transfer from VAX to FidoNet

The VAX mailing system uses the **VAX MAIL** which is a program file in binary form and which transfers mail between VAX users mail folders without an intermediary state. Due to lack of such an intermediate stage and the inability (or the overheads involved) to modify the VAX MAIL program to incorporate the FidoNet network, an alternative approach is required.

For a successful mail transfer from the VAXCluster to FidoNet external domains, the following steps are considered feasible;

-The FidoNet mail will be prepared in the VAXCluster environment. The program to process the mail will put in a directory that is accessible to all VAX system users. The program will be called **FDMAIL** and it will be put in the directory DISK$USER1:[PCCOMMON]. In this way each and every user can invoke it at any time he/she wants to send a FidoNet mail.

-Text files containing the messages/mails will be created each time the FDMAIL program is run. These text files will be stored in a common directory which shall be called DISK$USER1:[PCCOMMON.FMAIL]. The file names will be provided by the FDMAIL program at run time. The system should perform a check to establish that no new file names match the existing ones in the directory to avoid cases where the old files may be overwritten by new ones.

-The files have then to be moved to the FidoNet message base where they will wait for an event e.g to be sent. The transfer will be done with the aid of the program AUTOMAIL. The program AUTOMAIL can only run in the FidoNet Environment because it requires a set of look up tables. The whole process is to be defined as of one FidoNet events to be run periodically so that there is no human intervention.

So for a VAX user to send external mail, he/she has to invoke the program FDMAIL and supply the following entities:

FROM: <sender name>

TO:<destination address>

SUBJ:<subject of the message>

ATTR:<status of the message>

FROM : Is the name of the person sending the message. This has to be supplied because unlike in the FidoNet where the superuser becomes the default sender if the name is omitted, the entry will be left blank which may cause automail spawned process to abort. It is 36 characters.

TO : Is a valid FidoNet address of the receiving node. The sender is ask to ensure that the right address is supplied because the system does not do 'address checking' at the moment. It is 36 characters.

SUBJ: Is the subject of the message. 72 characters is given for this field.

ATTR(status): Is used to communicate to the FidoNet mailer about the urgency of the mail being sent i.e IMMEDIATE, CRASH e.t.c.

The field widths given above are not arbitrarily assigned but and made to coincide with those given in the FidoNet environment.

The steps described above can be represented in structured English thus:

```
START
  START FDMAIL PROGRAM;
  SUPPLY HEADER INFORMATION;
  TYPE MESSAGE TEXT;
  STORE MESSAGE FILE IN [.FMAIL] DIRECTORY;
  *transfer mail to FidoNet base*
   WHILE NOT END OF DIRECTORY FILES
     AUTOMAIL FILES;
   *endwhile
STOP.
```

4.2.2 Mail transfer from FidoNet to VAXCluster

The mail from the external FidoNet domain, generally referred to as FidoNet mail, is stored in a specific directory that is specified by the sysop (system operator). Currently the default directory is C:\FD\AUGMAIL.

The system will scan this directory for mail meant for VAX users and transfer them to PCSA file service from where a program from the VAXCluster environment will be run periodically by a system operator to distribute them to individual VAX users accounts. In this way, the VAX users will be able to receive mail from the external FidoNet domains the same way they do receive mails from other VAX users. Alternatively, a user if so wishes may peruse through the mail directory for his/her mail.

Just like in sending mail from the FidoNet to **Internet** networks, the sender must include a code (VAX username) as part of the destination address on sending a mail to a VAX user. (The code is not to be used by gateways for routing purposes but for use in sorting mails locally once they are received at the PC node) . A user file is to be maintained at the local FidoNet node that contains VAX users' usernames.

Thus to address a message from a remote FidoNet node to a VAX user, the following information should be entered:

FROM:<sender name> ; name of sender.

TO:<VAX username> <FidoNet destination address>

Subj:<MESSAGE SUBJECT> ; title of the message.

ATTR:<Status of the message>

Emphasis should be made on the fact that, the VAX mail is sorted out by searching **VAX username** in the username file which is maintained in the system. It is therefore mandatory that this attribute is not omitted or misspelt by the sender.

The design process can be presented in structured English as follows:

```
          START

            CHANGE  DIRECTORY  TO AUGMAIL;     * dir  where  mail
                                                 is received*
            WHILE MORE FILES IN DIRECTORY

              MOVE VAX-MAIL TO DISK$USER1:[PCCOMMON.VMAIL];

            *end while;

          STOP.
```

The VAX mail in the directory DISK$USER1:[PCCOMMON.VMAIL] can be periodically transferred to the respective NEWMAIL folders in the corresponding user accounts by running the transferring program, say, by an operator. The program has to incorporate the VMS command

```
> MAIL/SUBJECT = "string" <filename> USERNAME
```

in a batch command file. Where:

SUBJECT:<subject of the message>

filename:<the name of the message file>, which should have extension .TXT. The extension is not optional.

USERNAME:< valid VAX username> , a user account name in the VAX where the mail is to be sent.

For mail transferred from FidoNet to VAX the header information such as sender name, address and message subject should not be removed. In this way the recipient will be able to know the message source and therefore send back the reply. Further more no more header information will be added because the message will not be re-routed for another destination but will be consumed by VAX users.

