BROWN UNIVERSITY
Department of Computer Science
Master's Thesis
CS-91-M16

THEODORA: User's Manual for Version 0.6

by
Spyros-Nicholas Bartsocas

THEODORA

User's Manual for version 0.6

The Problem: How does a user efficiently read and respond to electronic mail when he is away from his computer?

In environments where electronic mail is used heavily, people have to check their mailbox very often. Checking mail can be problematic if you are away from your 'home' machine. Unfortunately a mailbox is physically located on a specific machine, which in turn is physically located in a specific geographical place. It is possible for a user's mail to be forwarded to another machine, but forwarding requires that the user have a mailbox on the other machine. It is not very likely that a user can have a mailbox on systems in every place to which he travels. Also, setting up mail forward is not worth the trouble if the trip is a short one. So mail forwarding is not a solution to this problem.

Advances in technology over the past few years have made common new "Laptop", "Notebook" and even smaller size computers which have the power of desktop personal computer. The new generation of laptop computers features cellular modems, simplifying the procedure of connecting to a remote computer. One can travel with a laptop and remotely access files, including mail, on one's 'home' computer via modem. This setup depends only on the availability and the reliability of the local phone system. Unfortunately, long distance phone calls can be very expensive. Line noise can make dial-in sessions both long and frustrating. Thus remote access solves the problem of how to get one's electronic mail, but a means of reducing the remote connection time is needed.

The Solution: Download a compressed copy of the mail file, manipulate it locally off-line, and send back a list of changes to be made to the remote mail file.

Theodora attempts to address all these problems. She connects to the computer that hosts the mailbox and downloads a compressed copy of the mailbox . Then the connection is broken and the user can manipulate the mailbox on the portable computer in the same way that he would do so on the remote system. Theodora executes the changes on the local copy of the mailbox and keeps information about changes that have to be made to the remote copy of the mailbox. When the user is ready, he reconnects to the remote host, and Theodora sends a list of commands to the host to be executed on the mailbox. The user can break the connection and let the remote computer work on those changes on its own time.

Theodora always appends to files. She never deletes or overwrites them. She can not get any other files except for the mailbox, so the user's files are secure. The current Theodora set-up includes a program running under Unix which is controlled from an MS-DOS program and an MS-DOS port of Unix mail. A Unix user, even if he has never used MS-DOS, before should have no problems using this software, as the interface is exactly the same as the Unix program. The advantages are:

a. The user does not depend on any local resources except for the phone line

b. The cost of the phone call is minimized as data is being transferred the whole time the two computers are connected, and almost all data sent is in compressed form. The user does not have to deal with phone line noise as noise is taken care of by the communication protocol.

c. The user has a copy of the mail received, which he can read or print as many times as he likes without having to stay on line.

d. The MS-DOS mail reader uses the same interface as the mail program on the host computer. The user does not need to learn a new interface.

Savings:

To estimate minimal savings let's make some totally unrealistic assumptions. Let's assume a user who is perfect: one who makes no typing errors and is able to type at the speed his modem can transmit data, who can read and memorize his mail as fast as his modem can received it, and who already knows exactly what to type in his outgoing messages. It would still be cheaper to use Theodora to access mail than through the regular Unix tools. Compress reduces the size of the mail to about 40% of the original. When the compressed file gets uuencoded, its size grows by 35% to 55% of the original. As a results the savings from compression are at least 35%. Note that the most important savings do not come from compression, but from most commands taking place off-line. The length of the phone call is more than cut in half, resulting in less than half the long distance charges.

Using Theodora

Send: get or send new mail

Type **SEND** on the DOS prompt. (This assumes a 2400[1] baud hayes-compatible modem connected to serial port COM1:). You will be greeted by the Theodora start-up banner indicating the version of the software you are using. SEND is the part of Theodora that handles all the communication for the MS-DOS part of the program.

The following instructions describe how to access the Brown University Computer Science Department Computers. To access any other computer the procedure is similar, but will differ in at least the phone number used. Type: ATDT and the phone number. For example to call from out of state you would dial **ATDT14014575120**. If the modem has a speaker you will hear the number being dialed, the phone ringing and eventually the call being answered. A CONNECT 2400 line will appear and a pound sign ('#') prompt (the localnet/20 prompt). You are now connected to BRUNET. Typing **CALL 1110** will connect you to the CS department annex. If nothing happens press Control-Q. When the annex prompt appears you can login to your favorite workstation with the rlogin command. (for example **rlogin igor**). Enter your userid and password when prompted. When your shell prompt comes up press the F1 key on the PC keyboard[2]. Theodora will then ask you to wait while she transfers mail and commands. A dot will get printed on the screen for every line received from the workstation. Note that in order for this to work the workstation part of Theodora has to be in your path. If you want to quit, before all transactions are completed, simply press the 'Q' key on the PC keyboard. When all transactions are made the Unix shell prompt will appear. You can then logout as you normally would, or enter any Unix command. To hang up press F4. To exit the SEND program hit Alt-F10 on the laptop keyboard.

1. To connect to the 1200 baud lines you need to give SEND the following parameters:

-b1200 -pe -d7

The phone number for the 1200 baud lines is (401) 457-5000

2. This assumes that you have "proto" (the workstation side of Theodora) in your path. If you do not change directory to the one proto is located

SEND is really a communications program with a simple VT100 emulator. Because it does not have many of the fancy features of other terminal emulation programs, it is very fast. When SEND starts up, it initializes the modem and waits for user commands. Everything typed at that point is sent to the modem (except for the functions keys listed bellow). After you hit F1, the program takes control and tries to communicate with the workstation side. Your laptop controls the session until it is done with its task, or until you interrupt it by hitting any key on the keyboard. Note that the key you press will be sent to the workstation side.

You control what Theodora will do after you hit F1 by command line options when starting up the SEND program. If no option is given it assumes you want to download all unread messages. This is the same as specifying the GETALL option. To get a specific message (which can be read or unread), use SEND with the LIST command option to get a list of all mail in your system mailbox, use the LIST program to select the mail you want, and then start SEND with the GET option. Note that mailbox changes are automatically made when you connect, and can not be turned on or off.

Hitting F2 will make SEND dial the string stored in the NUMBER environment variable. If the string is longer than 40 characters, SEND will dial it in groups of 39 characters of shorter. To set the variable type SET NUMBER=string at the DOS prompt. For example to make SEND dial 401-555-1212, you would type: SET NUMBER=4015551212. Any other character which is valid in a dial string (0-9,*,#,A,B,C,D,P,T,',',W) is valid except for the semi-colon. Send assumes it has the modem's attention. It waits until the modem replies. If the modem does not connect it prints its own educated guess of why the connection was not made. If you want to quit before the connection is made, hit return. The modem will be left in command mode

If you want to dial a number manually, Press F3. Dial the number on the telephone. After you dial the last digit, hit return on the PC keyboard. You can now hang up the phone. The modem will try to connect with the remote modem.

