IEEE P1003.2a Draft 8 – December 1991

Copyright © 1991 by the
Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017, USA
All rights reserved as an unpublished work.

This is an unapproved and unpublished IEEE Standards Draft, subject to change. The publication, distribution, or copying of this draft, as well as all derivative works based on this draft, is expressly prohibited except as set forth below.

Permission is hereby granted for IEEE Standards Committee participants to reproduce this document for purposes of IEEE standardization activities only, and subject to the restrictions contained herein.

Permission is hereby also granted for member bodies and technical committees of ISO and IEC to reproduce this document for purposes of developing a national position, subject to the restrictions contained herein.

Permission is hereby also granted to the preceding entities to make limited copies of this document in an electronic form only for the stated activities.

The following restrictions apply to reproducing or transmitting the document in any form: 1) all copies or portions thereof must identify the document's IEEE project number and draft number, and must be accompanied by this entire notice in a prominent location; 2) no portion of this document may be redistributed in any modified or abridged form without the prior approval of the IEEE Standards Department.

Other entities seeking permission to reproduce this document, or any portion thereof, for standardization or other activities, must contact the IEEE Standards Department for the appropriate license.

Use of information contained in this unapproved draft is at your own risk.

IEEE Standards Department
Copyright and Permissions
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331, USA
+1 (908) 562-3800
+1 (908) 562-1571 [FAX]

4.40 `mailx` — Process messages

Editor's Note: This clause contains text that is directly extracted from POSIX.2 Draft 11.2 and, as such, is not eligible for balloting by the POSIX.2a group. This text is marked with the '1''or '2''diff-marks.

⇒ **4.40 `mailx`.** *Replace the entire contents of clause 4.40 with:*

4.40.1 Synopsis

Send Mode:

```
mailx [-s subject] address ...
```

On systems supporting the User Portability Utilities Option, the following forms shall also be valid (Receive Mode):

```
mailx -e
mailx [-HiNn][-u user]
mailx -f[-HiNn][file]
```

4.40.2 Description

The `mailx` utility provides a message sending and receiving facility.

When this standard is implemented without the User Portability Utilities Option, an application using `mailx` shall have the ability to send messages directly from standard input to one or more addresses in an unspecified manner. (Send Mode.) Unless the first character of one or more lines is tilde (~), all characters in the input message shall appear in the delivered message, but additional characters may be inserted in the message before it is retrieved.

On systems supporting the User Portability Utilities Option, mail-receiving capabilities and other interactive features, Receive Mode, described below, shall also be enabled.

When reading mail, `mailx` provides commands to facilitate saving, deleting, and responding to messages. When sending mail, `mailx` allows editing, reviewing, and other modification of the message as it is entered.

Incoming mail shall be stored in one or more unspecified locations for each user, collectively called the system *mailbox* for that user. When `mailx` is invoked in Receive Mode, the system mailbox shall be the default place to find them. As messages are read, they shall be marked to be moved to a secondary file for storage, unless specific action is taken. This secondary file is called the *mbox* and is normally located in the **HOME** directory of the user (see **MBOX**, in 4.40.5.3, for a description of this file). Messages shall remain in this file until explicitly

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

removed. When the –f option is used to read mail messages from secondary files, messages shall be retained in those files unless specifically removed. All three of these locations—system mailbox, *mbox*, and secondary file—are referred to in this clause as simply ‘‘mailboxes,’’ unless more specific identification is required.

4.40.3 Options

The `mailx` utility shall conform to the utility argument syntax guidelines described in 2.10.2.

The following options shall be supported by the implementation. Only the –s *subject* option shall be required on all systems. The other options are required only on systems supporting the User Portability Utilities option.

–e Test for the presence of mail in the system mailbox. The `mailx` utility shall write nothing and exit with a successful return code if there is mail to read. 7

–f Read messages from the file named by the *file* operand instead of the system mailbox. (See also `folder`). If no *file* operand is specified, read messages from the *mbox* instead of the system mailbox.