The author draws the attention of the reader to the fact that the process of moving VAX mail files from the FidoNet default directory to the system server directory [PCCOMMON.VMAIL] is to be defined as a FidoNet event that is to be run at specific times.

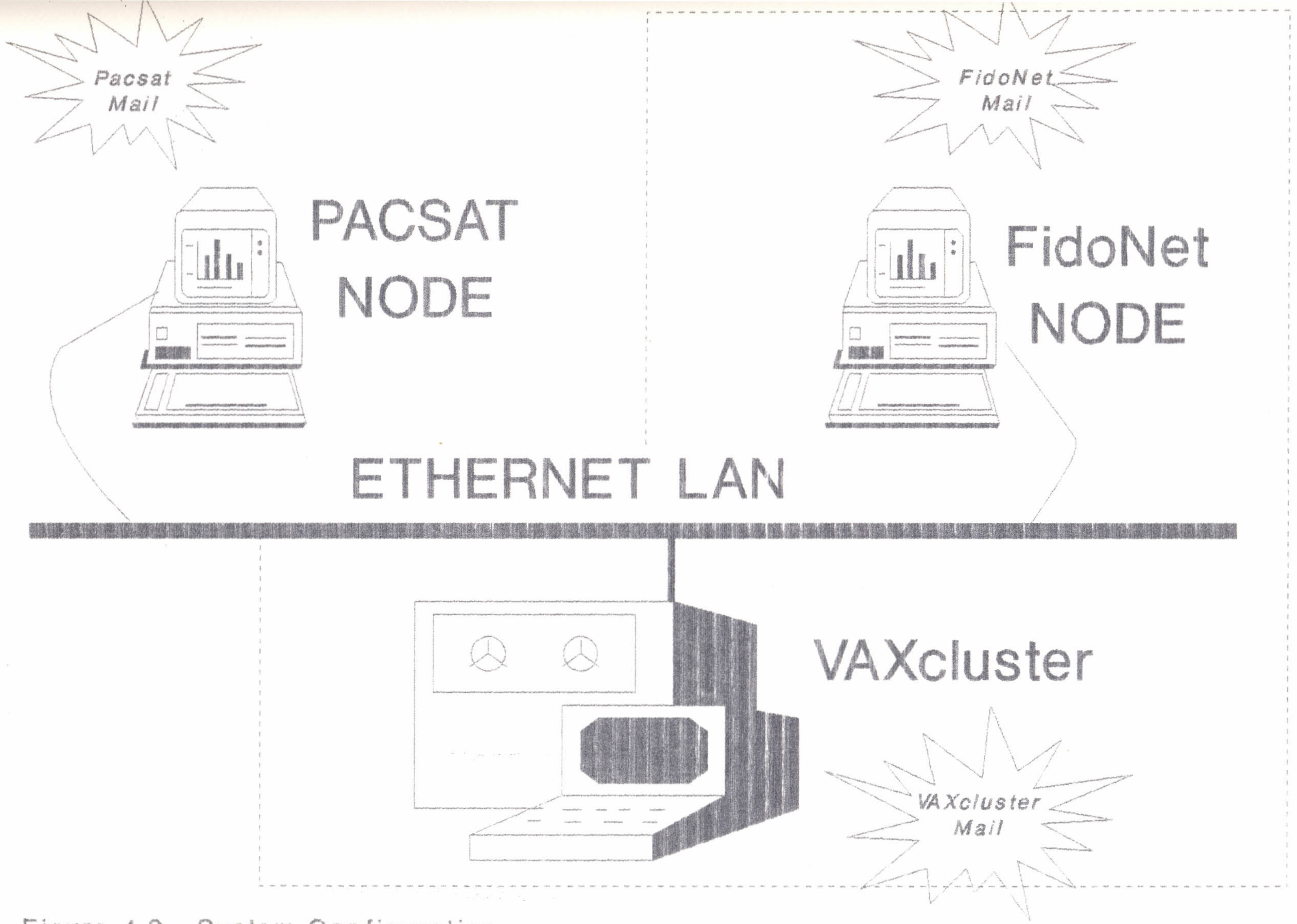


Figure 4.0 : System Configuration

--- Shows area of current study

CHAPTER 5 : SYSTEM IMPLEMENTATION

This chapter puts together the various components of the design into functional units. It discusses the materials used and the programs developed in relation to their functions.

5.0 SYSTEM IMPLEMENTATION

5.1 Materials Used

5.1.1 Hardware:

The hardware component of the system comprises the VAXCluster system, and the KSI PC (an IBM compatible PC) running local FidoNet node. The PC is connected to the VAXCluster Ethernet based LAN via BRAND REX LTD DIGITAL 1701320 20 AWG TYPE CL2 cable. The cable connects to the DEC EtherWORKS turbo card at the back of the of the PC.

5.1.2 software

Sets of programs were used to perform specific system functions. The main software that forms the backbone of the project is PATHWORKS. It was used to facilitate "file sharing" between DOS based KSI PC running FidoNet node and the VAXCluster system. See section 3.1.1 for full discussion of PATHWORKS.

Other software include various programs which were written for transferring mail between networks. These are discussed under the following sub-headings:

5.1.3 Programs

There are various program routines performing various functions of the system.