To hang up just hit F4. Three plus signs will appear. They will be followed by the string ATH0 which tells the modem to hang up. Please note that Quitting (Alt-F10) does not hang up. If Theodora is connected through a dedicated line instead of a modem, there is no way to hang up.

Important keys for SEND:

F1: Run the selected commands.
F2: Dial the number stored in the NUMBER environment variable.
F3: Dial manually.
F4: hang up phone.
F5: Set current position as tab stop for VT100 terminal
F6: Unset current position as tab stop for VT100 terminal
Alt-F1: Prints a summary of keyboard commands
Alt-F8: Send a Break to the remote host.
Alt-F9: Open a new DOS shell
Alt-F10: Quit SEND (This will not hang up the phone).

Command line options:

Processing Options:
-GET: get messages selected through the LIST program
-GETALL: get all new messages
-PUT: upload messages composed on the PC

-LIST: get a listing of mail in the workstation mailbox
You can give any number of processing options. Note that GET is incompatible with GETALL. If no processing option is entered "GETALL" and "PUT" will be assumed

Communication Options:
-Bx: Will set the baud rate to x. The default is 2400.
-Cx: Will set the communications port to COMx. The default is COM1:.
-Dx: Will set the number of data bits to x. The default is eight.
-Px: Will set the parity to x. The default is no parity.
-Sx: WIll set the number of stop bits to x. The default is one.
If no communication option is entered the defaults above are assumed.

To prepare the mail for reading:

When mail is received by your DOS machine it is compressed and encoded. In order to read it you have to type **PREPARE**. Prepare will call all the programs needed to convert your mail into the format used by the mail program. The processes performed by these commands include uncompressing the received mail and converting the file from a Unix format to MS-DOS. After running PREPARE any DOS application can use the MAIL.IN file. If you already have a MAIL.IN file, PREPARE will append the new mail to the end of your old file.

MAIL: read or compose mail

MAIL is the command that allows you to compose, send, and receive electronic messages. Mail is an adaptation of Berkeley mail. A list of differences is given at the end of this section. While reading messages, MAIL provides you with commands to browse, display, save, delete, and respond to messages. While sending mail, MAIL allows editing and reviewing of messages being composed, and the inclusion of text from files or other messages. Incoming mail can be found in the file MAIL.IN. You can use the MAIL environment variable to have it look in a different file. When you read a message, it is marked to be moved to a secondary file for storage. This secondary file, called the mbox, is normally the file mbox in your home directory on the workstation, or the current directory on the PC. The location of the PC file can also be changed by setting the MBOX environment variable. Messages remain in the mbox file until deliberately removed. When started, mail reads commands from a private start-up file called the mail.rc file. In your mail.rc file you should have personal commands and variable settings. Most mail commands are legal inside this file. This file stores your personal commands and variable settings. It plays the exact same role as the .mailrc file in your home directory on Unix workstations, and it is recommended that you download your Unix one. The file name and location can be changed by setting the MAILRC environment variable. The most common uses for this file are to set up initial display options and alias lists. The following commands are not legal in the start-up file: !, edit, hold, mail, preserve, reply, Reply, shell, and visual. Any errors in the start-up file cause the remaining lines in that file to be ignored. You can use the MAIL command to send a message directly by including names of recipients as arguments on the command line. When no recipients appear on the mail command line, it enters command mode, from which you can read messages sent to you. If you list no recipients and have no messages, MAIL prints the message: 'No mail for username' and exits. When in command mode (while reading messages), you can send messages using the mail command.

Options:

-?: Prints the syntax of the MAIL command.

-b users: Give list of blind carbon copy recipients.
-c users: Give list of carbon copy recipients list
-f [filename]: Read messages from filename instead of MAIL.IN. If filename is omitted mbox in the "home" directory will be used.
-i: Ignore Control-Break.(can also be set with the *ignore* variable in mail)
-I: run mail in interactive mode.
-s subject: Set the subject of the mail message.
-N: Do not print initial header summary.
-n: Do not initialize from the mail.rc file.

Sending Mail

To send mail type **MAIL** ***recipients*** where ***recipients*** is the addresses of the people you want to send mail to. Recipients can be any recipient that would be valid for the Unix mail command except for files and pipes. While you are composing a message to send, mail is in input mode. If no subject is specified as an argument to the command, a prompt for the subject is printed. After entering the subject line, MAIL enters input mode to accept the text of your message to send.

As you type in the message, mail stores it in a temporary file. To review or modify the message, enter the appropriate tilde escapes, listed below, at the beginning of an input line.

To indicate that the message is complete, type a dot (or EOF character, CTRL-Z or F6 for MS-DOS) on a line by itself. mail appends the message in a file called $SEND$.TMP Before you use SEND to send the mail to its recipients, you have to use the UP program. UP will convert the $SEND$.TMP to a Unix file and compress it. For more information read the "before sending back commands" section bellow.

Tilde Escapes

The following tilde escape commands can be used when composing messages to SEND. Each must appear at the beginning of an input line. The escape character (~), can be changed by setting a new value for the escape variable. The escape character can be entered as text by typing it twice.

~! [*shell-command*]
Escape to the shell. If present, run shell-command.

~:[*mail-command*]
Run a mail command while composing mail. Mail commands are listed on page 8

~.
Simulate EOF (terminate message input). This is the only way that will always work for indicating the completion of a mail message. Control-Z can be overridden by setting the *ignoreeof* variable, and the dot method by setting the *nodot* variable. For more information about this variables take a look at the "Mail Variables" section on page 13.

~?
Print a summary of tilde escapes. This is done by displaying the TILDE.HLP file.

~b *name ...*
Add the names to the blind carbon copy (Bcc) list. This is like the carbon copy (Cc) list, except that the names in the Bcc list are not shown in the header of the mail message.

~c *name ...*
Add the names to the carbon copy (Cc) list.

~d
Read in the DEAD.LET file. The name of this file is listed in the variable *DEAD*.

~e
Invoke the editor to edit the message. The name of the editor is listed in the *EDITOR* variable. The default editor is edlin.

~f [*message-list*]
Forward the listed messages, or the current message being read. Valid only when sending a message while reading mail; the messages are inserted without alteration (as opposed to the ~m/~M escape).

~F [*message-list*]
same as ~f, but keep all header lines

~h
Prompt for the message header lines: Subject, To, Cc, and Bcc. The new text inserted replaces the original text. Note that this is different from the way Unix mail works where you can edit the text which was originally in the header fields.

~m [*message-list*]
Insert text from the specified messages, or the current message, into the letter. Valid only when sending a message while reading mail; the text in the message is shifted to the right, and the string contained in the *indentprefix* variable is inserted as the leftmost characters of each line. If *indentprefix* is not set, a TAB character is inserted into each line. To forward mail without alteration use the ~f/~F escape.