5.1.3.1 Mail transfer from VAX to FidoNet

FDMAIL:

The program is used by VAX users for preparing external FidoNet mail. It accepts the message header together with the text and puts the information in a file which it assigns a unique name. The file(s) is then put in the server volume directory DISK$USER1:[PCCOMMON.FMAIL] from where it will be transferred to the FidoNet message base indirectly by the program TOFBASE.

The program is activated when the sender types SMAIL (read send mail) at the VAX prompt i.e $ SMAIL. See user manual in appendix A for more information.

TOFBASE:

This program scans the server volume directory DISK$USER1:[PCCOMMON.FMAIL] prepares each file and calls the FidoNet program AUTOMAIL which transfers these files to the FidoNet message base.

The program is invoked by typing FIDOMAIL (which is a batch command file). The file therefore has the following characteristics:

usage: TOFBASE <directory path>

input: files in the directory { [.FMAIL] for this case }

output: files in the FidoNet message base

5.1.3.2 Mail transfer from FidoNet to VAX

SYSMAIN:

Is a program that maintains the VAX user accounts. The program SYSMAIN checks for valid VAX users in this file. To run the program one required to know the password. The allocation of usernames is left for the operator. However, for easy communication the individual names (surnames or any other names with not more than 12 characters) are used. This enables remote senders to address the mail in the normal way by typing the receiver´s name together with the address. The limitation on the 12-characters surnames is due to the fact that VAX allows a maximum of 12 characters for usernames. Thus a one to one correspondence in "usernames" is guaranteed.

The operator can add and delete user accounts.

FFTOVAX:

The program scans the received files in the default FidoNet directory,selects the VAX mail/files and changes their extensions from the default .MSG to .TXT. The .TXT files are later transferred to the sub-directory [.VMAIL] in the server file volume. The program expects user accounts maintenance file as one of its inputs. It is run as part of a batch file.

usage: FFTOVAX <directory specification>

input: files in the directory specified with extension .MSG

output: VAX files with extension .TXT

DMAIL:

The program distributes mail received from the remote FidoNet nodes to the respective VAX user accounts. This in effect allows the users to receive their FidoNet mail in the same way they do receive the VAX mail.

5.1.4 Other materials

Programming language

The programs listed in section 5.1.3 above were written in C programming language i.e TURBOC by Boland international in DOS and VAX C in the VMS environments respectively. The language was chosen for a number of reasons:

-Since it was the language used in the previous phase, its use here guarantees continuity.

-The language has great functionality that interfaces the features characteristic of a high level language and those of operating system ; a characteristic that is just so right for this system.

Wordprocessor:

The wordstar word processing, a MicroPro international corporation product, was chosen for report writing mainly because it was installed in all the PCs at the ICS and therefore more available compared to Wordperfect and Microsoftword.

5.2 System Integration

Naturally a system user need not know the underlying technicalities of a system, and this system is no exception. This calls for a way of shielding the system details from the user while at the same time giving him/her all the facilities that the system offers.

For this system batch command files are used to run various program sets. The user will then simply type commands synonymous to the program functions. Use of function keys was also considered.
(see chapter 6 under the section ´further suggestions´).

Some of the batch programs are discussed below:

VAXMAIL:

This is a batch command file that is used for transferring VAX mail from the FidoNet environment. It invokes the program FFTOVAX program. Further it deletes the transferred mail files from the FidoNet node default directory.

FIDOMAIL:

The command file invokes the program TOFBASE (see section 5.1.3.2). For files with various forms of errors e.g incorrect addressing, the program assigns them an extension .MIS . The .MIS files are then stored in the SUB-directory BAD. In this way the system keeps information about mail not transmitted.

RMAIL:

This batch program is used by VAX users to access the VAX mail messages. This they do incase of urgency where they cannot wait for the normal dispatch by the operator. After reading the message(s), control is taken back to the users own directory.

CHAPTER 6:CONCLUSION

The chapter gives a summary of the results, statement of conclusion and suggestions for further work as well as the constraints encountered in the cause of the project.

6.1 Summary of Results

To recapitulate, the objective of this project was to analyse, design and implement a functional system for transferring mail between the FidoNet and VAXCluster networks. However, the stated objective was not fully realised due to lack of time. Nevertheless, it is the authors humble opinion, that the bulk of the work was accomplished as is and that with the suggestions given below, a full blown system can be realised with minimal overheads.

The project, however, was not without something to cheer about. VAX users can now receive messages from remote FidoNet nodes. On the other hand, mail transfer from VAX to FidoNet was not 'polished' well enough due to lack of time.

6.2 Constraints

There were a number of setbacks experienced during the lifetime of the system development but only two will be highlighted in this section.

(i) There is lack of proper documents on FidoNet network. There are only user manuals and guides which are merely adequate for operators and users but not for someone wanting to study the underlying details of the system. The few materials from which references were made are given in appendix D.

(ii) Course regulations stipulate that the project be undertaken for a period of 12 weeks. This period is fairly short considering the amount of work that was to be covered e.g studying the FidoNet and VAX networks, learning details about DOS and VMS operating systems, Turbo C and VAX C programming languages e.t.c . Due to this time limitation not all the components of the project were implemented.

A combination of the two constraints discussed above created an experience the author can summarise as "spending unavailable time looking for unavailable documents!".

6.3 Suggestions for further improvement

For the system to operate fully, the following developments are recommended:

(i) The system operates in a store and forward kind of fashion to transfer mail; more or less like batch processing.

At the moment, an operator has to effect the transfers e.g to transfer mail from VAX to FidoNet, the operator has to run the batch program VAXMAIL. Also to transfer mail from the FidoNet to VAX, the batch program FIDOMAIL must be run. This kind of human intervention is uncalled for as it is undesirable. The processes can be defined in the FidoNet SETUP under manager program EVENT.SYS as external events. For each process, the following should be defined:

Tag:	The event ID. A letter A - Z or @. X is used for external events.
Days:	The days you want the event to be active. The @ is always active.

Modifier:	Used to define the start time of an event.
Length:	Defines the length of an event.
	e.t.c.

(ii) The author recommends the use of function key sequences as a way of invoking commands or programs. This will simplify the system and make it more user friendly.

(iii) Since VAX mail received from the remote domains are sorted by usernames (receiver names in some cases), the message senders should be notified of this fact so that a standard name is always used. For example, to the system, Lucas@Odemba, Odemba@Lucas, L@Odemba e.t.c are totally different users. (But the system is case insensitive to user names).

BIBLIOGRAPHY

The following documents and manuals were referenced/consulted at various stages of the project.

[1.] E-MAIL DEMYSTIFIED:LOST-COST NETWORKING WITH FRONTDOOR
-Ochuodho S.J ; A publication for ARCC.

[2.] FIDONET : description of the architecture and available services
-Dieni C - Lanari A.

[3.] FRONTDOOR User manual and administrator´s guide.

[4.] MESSAGE ROUTING SYSTEM -Report (ICS Library).

[5.] PATHWORKS for DOS documents.

[6.] TURBO C User´s and Reference guides.

[7.] Using Turboc C
-Harbert Schildt, BORLAND.OSBORNE/McGRAW.HILL

[8.] VAX C Guide and VAX RTL Reference manuals.

[9.] VMS Guide.

APPENDIX A: USER MANUAL FOR THE PROTOTYPE

The appendix provides a guide to the user on how to read and send mail.

A1: Internal Mail

This is mail that is sent and consumed locally. What this means is that a fellow working on the PC running FidoNet node can send and read mail from a VAX user and vice-versa.

A111: Sending mail from VAX to FidoNet

To send mail to the PC, the following information must be supplied :

From:<nodename::username (of sender)>

To:<nodename::username (of destination)>

Subj:<the message title>

note: in cases where the the users share the same node (e.g KAMILI or AKILI) the nodename becomes optional.

A112: Reading mail on VAX

When a user logs on to the VAX, he/she is notified by the system in case there is mail for him/her i.e the message

You have n new mail

appears on the screen.(n represents the number of mail). To read the messages follow the steps below:

(i) Type MAIL at the system prompt and press. The prompt MAIL> will appear.

(ii) At the MAIL prompt press return key again.

(iii) The first mail will be displayed on the screen. To read other mail press return key subsequently.

(iv) To exit from the mailer type exit.

A12: Sending and reading from the PC

In both cases, the user types ´MAIL´ at the drive C root prompt. A menu called Browser will pop up. The user can then select the appropriate options. E.g send, read, forward etc. For more information refer to PATHWORKS for DOS Mail reference manual.

A2: External Mail

A21: Sending mail from VAX to FidoNet

The steps are as follows:

(i) Type SMAIL at the system prompt. The system will prompt you for the following:

From: <type your local FidoNet address> e.g

From: Odemba L (5:731/4.1) ; note that the name Odemba L must be in the FidoNet nodelist as a user.

To: <type the destination address> e.g to send private mail to Keith Scroggin at Thieves Guild on address 1:109/534. Enter,

To:1:109/534 [the thieves guild] {[] shows that entry is optional}.

Subj:<the title of the message>

Attr:<message status> e.g

Attr:PVT {PVT for private mail}

The system then allows you to enter the message text.

You can type ctrl/z to send the message or ctrl/c to cancel sending.

A22: Reading mail from FidoNet on VAX

There are two ways of reading such a mail.

Method 1:

(i) Type @RMAIL at the system prompt. The system will display all the received mail and then give the prompt:

Enter Message File:

(ii) Enter the file name of the message you want read.

(iii) Press return key to return to the prompt.

Method 2:

In case the user is not in a hurry to get the latest of mail. He/She should wait until the mail becomes available in his/her folder. He/She can then follow the procedure in section A112 above.

How to operate the system:

(i) To add, delete or list usernames(s), type

```
C:\PROJECT>SYSMAIN
```

The system will then prompt for password. In case of incorrect password the system will allow up to three tries after which it terminates the process. If the correct password is supplied, the operator/user is prompted with a menu from where any of the options

add, delete, list, quit, or ? for help

may be chosen.

(ii) To sort VAX mail, type

```
C:\PROJECT>VAXMAIL
```

The operator will then be prompted to enter his/her password at some stage. He will also be notified when the transfer is complete.