~M [*message-list*]
Same as ~m, but keep all header lines.

~p
Print the message being entered.

~r *filename*
Read in text from the specified file.

~s *subject*
Set the subject line to subject.

~t *name ...*
Add each name to the list of recipients.

~v
Invoke a visual editor to edit the message. The name of the editor is listed in the *VISUAL* variable. EDIT is the default editor. As EDIT is available only with MS-DOS 5.0 and above, this test version of Theodora is distributed together with a shareware full-screen editor.

~w *filename*
Write the message text onto the given PC file, without the header.

If in your .mailrc in your home directory on the workstation you set the *record* variable with a filename to save a record of your outgoing mail the workstation side of Theodora will use it. This is the only use Theodora makes of your .mailrc file.

Reading Mail

If you have downloaded new mail, you can not access it until you run the PREPARE program. PREPARE will uncompress the new mail, and append it to the MAIL.IN file. See the "to prepare mail for reading " section above. When you enter command mode in order to read your messages, mail displays a header summary of the first several messages, followed by a prompt for one of the commands listed below. The prompt is the & (ampersand character).

Message are listed and referred to by number. There is, at any time, a current message, which is marked by a > in the header summary. For commands that take an optional list of messages, if you omit a message number as an argument, the command applies to the current message.

A message-list is a list of message specifications, separated by SPACE characters, which may include:

. The current message.
n Message number n.
^ The first undeleted message.
$ The last message.
+ The next undeleted message.
- The previous undeleted message.
* All messages.
n-m An inclusive range of message numbers.
user All messages from user.
/string All messages with string in the subject line (case ignored).
:c All messages of type c, where c is one of:
d deleted messages
n new messages
o old messages
r read messages
u unread messages

Note: the context of the command determines whether this type of message specification makes sense.

Additional arguments are treated as strings whose usage depends on the command involved. File-names, where expected, are not expanded. Special characters, recognized by certain commands, are documented with those commands.

Commands:

While in command mode, if you type in an empty command line (a RETURN or NEWLINE only), the print command is assumed. The following is a complete list of mail commands:

! [*shell-command*]

Escape to the shell. The name of the shell to use is listed in the COMSPEC variable. If no command is given a new DOS shell will start. You can return back to mail by using the DOS exit command.

# arguments

The comment command. This may be used the same way as comments in mail.rc files, but note that it must be separated from its arguments (commentary) by white space.

=

Print the current message number.

?
Print a summary of commands. This is done by printing to the screen the MAIL.HLP file. Same as the help command.

alias [*alias recipient....*]
Declare an alias for the given list of recipients. When alias is used as a mail recipient mail will substitute the list. With no arguments displays the list of defined aliases. Same as the group command.

alternates [*name* ...]
Declare a list of alternate names for your login. This way mail will not send copies to these addresses when replying to mail. With no arguments prints current list of alternate names. It is recommended that you put your workstation login as an alternate name. The MS-DOS part of Theodora does not know what you workstation login is.

cd [*directory*]
chdir [*directory*]
Change directory. One difference from the MS-DOS commands with the same name is that it uses the Unix convention of changing to the HOME directory when no parameter is given, instead of the MS-DOS convention of printing the current directory.

copy [message-list] [filename]
Copy messages to the file without marking the messages as saved. Otherwise equivalent to the save command.

delete [*message-list*]
Delete messages from the system mailbox. If the variable *autoprint* is set, print the message following the last message deleted. Deletions are stored in the file $MAIL$.TMP and sent to the workstation during the next remote connection.

discard [*header-field...*]
Suppress printing of the specified header fields when displaying messages on the screen, such as "Status" and "Received". The fields are included when the message is saved unless the variable *alwaysignore* is set. The Print and Type commands display all header fields, ignored or not. Same as the ignore command

dp [*message-list*]
dt [*message-list*]
Delete the specified messages from the system mailbox, and print the message following the last deleted message. Equivalent to a delete command followed by a print command.

echo [*string* ...]
Echo the given strings. The following characters have special meaning to echo:
% Your system mailbox, usually MAIL.IN
# The previous mail file.
& Your mbox file (of messages previously read), usually MBOX
+filename The named file in the folder directory (listed in the folder variable).
~ As the first character your HOME directory.

edit [*message-list*]
Edit the given messages. The messages are placed in a temporary file and the *EDITOR* variable is used to get the name of the editor. The default editor is edlin.

exit
Exit from mail without changing the system mailbox. No messages are saved in the mbox (see also quit). Same as xit command.

file *[filename]*
folder *[filename]*
Quit from the current mailbox file and read in the named mailbox file. Several special characters are recognized when used as file names:

% Your system mailbox, usually MAIL.IN
# The previous mail file.
& Your mbox file (of messages previously read), usually MBOX
+filename The named file in the folder directory (listed in the folder variable).
~ As the first character will start the search at your HOME directory instead of the current.

With no arguments, file prints the name of the current mail file, and the number of messages and characters it contains.

folders
Print the name of each mail file in the folder directory.

from [*message-list*]
Print the header summary for the indicated messages or the current message.

group [*alias recipient...*]
Declare an alias for the given list of recipients. When alias is used as a mail recipient, MAIL will substitute the list. With no arguments displays the list of defined aliases. Same as the alias command.

headers [*message*]
Print the page of headers that includes the message specified, or the current message. The screen variable sets the number of headers per page. See also the z command.

help
Print a summary of commands. This is done by printing to the screen the MAIL.HLP file. Same as the '?' command.

hold [*message-list*]
Hold the specified messages in the system mailbox. Same as the preserve command.

if s|r|t
mail-command
...
else
mail-command
...
endif
Conditional execution, where 's' will execute following mail-command up to an else or endif, if the program is in send mode, 'r' executes the mail-command only in receive mode, and 't' executes the mail-command only if mail is being run from a terminal. Useful in the mail.rc file.

ignore
Ignore Control-Break and Control-C.

list
Prints all commands available. No explanation is given.

mail ***recipient ...***
Mail a message to the specified recipients. Save as typing MAIL recipient at the DOS prompt.

mbox **[*message-list*]**
Arrange for the given messages to end up in the mbox file on the remote computer. MAIL has to terminate normally. At quit time an MBOX command will be placed in the $MAIL$.TMP file. See also the exit and quit commands.

more **[*message-list*]**
Prints the messages using the program defined in the *PAGER* variable. If no program is defined the MORE program is used. If MORE, or the program specified in the *PAGER* variable can not be found, this command works the same as the print/type command. Uses the *crt* variable if set. As the MS-DOS version of more can not be used with MAIL, a public domain more is included with this package. Same as the page command

More **[*message-list*]**
Prints the messages using the program defined in the *PAGER* variable, including all header fields. If no program is defined the MORE program is used. If the MORE program, or the program defined in the *PAGER* variable is not found, this command is the same as the Print/Type command. Overrides suppression of fields by the ignore and retain commands. As the MS-DOS version of more can not be used with MAIL, a public domain more is included with this package.

next ***message***
Go to next message matching message. A message-list can be given instead of message, but only first valid message in the list is used. (This can be used, for instance, to jump to the next message from a specific user.)

page **[*message-list*]**
Prints the messages using the program defined in the *PAGER* variable. If no program is defined the MORE program is used. If MORE, or the program specified in the *PAGER* variable can not be found, this command works the same as the print/type command. Uses the *crt* variable if set. Same as the more command. As the MS-DOS version of more can not be used with MAIL, a public domain more is included with this package.