(iii) To sort FidoNet mail, type

```
C:\PROJECT>FIDOMAIL
```

(iv) To dispatch mail, the system operator enters the command procedure name DMAIL (read distribute mail) at the VAX system prompt i.e

```
$ @DMAIL
```

A23 Sending mail from remote FidoNet node to VAX User

To successfully send the mail, the following entries are made using the FrontDoor message editor.

From:<local node address>

this is by default the superuser name and address.

For the ICS node this is Tony Rodrigues,ICS (5:731/4@FidoNet).

To:<destination address>

e.g to send mail to Dr Okelo-Odongo whose VAX username is BILL, the following entry is made:

```
To: bill@ICS@(5:731/4.1)
Subj:<headline of the message>
attr:<message status> e.g pvt for private message.
```

The message text can then be entered. Each line of the message text MUST be terminated by a CR/LF (carriage return/linefeed) combination for the message to be formatted in a manner that is readable to the VAX receiver in VMS environment.

APPENDIX C: PROGRAM LISTINGS

The source codes of the programs used in implementing some system modules are listed in this appendix. For all the programs, a consistency in the cause of termination or aborting was maintained by assigning peculiar error levels to specific causes throughout the system. This system of keeping track of causes of premature program termination was borrowed from the previous project phase.

The following table summarises the error levels and their possible causes.

Error Level	Possible cause
0	Program executes to completion
1	Opening or Creation of file failed
2	Subprocess initiation call fail
3	User access denied
4	changing current directory fail
5	Incorrect program usage
6	File(s) not found