Page **[*message-list*]**
Prints the messages using the program defined in the *PAGER* variable, including all header fields. If no program is defined the MORE program is used. If the MORE program, or the program defined in the *PAGER* variable is not found, this command is the same as the Print/Type command. Overrides suppression of fields by the ignore and retain commands. Same as the More command. As the MS-DOS version of more can not be used with MAIL, a public domain more is included with this package.

preserve **[*message-list*]**
Hold the specified messages in the system mailbox. Same as the hold command.

print **[*message-list*]**
Print the specified messages. Same as the type command.

Print **[*message-list*]**

Print the specified messages on the screen, including all header fields. Overrides suppression of fields by the ignore and retain commands. Same as the Type command.

quit
Exit from mail storing messages that were read in the mbox file and unread messages in the MAIL.IN file. Messages that have been explicitly saved in a file are deleted unless the variable *keepsave* is set.

reply [*message-list*]
respond [*message-list*]
Sends a response to the author of each message in the message-list. The subject line is taken from the first message. If record is set to a filename, a copy of the reply is added to that file. If the *replyall* variable is set, the actions of Reply/Respond and reply/respond are the same.

Reply [*message*]
Respond [*message*]
Reply to the specified message, including all other recipients of that message. If the variable *record* is set to a filename, a copy of the reply added to that file.
retain [header-field...]
Add the list of header fields named to the retained list. Only the header fields in the retain list are shown on your terminal when you print a message. All other header fields are suppressed. The set of retained fields specified by the retain command overrides any list of ignored fields specified by the ignore command. The Type and Print commands can be used to print a message in its entirety. If retain is executed with no arguments, it lists the current set of retained fields.

save [*message-list*] [*filename*]
Save the specified messages in the named file. The file is created if it does not exist. If no filename is specified, the file named in the MBOX variable is used, mbox in your HOME directory by default. Each saved message is deleted from the system mailbox when mail terminates unless the *keepsave* variable is set. A SAVE command will be added to your $MAIL$.TMP file. See also the exit and quit commands. Same as the write command.

savediscard [*header-field* ...]
saveignore [*header-field* ...]
Suppress printing of the specified header fields when saving messages to disk, such as "Status" and "Received". The fields are included when the message is saved unless the variable *alwaysignore* is set. The save/write command saves all header fields, ignored or not. Without any parameters prints out the list of saveignored fields.

saveretain [*header-field* ...]
Add the list of header fields named to the saveretained list. Only the header fields in the saveretain list are saved when you write a message. All other header fields are suppressed. The set of retained fields specified by the retain command overrides any list of saveignored fields specified by the savediscard/saveignore command. The save/write command do not use the saveretain list. If saveretain is executed with no arguments, it lists the current set of saveretained fields.

set [variable[=value]]
Define a variable. To assign a value to variable, separate the variable name from the value by an '=' (there must be no space before or after the '='). A variable may be given a null, string, or numeric value. To embed SPACE characters within a value enclose it in quotes. With no argu-

ments, set displays all defined variables and any values they might have. See Variables for a description of all predefined mail variables.

shell
Invoke the interactive shell listed in the COMSPEC variable.

size [*message-list*]
Prints the size in characters of the specified messages.

source *filename*
Read commands from the given file and return to command mode.

top [*message-list*]
Prints the top few lines of the specified messages. If the *toplines* variable is set, it is taken as the number of lines to print. The default number is 5.

touch [message-list]
Touch the specified messages. If any messages are touched either by this command or a print command, and are not saved, an MBOX command will be placed in the $MAIL$.TMP file upon normal termination, unless the *hold* variable is set. See also the exit and quit commands.

type [*message-list*]
Print the specified messages. Same as the print command.

Type [*message-list*]
Print the specified messages on the screen, including all header fields. Overrides suppression of fields by the ignore and retain commands. Same as the Print command.

undelete [message-list]
Restore deleted messages. This command only restores messages deleted in the current mail session. If the *autoprint* variable is set, the last message restored is printed.

unread [*message-list*]
Take a message list and mark each message as not having been read.

unset variable ...
Erase the specified variables. If the variable was imported from the environment (that is, an environment variable), it cannot be unset from within mail.

version
Print the current version and release date of the mail utility.

visual [message-list]
Edit the given messages with the screen editor listed in the VISUAL variable. Each message is placed in a temporary file for editing.

write [message-list] [filename]
Write the given messages onto the specified file. This is equivalent to the save command.

xit
Exit from mail without changing the system mailbox. No messages are saved in the mbox (see also quit). Same as exit command.

z[+|-]
Scroll the header display forward (+) or backward (-) one screenfull. The number of headers displayed is set by the screen variable.

Variables

The behavior of mail is governed by a set of predefined variables that are set and cleared using the set and unset commands.

Environment Variables:

MS-DOS Environment Variables are set and displayed using the MS-DOS **SET** command. To set variable A to value B just type SET A=B at the DOS shell prompt. To display the values of all variables type SET with no parameters at the DOS prompt

HOME: Location of MBOX and MAIL.RC files (default: current directory)
MAIL: file to find new mail in (default: MAIL.IN in current directory)
MAILRC: the MAIL.RC file (default: MAIL.RC in HOME directory)
USER: Your workstation login. Required if you want mail to know who you are.
TEMP: Where temporary files should be created (default: current directory)

Mail Variables:

The following variables can be initialized within the mail.rc file, or set and altered interactively using the set command. The unset command clears variables. The set command can also be used to clear a variable by prefixing the word "no" to the name of the variable to clear.

append
Upon termination, append messages to the end of the mbox file instead of prepending them. Default is *noappend.*

askcc
Prompt for the Cc list after message is entered. Default is *noaskcc.*

asksub
Prompt for subject if it is not specified on the command line with the -s option. Enabled by default.

autoprint
Enable automatic printing of messages after delete and undelete commands. Default is *noauto-print.*

crt=number
Pipe messages having more than number lines through the command specified by the value of the *PAGER* variable (MORE by default). Disabled by default.

DEAD=filename
The name of the file in which to save partial letters in case of untimely interrupt. Default is the file DEAD.LET in your HOME directory.

dot
Take a period on a line by itself during input from a terminal as EOF. Default is nodot.

EDITOR=DOS-command
The command to run when the edit or ~e command is used. Default is edlin. The only requirement for the program defined by this variable is that it can accept the filename to edit as a command line parameter.

escape=c
Substitute c for the ~ escape character.

folder=directory
The directory for saving standard mail files. User specified file names beginning with a plus (+) are expanded by preceding the filename with this directory name to obtain the real filename. If directory does not start with a slash (/), the value of HOME is prepended to it. There is no default for the *folder* variable. See also *outfolder* below.

header
Enable printing of the header summary when entering mail. Enabled by default.

hold
Preserve all messages that are read in the system mailbox instead of putting them in the standard mbox save file. Default is *nohold.*

ignore
Ignore interrupts while entering messages. Default is *noignore.*

ignoreeof
Ignore EOF during message input. Input must be terminated by a period '.' on a line by itself or by the '~.' command. Default is *noignoreeof.* See also *dot* above.

indentprefix=string
When *indentprefix* is set, string is used to mark indented lines from messages included with ~m. The default is a TAB character.

keep
When the system mailbox is empty, truncate it to zero length instead of removing it. Disabled by default.

keepsave
Keep messages that have been saved in other files in the system mailbox instead of deleting them. Default is *nokeepsave.*

LISTER=DOS-command
The command (and options) to use when listing the files in the *folder* directory. The default is dir.

MBOX=filename
The name of the file to save messages which have been read. The xit command overrides this variable, as does saving the message explicitly to another file. Default is the file mbox in your home directory.

metoo
If your login appears as a recipient, do not delete it from the list. Default is *nometoo.*

no
When used as a prefix to a variable name, has the effect of unsetting the variable.

outfolder
Locate the files used to record outgoing messages in the directory specified by the folder variable unless the pathname is absolute. Default is *nooutfolder*. See *folder* above and the Save command.

PAGER=DOS-command
The default pager to be used when paging is needed. MORE is the default command. The only requirement for the DOS-command used is that it can accept the file to display as a command line parameter. Unfortunately the MORE.COM file distributed with DOS does not fullfill this require-

ment. Use the MORE.EXE file distributed with Theodora instead. See crt above and the More/ Page and more/page commands.

quiet
Refrain from printing the opening message and version when entering mail. Default is *noquiet*.

record=filename
Record all outgoing mail in filename. Disabled by default. See also the variable *outfolder*.

replyall
Makes the reply command the same as the Reply command.

save
Enable saving of messages in the DEAD.LET file on interrupt or delivery error. See *DEAD* for a description of this file. Enabled by default.

screen=number
Set the number of lines in a screen-full of headers for the headers command.

toplines=number
The number of lines of header to print with the top command. Default is 5.

VISUAL=DOS-command
The name of a preferred screen editor. The only requirement for the DOS-command used is that it can accept the filename to edit as a command line parameter.

There are many different ways to specify the value of a variable that corresponds to a DOS command. The way you specify it changes the way DOS will look for it. If you enter a full pathname, only that pathname will be searched. If no pathname is specified the current directory will be searched first. If it is not found, then the path specified in the PATH environment variable will be searched. If you specify a file extension then only that extension will be searched. If you do not specify an extension, MAIL will first search for a .COM, .EXE and .BAT file in that order. Note that the default commands for EDITOR, LISTER, PAGER and VISUAL are searched without an extension.

Differences between Berkeley Mail and MAIL:
-MAIL can not send mail to files or pipes
-MAIL in input mode does not support inclusion of the output of a command
-Mail and MAIL use different file names, and default locations

To update the workstation mailbox:

Use the same procedure as in the get mail section. Theodora will automatically upload the script and then proceed to receive any new mail that might have come since your last connection.

Before sending back commands:

When you are ready to reconnect, you have to compress and uuencode the command file. In order to perform this processing you have to type UP. Up will call all the programs needed to convert the ASCII command file to a format that minimizes connect time.

List: select mail for reading

You can tell SEND to download a list of messages in your system mailbox by invoking it with the -LIST flag. You can then use the LIST program to select one or more messages for downloading. LIST displays the available messages in a manner similar to the mail 'headers' command. Each line of the display represents one message. The first column displays the status code for the message. The Theodora server produces one of the three following codes:

N: A new message

U: An unread message

 : A read message [A blank followed by a colon]

The next two columns contain the address of the person that sent the mail and the date and time that they sent it. This is followed by the number of lines and bytes in the message. Note that the number of bytes a message takes on a Unix workstation is smaller than the number of bytes it takes on an MS-DOS machine. The rest of the line is used to display the subject of the message. You can select a message by pressing RETURN, or by clicking with the mouse when it is highlighted. Follow the same procedure to cancel a selected message. Use the Up and Down arrow keys to move the highlight. LIST always displays as many messages as can fit on the screen. Press the PageUp, or PageDown keys to access teh messages that do not fit on the screen. If you try to page out of the screen, a beep will sound, and you will still be looking at the last page. Press Escape to exit or click the mouse outside the area of the messages. List can handle up to four hundread messages

Key Summary:

Up: Go up one item

Down: Go down one item

PageUp: Go up one screenfull

PageDown: Go down one screenfull

Enter: select current item

Esc: Update selection file and quit

FILES

The following is a list of files that make up the MS-DOS component of Theodora and an explanation of what each is:

COMMANDS: Temporary file used by UP.BAT to store the commands it is processing.

COMMANDS.Z: Compressed version of commands. This file is sent to the workstation side by SEND

COMPRESS.EXE: The program that compresses and uncompresses the mail and commands files.

CRLF.EXE: Used by the prepare program

DEAD.LET: undeliverable mail message

MAIL.EXE: The mail reader/editor

MAIL.HLP: Help for MAIL command mode commands

MAIL.IN: The incoming mail

MAIL.LST: the list of mail in the system mailbox

MAIL.RC: your private start-up

MBOX: similar to the file with the same name in your home directory

PREPARE.BAT: The program that prepares the mail to be used by the MS-DOS utilities

SEND.EXE: The communication program

SEND.HLP: Help for SEND keyboard commands

THEODORA.HLP: General information for MS-DOS component of program

TILDE.HLP: Help for MAIL tilde commands
UP.BAT: The program that prepares the commands to be uploaded
XCR.EXE: Used by the UP program
GET.TMP: The requests for the GET command of SEND
GET.OLD: The previous requests
$MAIL$.TMP: The outgoing script
$MAIL$.OLD: The previous outgoing script
$SEND$.TMP: The "prepared" outgoing script

Mail creates a small, but significant number of temporary files. These files get created in the directory the TEMP environment variable points to, or the current directory. The filenames of these files are always 8 characters long, and begin with an 'R'. The second character with be on of the following E, M, Q, S, X. If mail terminates abnormally (or the machine gets rebooted before it terminates) these files will stay around. You can delete them using the DOS **ERASE** command.