The programs are listed in the next appendix pages.

```
  /*************************************************************

   *             Program:FDMAIL
   *   This program accepts the message text from all VAX
   *   users, creates unique file names each time and
   *   stores the files in the directory
   *   DISK$USER1:[PCCOMMON.FDMAIL]

                Written by:
                Odemba L.O.R
                P64/7297/92

   *************************************************************/

#include ctype
#include stdio
#include stdlib
#include rms
#include ssdef
#include unixio

#define MAXCHAR 3000
#define temp "fnnXXX.XXX"

int  rval,i;
char *dirc="DISK$USER1:[PCCOMMON.FMAIL]";

char ring[MAXCHAR];
char *addr=temp;
char ch;

FILE *fpt;

struct partc {
 char SNAME[36];
 char RNAME[36];
 char MSUBJ[72];
 char ATTR[24];
};

struct partc specf;

main()
{

 rval = chdir(dirc);
 if (rval == -1) {
  printf("%DCF- directory %s not found\n",dirc);
  exit(4);
 }

 /* creating a unique file name from the rest
   already in the file block */
```

```
addr = mktemp(temp);

if (*addr == NULL)  {
 printf("%UFNF- unique file name creation fail");
 exit(1);
}
else {
 i=0;
 printf("FROM:");
 gets(specf.SNAME);
 printf("TO:");
 gets(specf.RNAME);
 printf("Subj:");
 gets(specf.MSUBJ);
 printf("ATTR:");
 gets(specf.ATTR);

 printf("Enter Message here.Type ctrl/z to send ctrl/c to quit.\n");

 for(;;) {
  ch = getchar();
  ring[i] = ch;
  if (feof(stdin))
    break;
  i++;
 }

 fpt = fopen(addr, "w");
 if (!fpt) {
  printf("%EOIF- error opening file");
  exit(1);
 }

 else {

   /* write the header information into
      a file   */

  fprintf(fpt,"From %s\n",specf.SNAME);
  fprintf(fpt,"To %s\n",specf.RNAME);
  fprintf(fpt,"Subj %s\n",specf.MSUBJ);
  fprintf(fpt,"Attr %s\n",specf.ATTR);
  fprintf(fpt," %s\n","\\END");       /* marks the end of the header
                                          information */

  /* now enter the message text
     to the file */

  fprintf(fpt,ring);

  fclose(fpt);
 }
}
```

```
  printf(">>>>> message sent <<<<<");

}

/********* end of program ********/
```

```
/* ///////////////////////////////////////////////////////////

                 Program name: SYSMAIN.

      This program is used for maintaining VAX user
      account.

                 written by:
                 Odemba L.O.R
                 P64/7297/92.

   ///////////////////////////////////////////////////////////   */


#include <stdio.h>
#include <fcntl.h>
#include <stdlib.h>
#include <conio.h>

#define BUFF_SIZE 12
#define REC_SIZE 36
#define messg(messno, mtext) { puts( #mtext); exit(messno);}

typedef enum{TRUE,FALSE} boolean;

char ch,record[REC_SIZE],frecord[REC_SIZE];
char response[BUFF_SIZE],prompt[BUFF_SIZE];
char *addr = prompt;

boolean test;
int fhand, fstat,i,j,k;
FILE *stream, *sfl, *lfil, *temp;

main()
{

  password();

  for(;;)  {
    printf("\n\n\n");
    printf(" Enter option (A,D,L,Q OR ? for help):");
    gets(response);
    ch = tolower(response[0]) ;
    switch(ch)  {

      case 'a': add_record() ;
                break;
      case 'd': del_record();
                break;
      case 'q': quit_menu();
                break;
      case 'l': list_users();
                break;
      default : printf("ERROR-%IVOP- Unknown operation\n");

      case '?': case '\0':
                type_options();
    }
  }
}

/* quit function */

quit_menu()
{
  clrscr();
  exit(0);
}


/* this function adds new usernames to
   the file */

add_record()
{
  clrscr();
  i=0;
  printf(" Enter username or type QUIT to exit.\n\n");
  printf("record? :");
  gets(frecord);
  j=stricmp(frecord,"quit");
  if (j == NULL)
    return;
  else {
    stream = fopen("C:\\PROJECT\\FDIR\\USERF.DAT","a+t");
    if (!stream )
      messg(1, ERROR-%FOF- User file opening failed)
    else {
      fread(record,36,1,stream);
      for (;;) {
        fseek(stream,0L,SEEK_SET);
        while (!feof(stream))  {
          j = stricmp(frecord,record);
          if ( j == 0)  {
            printf("Username already exists\n");
            break;
          }
          else
            fread(record,36,1,stream);
        } /* end while */


        if (feof(stream)) {
          fwrite(frecord,36,1,stream);
          printf("username added to file\n");
        }
        printf("record ?: ");
        gets(frecord);
        j=stricmp(frecord,"quit");
        if (j == NULL) {
          clrscr();
          fclose(stream);
          return;
        }

      }     /* end for */

    }

  fclose(stream);

  }

}

  /* the function type_options gives the
     options in full */


type_options()
{
  clrscr();
  printf("\n\n\n\n");
  printf("    USER FILE MAINTENANCE MENU ");
  printf("\n\n\n\n");
  printf("         Enter one of the following:\n\n");
  printf(" A   Add new username.\n\n");
  printf(" D   Delete User.\n\n");
  printf(" L   List Usernames already in file\n\n");
  printf(" ?   Type menu in full.\n\n");
  printf(" Q   quit.\n\n");
  printf("\n\n\n");

}

  /* this function returns true if password accepted;
     false otherwise                                   */


password()
{
  clrscr();
  i = 0;
  k = 0;
  printf("Enter password:");
  addr = getpass(prompt);
  j = stricmp(addr,"sysm");

  while (j < 0 || j > 0) {
    printf("User authorisation failure\n");
    printf("Enter password:");
    addr=getpass(prompt);
    j=stricmp(addr,"sysm");
    ++i;

    if (i > 2)
      exit(0);
    /* endif */

  } /* endwhile */

}


del_record()
{
  char string[36];
  char recdel[255];

  clrscr();

  printf("Are you sure you want to erase a user ?(Y/N):");
  gets(response);
  clrscr();
  while (tolower(response[0]) != 'y' && \
        tolower(response[0]) != 'n') {
    printf("invalid option\n");
    printf("Sure ? (Y/N):");
    gets(response);

  }

  if (tolower(response[0]) == 'n')
    return;
  else {
    test = FALSE;
    while (test == FALSE) {
      printf("Enter record to delete:");
      gets(string);
      sfl=fopen("C:\\PROJECT\\FDIR\\USERF.DAT","r");
      fseek(sfl,0L,SEEK_SET);
      fread(record,36,1,sfl);

      while (!feof(sfl) ) {
        j = stricmp(record,string);
        if (j == 0)
          break;
        else
          fread(record,36,1,sfl);
      }  /* end while */
```

```
if (feof(sfl))
  printf("username does not exist\n");
else {
  fseek(sfl,OL,SEEK_SET);
  fread(record,36,1,sfl);

  temp = fopen("TEMP.DAT","w");

  while (!feof(sfl) ) {
    j = stricmp(string,record);
    if (j < 0 ¦¦ j > 0)
      fwrite(record,36,1,temp);
      fread(record,36,1,sfl);
  }  /* end while */

  fcloseall() ;

 /* copy back the records to the user file minus
    the tagged record */

  sfl  = fopen("C:\\PROJECT\\FDIR\\USERF.DAT","w");
  temp = fopen("TEMP.DAT","r");
  fseek(sfl,OL,SEEK_SET);
  fseek(temp,OL,SEEK_SET);

  while (!feof(temp))  {
    fread(record,36,1,temp);
    fwrite(record,36,1,sfl);
  }

  fcloseall();

  strcpy(recdel,"del");
  strcat(recdel," ");
  strcat(recdel,"TEMP.DAT");

  /* delete the temporary file */

  system(recdel);

  printf("username successfully deleted\n");
}
do {
  printf("Delete another ? (Y/N):");
  gets(response);
  if (tolower(response[0]) != 'y' && \
    tolower(response[0]) != 'n')
    printf("invalid option\n");
} while (tolower(response[0]) != 'y' \
  && tolower(response[0]) != 'n' );

if (tolower(response[0]) == 'n') {
  clrscr();
```

```
          test = TRUE;
        }

        fcloseall();

      }

    }

}


 /* this function lists usernames */


list_users()
{
  clrscr();
  i=0;
  lfil = fopen("C:\\PROJECT\\FDIR\\USERF.DAT","r");
  if (!lfil)
    messg(1, ERROR-%FOP- User file opening fail)
  else {
    fseek(lfil,0L,SEEK_SET);
    fread(record,36,1,lfil);
    do  {
      ++i;
      printf(" %s \n",record);
      if (i > 23) {
        printf("Press any key to continue ...........\n");
        ch=getch();
        i=0;
      }

      fread(record,36,1,lfil);

    }  while(!feof(lfil));

    fclose(lfil);
  }
  return;
}
```

```
/* ///////////////////////////////////////////////////////////////

                 Program: FFTOVAX
  The program scans the message files in the default FidoNet
  directory for receiving mail and changes the extension
  to .TXT . Expects the user maintainance file in the
  directory C:\PROJECT\FDIR

                  written by:
                  Odemba L.O
                  P64/7297/92.

  /////////////////////////////////////////////////////////////// */

#include <stdio.h>
#include <dir.h>
#include <process.h>
#include <dos.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>

#define messg(messno, mtext)  { puts( "ERROR-"  #mtext); exit(messno);

int  gfile,cntrl,rstate, nfil, eqstat;
long curpos1,curpos2;
char stdchr[36], *fext=".TXT", *pathname;
char filepath1[MAXPATH], filepath2[MAXPATH], filepath3[MAXPATH];
char drive1[MAXDRIVE], drive2[MAXDRIVE], dir1[MAXDIR], dir2[MAXDIR];
char filename1[MAXFILE], filename2[MAXFILE];
char ext1[MAXEXT], ext2[MAXEXT];
FILE *fptr1, *fptr2;

struct ffblk ffblk;
struct accpt{
  char sender[36];
  char receiver[36];
} fhinfo;


main(int argc, char *argv[])
{
 if (argc < 2) {
   printf("give full pathname as the second argument");
   exit(5);
 }
 else  {
   if (argc == 2)
     strcpy(filepath1,argv[1]);
   else
     messg(5, %TMP- too many parameters)
 }
```

```
/* setting the file path for reading */

cntrl = 0;
nfil = 0;
fnsplit(filepath1,drive1,dir1,NULL,NULL);
gfile=findfirst(filepath1,&ffblk,0);
if (gfile < 0 || gfile > 0)
  messg(6, %NFF- no message files);
while(!gfile ) {
  fnsplit(ffblk.ff_name,NULL,NULL,filename1,ext1);
  fnmerge(filepath2,drive1,dir1,filename1,ext1);
  fptr1=fopen(filepath2,"rt");
  if(!fptr1)
    messg(1,%EOFR- opening file for reading)
  else {
    curpos1=ftell(fptr1);
    fseek(fptr1,curpos1,SEEK_SET);
    fread(fhinfo.sender,36,1,fptr1);
    fread(fhinfo.receiver,36,1,fptr1);
    fclose(fptr1);
  }

  /* checking whether the message is for VAX and if so
     changes the file extention to .TXT                         */

  if (cntrl== 0) {
    pathname = "C:\\PROJECT\\FDIR\\USERF.DAT" ;
    fptr2=fopen(pathname,"rt");
      if (!fptr2)
         messg(1,%EOFR- error opening user file)
    cntrl=1;
  }

  fseek(fptr2,0L,SEEK_SET);
  fread(stdchr,36,1,fptr2);
  do  {
    eqstat=stricmp(stdchr,fhinfo.receiver);
    if (eqstat == 0) {
      fnsplit(filepath2,drive2,dir2,filename2,ext2);
      fnmerge(filepath3,drive2,dir2,filename2,fext);
      rstate=rename(filepath2,filepath3);
      if (rstate== -1)
        messg(4, %CNRF files cannot be renamed)
      else {
        if (rstate==0)
          nfil++;
      }
      break;
    }
    else {
      fread(stdchr,36,1,fptr2);
      continue;
    }
    } while (!feof(fptr2));                          /* end do */
    gfile=findnext(&ffblk);
  }                                  /* end while */
   clrscr();
   printf("%d m e s s a g e   f i l e s   f o u n d\n",nfil);
   fclose(fptr2);
 }
```

```
/* *****************************************************************

                 Program: TOFBASE
   The program prepares VAX external mails and call the
   AUTOMAIL program to transfer them to the FidoNet
   message base.
                 written by:
                 Odemba L.O.R
                 P64/7297/92

*******************************************************************/

#include <stdio.h>
#include <process.h>
#include <dir.h>
#include <dos.h>
#include <stdlib.h>
#include <string.h>

#define messg(messno, mtext) { puts("ERROR" #mtext); exit(messno);}

int  tran, gfile;
char path1[MAXPATH], path2[MAXPATH], path3[MAXPATH];
char path4[MAXPATH],path5[MAXPATH],path6[MAXPATH];
char path7[MAXPATH],drive1[MAXDRIVE], drive2[MAXDRIVE];
char dir1[MAXDIR], dir2[MAXDIR];
char fname1[MAXFILE], fname2[MAXFILE];
char ext1[MAXEXT], ext2[MAXEXT], sfail[255];
char sname[9],rname[9],dummy[1];

FILE *fp;

struct ffblk ffblk;

main(argc, argv)
int argc ;
char *argv[];
{

  /* create a pathname */

  char *pathname = "C:\\FD\\AUTOMAIL.EXE" ;

  /*create an argument string */

  char *args[] = {"automail.exe",
  "                              ",
  NULL};


  if (argc < 2)
    messg(5, %SFM- file name not supplied as second arg)
  else {
    if ( argc == 2)
      strcpy(path1,argv[1]);
    else  {
       if (argc > 2)
         messg(5,%INP- incorrect number of parameters)
    }

  }

  strcpy(path2,path1);
  gfile=findfirst(path1,&ffblk,0);
  while(!gfile) {
    printf("file is %s\n",ffblk.ff_name);
    fnsplit(ffblk.ff_name,NULL,NULL,fname1,ext1);
    fnsplit(path2,drive1,dir1,NULL,NULL);
    fnmerge(path3,drive1,dir1,fname1,ext1);
    strcpy(args[2],path3);
    fp = fopen(path3,"r");
    fseek(fp,0L,SEEK_SET);
    fread(sname,9,1,fp);
    fread(dummy,1,1,fp);
    fread(rname,9,1,fp);


    fclose(fp);


    strcpy(args[1],rname);

    /* call automail as a child process */

    tran = spawnv(P_WAIT,pathname,args);

    if (tran < -1 || tran == -1)
      messg(2, %SPF- spawn... process failed)
    else {
      strcat(sfail,"rename");
      strcpy(path3,path4);
      fnsplit(path4,drive2,dir2,fname2,ext2);
      if (tran == 0)
        fnmerge(path5,drive2,dir2,fname2,".MOK");
      else
        fnmerge(path5,drive2,dir2,fname2,".MIS");
      strcat(sfail,path3);
      strcat(sfail,path5);
      system(sfail);
    }
    gfile = findnext(&ffblk);
  }
}
```