Example Use

For the following example we will assume that the user of the program has "tu" as her workstation login. The login directions are those a user would follow to access the Brown University Computer Science Department workstations. These might be different for a user in a different environment. What user types will appear in *italics*. System responses will appear in **bold**. Regular text are comments and would not appear in the real screen.

The user is far away from her workstation when she realizes that she wants to send mail to user abc. She starts up her laptop computer and types:

C>*set NUMBER=8,14014575120*

She first set the default number that will be used to call the mailbox machine. As the user's phone is located in a hotel PBX, she has to dial 8 before getting an outside line. The comma tells the modem to wait for two seconds before dialing the rest of the number. This only needs to be done once per session.

C>*mail abc*

The user starts mail in send mode. The mail will be sent to user abc.

Subject: *sample subject*

The user gets prompted for a subject. She types in the subject "sample subject" This prompting can be overriden by the MAIL.RC file

This is the body of a sample message.

After typing the line above the user wants to review the message , so she enters the ~p command:

~p

Message contains:
Subject: sample subject

This is the body of a sample message.
(continue)

We are now back in input mode.

~.

The user is done with the message, and enters the ~. command to end the message.

```
Cc: ut
```

The user wants to receive a copy of this message in her workstation mailbox, and adds her address to the Cc: list of recipients. The Cc: prompting can by controled by the MAIL.RC

```
C>UP
```

The user is now ready to send the mail to the remote workstation. She first has to prepare it for uploading. UP will prepare all queued messages for uploading. Should be executed only once between MAIL and SEND.

```
C>send -PUT -LIST
-- theodora v1.0 --
Default Port Settings Used
```

Now it is time to send the new mail message. At the same time she wants to get a list of the mail that is waiting for her back at her office. The -PUT option tells SEND to upload the mail, and the -LIST to download a listing of the workstation mailbox.If the user had entered one or more communication options the current port settings would have been printed instead of the Default message. The option required most often is -C2. This option tells SEND to use Communication port 2 instead of the default communication port.The user then presses the F2 key. The screen gets cleared and the number being called will appear. This is the number we entered earlier with the DOS SET command

```
ATDT8,4575120
CONNECT 2400
```

If the modem was not successful in connecting to the remote modem, Theodora will print a message that describes the problem better than the message supplied by the modem.

```
#CALL 1110
```

The pound sign is the localnet/20 prompt. 1110 is the call number for the CS department annex. If there is a problem connecting to the annex brunet will print a message explaining why it can not connect

```
Annex Command Line Interpreter * Copyright 1988 Encore Computer Corporation

annex0: rlogin igor
```

In this example the user connected to igor. She could have selected any other workstation in the department

```
login: ut
password:
```

The password will not appear when you type it. Then something similar to the lines bellow will appear followed by what usually appears when the user logs in.

```
Last login: Wed Apr 17 11:37:42 from console
SunOS Release 4.1.1 (CSD4_60) #3: Tue Mar 19 13:37:14 EST 1991

You have new mail
```

A regular login message similar to the one above will appear, followed by the user's shell prompt.

```
%
```

The user can now run Theodora. She presses the F1 key and wait for Theodora to transfer the mail:

```
Running. please wait.........
```

A dot will appear for each line transfered

```
QUIT
221 igor closing connection
```

Theodora is now done. The user can either logout, or run any workstation command. In this example she will logout.

```
%logout
Connection closed.
annex0:
```

The connection is closed, and she is back at the annex port. To hangup she presses F4 on the laptop keyboard

```
+++
OK
ATH0
OK
```

The telephone connection is now closed. User presses Alt-F10 and she is back at the DOS prompt:

```
C>list
```

By typing list the user calls up the LIST program. A list of mail in the remote system mailbox will appear. The following is a typical line as displayed by LIST:

```
U scb Mon Jun 10 19:02:25 1991 19/508 "test"
```

She can select this message by clicking the mouse or pressing the ENTER key while the message is highlighted. The procedure is described on page 16. She quits the program by pressing the Escape key. To download those messages she issues the following command:

```
C>send -GET
```

The GET flag tells theodora to download the selected messages. The user then follows the procedure described in the previous page to connect back to the workstation.

```
C>prepare
```

After quitting SEND and before reading mail the user has to uncompress the mail. If she already had a MAIL.IN file mail will append the new mail to it. PREPARE has to be executed for MAIL to access the new mail

```
C>mail
```

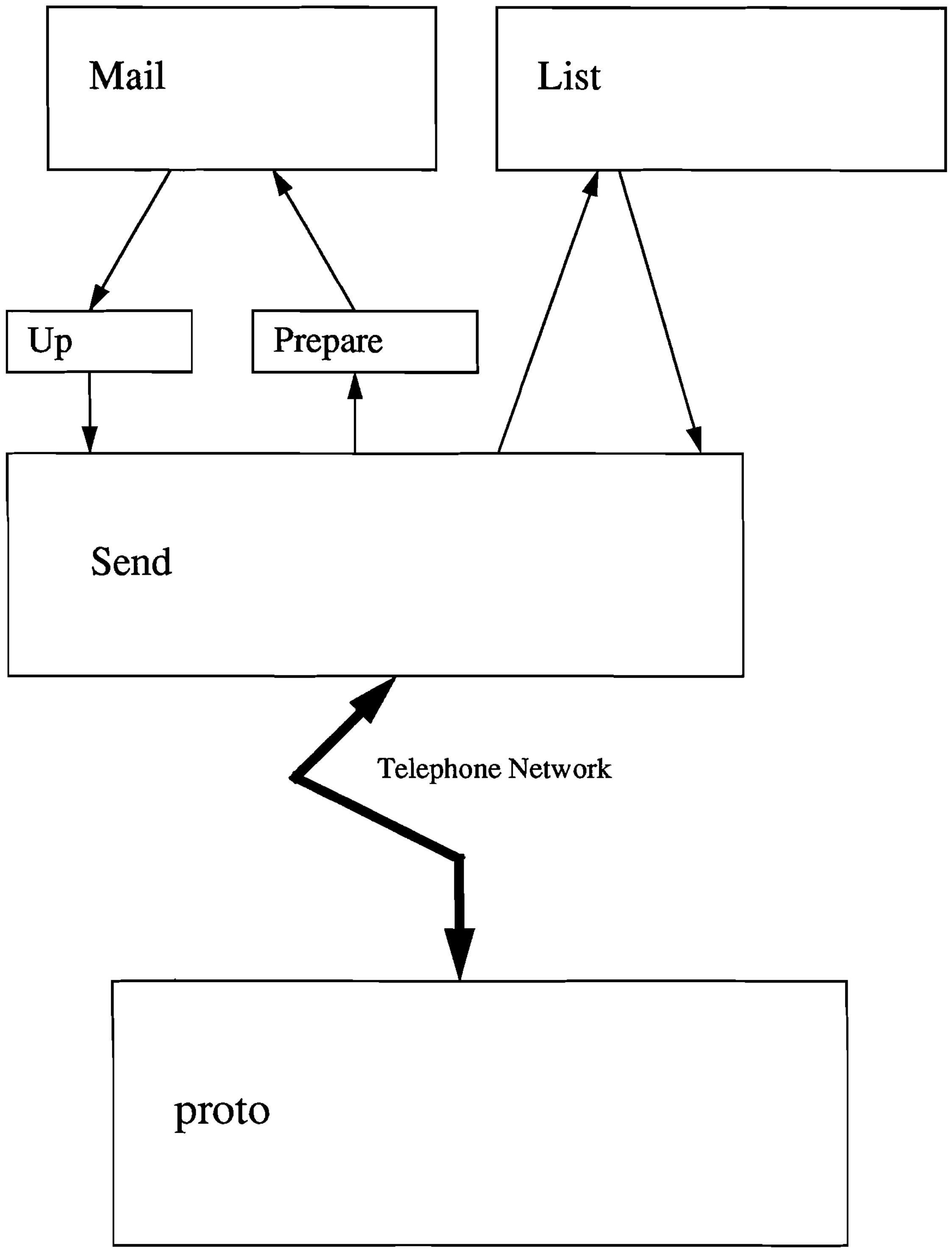
The iuser s now running mail in read mode.

```
Mail version 1.0. Type ? for help.
"mail.in": 2 messages 2 new
&
```