APPENDIX C: SOME BATCH FILES:

FIDOMAIL

```
rem this batch command file collects the FidoNet mail
rem from the VAX and transfers them to the FidoNet
rem message base

@echo off
cls
:start
  echo collecting remote node messages from VAX ................
  n:
  cd \fmail
  if not exist .\*.* goto end
  c:
  tofbase n:\fmail\*.*
  if exist *.mis copy *.mis c:\project\waste
  cls
  echo FidoNet mail processing completed
  goto finish
:end
  cls
  echo N o   m e s s a g e s   f o u n d !
:finish
  cd \fd
  fdm
```

VAXMAIL

```
@echo off
rem changes the extension for files destined for
rem VAX from the default .MSG to .TXT. The messages
rem are then transferred to the file server volume
rem directory.
cls
echo                      s   y   s   t   e   m
echo                 s o r t i n g   V A X   m a i l
cd \
cd \fd\mail
if not exist .\*.msg goto nomail
 cd \project
 FFTOVAX  c:\fd\mail\*.msg
 cd\
 cd \fd\mail
 nft copy/ascii/stream;
        *.txt kamili"pgd92001"::disk$user3:[pgd.pgd92001.vmail]
 exit
 cls
 cd\
 cd fd\mail
 erase *.txt
 cls
 echo V A X   m a i l   p r o c c e s s i n g   c o m p l e t e d
 goto end
:nomail
   echo n o   V A X   m a i l   f o u n d
:end
 cd \fd
```

```
$!xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
$!                                                                 x
$!              VAX COMMAND PROCEDURE PROGRAM                      x
$!                                                                 x
$!                     name: DMAIL                                 x
$!                                                                 x
$!                                                                 x
$!The command procedure program is run by the system operator,    x
$!in the VAX/vms environment, to distribute the received          x
$!FidoNet mail to the respective VAX users.                       x
$!                                                                 x
$!                   written by:                                   x
$!                   Odemba L.O.R                                  x
$!                   P/64/7297/92                                  x
$!xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
$!
$begin:
$ set def [.vmail]
$!
$time=f$time()
$ progress:
$ clr
$ spaces = "                          "
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output "            ********VAX  MAIL  DISPATCH IN PROGR
$ write sys$output spaces,time
$ save_verify_image=f$environment("verify_image")
$ save_verify_procedure=f$verify(0)
$!
$ p1=f$parse(p1,"*.*")
$ first_time = "true"
$loop:
$ filespec=f$search(p1)
$!
$!******find next file *******
$!
$ if filespec .eqs. "" then goto end_search
$!else perform search_file
$ if .not. first_time then goto search_file
$!
$ first_time = "false"
$!
$ dirspec=f$parse(filespec,,,"device")-
       +f$parse(filespec,,,"directory")
$!
$search_file:
  filename = f$parse(filespec,,,"name")-
           +f$parse(filespec,,,"type")
```

```
$!
$ open/read fname 'filename'
$ read fname record
$!
$ sender=f$extract(0,36,record)
$ username=f$extract(36,36,record)
$ subj1=f$extract(72,72,record)
$!
$ dummy0 := 'subj1'
$ dummy1 = """"
$ dummy2 = dummy1 + dummy0
$ dummy3 = """"
$ dummy4 = dummy2 + dummy3
$ subj   = dummy4
$!
$ mail/subject='subj' 'filename' 'username'
$!
$ close fname
$ goto loop
$end_search:
$ clr
$!
$!********* display completion banner *************
$!
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output spaces
$ write sys$output spaces
$!
$ write sys$output "      ********** VAX MAIL DISPATCH COMPLETED******
$ write sys$output spaces,time
$!
$ delete *.txt;*
$!transfer control to the operator directory
$  set def [-]
$  exit
$!
$!                              xxxx THE END xxxx
```