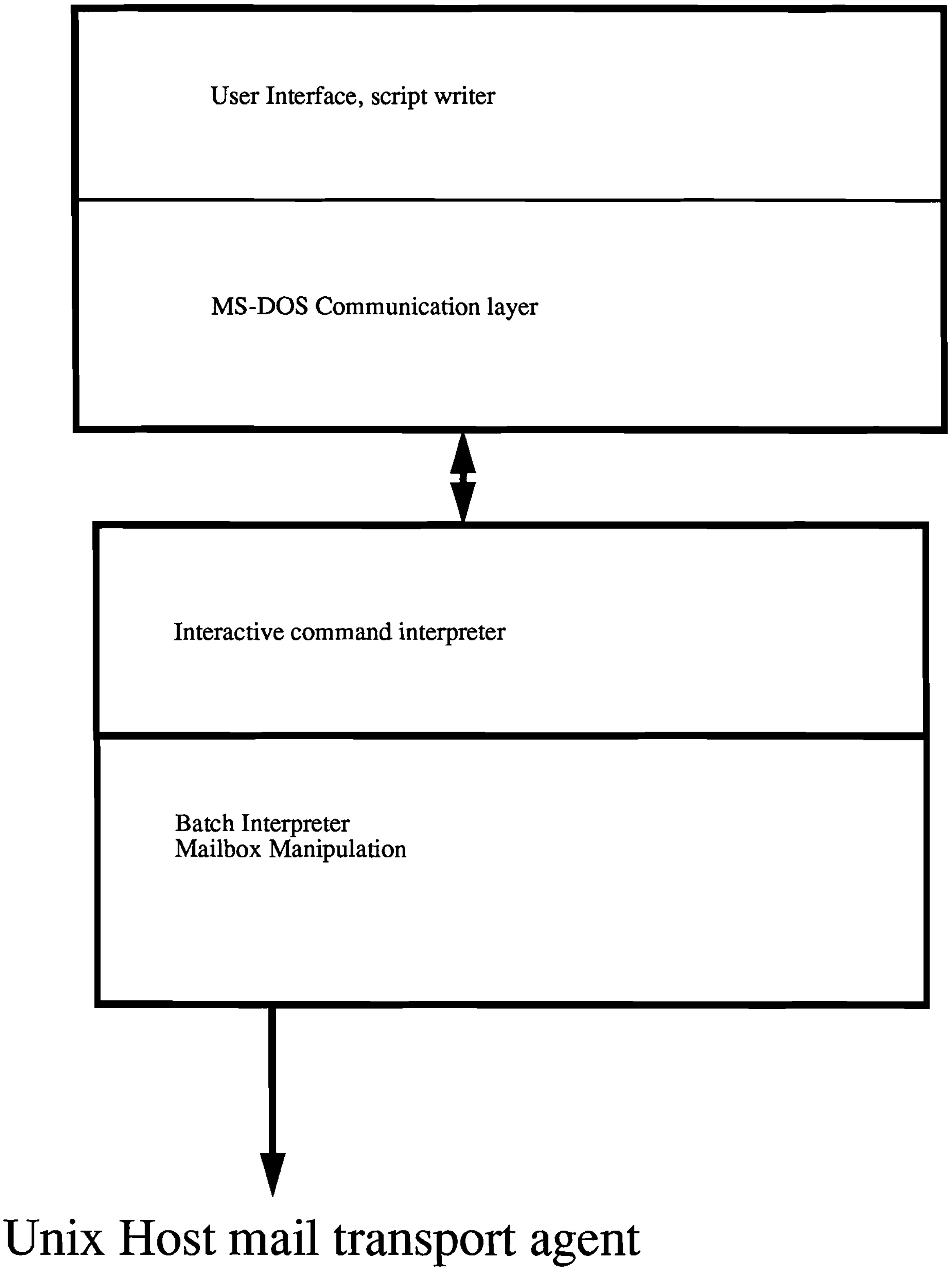
The user can now read mail, or do any of other mailbox manipulations as described on page 7.

Notes:

1. This product includes software developed by the University of California, Berkeley and its contributors



Flow of data between the different programs that make up Theodora


User Interface, script writer
MS-DOS Communication layer
Interactive command interpreter
Batch Interpreter
Mailbox Manipulation
Unix Host mail transport agent

Description of modules in diagram

PC front end (User Interface, script writer): MAIL.EXE

This module has two purposes: The first one is to interactively manipulate the mail for the user in a manner similar to the Unix mail command. The second is to translate the user commands to the Theodora server commands. These server commands are saved into a script which is sent to the server the next time the PC connects to the Unix server. The interface and the commands are the same as the Unix mail program. Commands which are not possible, or do not make sense in the single user environment of MSDOS are not available. The commands which actually manipulate the mailbox are both executed on the local copy of it, and recorded in the script file. This module is also responsible in maintaining a private (PC based) mailbox. This mailbox is similar to the one maintained on the server, but as there is only one instance of it, there is no reason for it to stay in synch with anything on the remote system.

PC front end (Mail selection):

This part of the program displays a list of messages in the remote host system mailbox. The user can click on messages and select them for download. The display includes the following information about a message:

1.Sender, taken from the From field of the message
2.Time and Date of the message, taken from the date field of the message
3.Size in lines and Bytes of the message
4.The subject of the mail, taken from the subject field of the message.
5.The status of the mail, displayed in the same way Unix mail does.

To download them the user has to start the SEND.EXE with the '-get' option.

PC back end (Communication Layer): SEND.EXE, PREPARE.BAT, UP.BAT

This part of the program communicates between the PC and the Unix based server. It compresses the script created by the front end, calls to the server, gets the new mail and sends the script. When needed it uncompress the packets that are sent to it from the server and passes them to the front end. A log of what packets have been sent and successfully received is kept. If the communication is broken before all packets are received, it is able to resume the session at the packet level after it reconnects to the server. The server keeps the information for 24 hours. The user can change the behavior of this part of the program by the use of different flags. The GET flag will get all selected mail for the user. The GETALL flag will get all unread mail for the user. The LIST flag will get a list of all mail in the system mailbox so the user can select (by using the LIST.EXE program) what messages to download. If no options are specified it defaults to GETALL. Changes to the mailbox are automatically sent.

Interactive command interpreter:

Here is where the commands received from the PC are interpreted and executed. These commands are get, put, list, del, mbox, mark, do, save, quit.

Get tells the server to send the user's new (unread) mail. The syntax is **GET**. There is another set of commands to download messages. The user can mark arbitrary messages, and then request them. The syntax is **MARK *message-id***. To get the marked commands the syntax is **DO**. Marks are not saved between sessions. Both versions of get, compress and uudecode their output.

Put tells the server to receive the script from the remote machine. This script contains mail and commands. This script does not get executed until the whole script has been transmitted, uudecoded, and uncompressed; most likely after the connection has been closed. The syntax is **PUT**.

List tell the server to send a list of all queued mail messages. The syntax is **LIST**.

The syntax for delete is **DEL *message-ID***, where message-ID is the message id of the mail message the user wants deleted. The reason we need to specify the message-ID is because it uniquely identifies a mail message. We can not use the message index number as Unix mail does, because it might change between sessions. Note that as the script is written by the program and not the user, saying DEL 1234567890.ABCDEF@host.domain.edu is not as bad as it sounds. Similarly the save command has syntax **SAVE *message-ID filename***. The filename path begins at the user's home directory. The MBOX command is similar to the SAVE command. It saves a copy of the message in the mbox file, located in the user's home directory, and deletes it from the mailbox. The syntax is **MBOX *message-ID***. The message-ids in the above commands have to be enclosed in a pair of '<' and '>' symbols. Also note that SAVE is executed immediately, but MBOX and DEL are executed when a Quit command is received.

Quit tells the server that the client wishes to close the connection, and update the system mailbox. This should always be the last command the client sends to the server. The syntax is **QUIT**. This command can be abbreviated as **Q**.

Batch interpreter:

This is the part of the program that runs when the user is off line. It interprets the script created by the PC front end. The sendmail command sends mail to one or more addresses. This is done by the SEND command. The format is **SEND <address>**. Address is any valid address for the sendmail program of the host on which the server resides. The server does not perform any checking for address validity. These addresses are used for the next **DATA** command. Everything between a DATA and a line consisting of a period "." and an end of line is not parsed by theodora, but it is used to construct the body of the message that is going to be sent. All mail to be sent is passed to the servers host mail agent for delivery. Theodora will report on the commands it executed in batch mode. The report is sent as mail from the server to the user. This way it can either be retrieved through the next invocation of theodora, or through the regular remote host mail commands.

THEODORA

Programming Notes

Workstation side:

The workstation side of the program is made up from eight "C" files and three header files:

main.c: This is where main(), entry point of the program can be found. It also contains init(), the routine that initialize the program, recordmessage to save copies of outgoing mail, and sendmessage() the interface between theodora and sendmail.

file.c: Contains the source to the LIST, DEL, SAVE, MBOX, and MARK commands (all commands that modify the message data structure). It also includes routines to parse and manipulate message headers. List is performed by the listmail() routine in a strait forward manner. Note that although the output is similar to the 'header' mail command, unnecessary spaces have been taken out to save on the transmission. Savemail() executes the SAVE command by looping through the message data structure and appending matching messages to the given file. The other three commands are implemented with the do_mail() command. This command simply sets the appropriate bits to all messages that match the given message id.

mailbox.c: This file contains various message file I/O functions. Most of the routines have been taken from the mail source, and modified for the need of this program. This is the "backend" of the program.

srvr.c: This is where the command interpreter is. It also contains the routines that implement the HELP, SEND, and DATA commands. The interpet() routine performs some preprocessing on the read line, and then gets into a long "switch" statement that implements most of the commands. Note that interpet() is used both in "foreground" and "background" mode. The calling routine is responsible to set the InChannel file pointer to the appropriate file. The difference in behavior in the two modes is implemented by checking the global variable batch.

util.c: This contains various functions taken from Unix source.

uu.c: Contains the encode and decode routine to read and decode, or write and encode the incoming or outgoing messages. It is the modified source of uuencode and uudecode. It also contains the routines that initialize the uuencoding and link it with the error checking routines

exec.c: This module contains the routines that deal with downloading messages, and uploading scripts for execution. These routines correspond to the GET, PUT, and DO commands. A GET is performed by the give() routines. Give() sets the pipes to compress and encode and calls the getunmarkedmail() routine to run through the message structure select the unread messages and pipe them to compress. Give() also runs the DO command. The only difference between the two routines is that the getmarkedmail() routine is the one sending out the data. A PUT is done in two stages. In the first stage receive() collects the data from the input channel, decodes it and uncompresses it using pipes. In the second stage calls execute() to setup the background processing. Execute forks, and the new process calls intepret() again to now process the script.

check.c: Contains the routines that receive (or send) the data from the remote machine. The data is then piped to the uudecode routines.

theodora.h: Contains the definitions for the various data structures and constants used by the program. It includes globals.h and a small number of system include files needed by the various modules. Most of the data structures are adaptations of data structures used by the mail and sendmail programs

Data transfers

When theodora needs to transfer mail between the mailbox computer and the laptop it compresses the mailbox to save on transmission time. It then uuencodes it. She has to make sure that this data is received correctly on the other side, because it is very expensive to resent it. For this she adds the ASCII representation of sum of all the characters in the line in the end of each line. The sum is separated from the data by a Control-L character (ASCII 12). When the other side receives the line, it first checks the sum. If it matches it passes it to uudecode, and sends a received message to the other sending side. If it does not, it requests that line again. A received line starts with the code "200". The resend line starts with the code "550". The text after the 200 or 550 is ignored. If the sending side receives a "550" it will resent the line until it receives a "200". Note that 200, and 550 are the only codes that ever get generated by the PC side.
This method of data transfer should be replaced by a sliding windows protocol. The protocol currently used is very reliable, but the transfers are much slower than before the protocol was added.