–H Write a header summary only.

–i Ignore interrupts. (See also `ignore`).

–n Do not initialize from the system default startup file. See 4.40.7.

–N Do not write an initial header summary.

–s *subject* Set the Subject header field to *subject*. All characters in the *subject* string shall appear in the delivered message. The results are unspecified if *subject* is longer than {LINE_MAX} – 10 bytes or contains a <newline>. 2

–u *user* Read the system mailbox of the login name *user*. This shall only be successful if the invoking user has the appropriate privileges to read the system mailbox of that user. 7

4.40.4 Operands

The following operands shall be supported by the implementation:

address Addressee of message. When –n is specified and no user startup files are accessed (see 4.40.7), this shall be an address to pass to the mail delivery system. Any system or user startup files may enable aliases (see 4.40.7.2.1), that may modify the form of *address* before it is passed to the mail delivery system. 7

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

file A pathname of a file to be read instead of the system mailbox when `-f` is specified. The meaning of the *file* option-argument shall be affected by the contents of the `folder` internal variable; see 4.40.7.1.

4.40.5 External Influences

4.40.5.1 Standard Input

When `mailx` is invoked in Send Mode (the first synopsis line), standard input shall be the message to be delivered to the specified addresses. If the User Portability Utilities Option is not supported, standard input lines beginning with a tilde (~) character produce unspecified results.

When `mailx` is invoked in Receive Mode, (the second to fourth synopsis lines), standard input can contain escaped commands (by default, lines beginning with tilde), as described in 4.40.7.

4.40.5.2 Input Files

When `mailx` is used as described by this standard, the *file* option-argument (see the `-f` option) and the *mbox* shall be text files containing mail messages, formatted as described in 4.40.6.3. The nature of the system mailbox is unspecified; it need not be a file.

4.40.5.3 Environment Variables

The following environment variables shall affect the execution of `mailx`:

DEAD This variable shall be interpreted as a pathname of the file in which to save partial messages in case of interrupts or delivery errors. The default shall be `dead.letter` in the directory named by the **HOME** variable. The behavior of `mailx` in saving partial messages is unspecified if the User Portability Utilities Option is not supported and **DEAD** is not defined with the value `/dev/null`.

EDITOR This variable shall be interpreted as a name of a utility to invoke when the `edit` (see 4.40.7.2.8) or `~e` (see 4.40.7.3), command is used. The default is unspecified. This standard specifies the effects of this variable only for systems supporting the User Portability Utilities Option.

HOME This variable shall be interpreted as the pathname of the user's home directory.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

LANG This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

LC_ALL This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

LC_CTYPE This variable shall determine the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments and input files) and the handling of case-insensitive address and header-field comparisons.

LC_MESSAGES This variable shall determine the language in which messages should be written.

LC_TIME This variable may determine the format and contents of the date and time strings written by `mailx`. This standard specifies the effects of this variable only for systems supporting the User Portability Utilities Option. 7

LISTER This variable shall be interpreted as a string representing the command for writing the contents of the `folder` directory to standard output when the `folders` command is given (see 4.40.7.2.11). Any string acceptable as a *command_string* operand to the `sh -c` command (see 4.56) shall be valid. If this variable is null or not set, the output command shall be `ls` (see 4.39). The default value shall be unset. This standard specifies the effects of this variable only for systems supporting the User Portability Utilities Option. 7

MAILRC This variable shall be interpreted as a pathname of the startup file. The default shall be `.mailrc` in the **HOME** directory. The behavior of `mailx` is unspecified if the User Portability Utilities Option is not supported and **MAILRC** is not defined with the value `/dev/null`. 7

MBOX This variable shall be interpreted as a pathname of the file to save messages from the system mailbox that have been read. The `exit` command shall override this function, as shall saving the message explicitly in another file. The default shall be `mbox` in the directory named by the **HOME** variable. This standard specifies the effects of this variable only for systems supporting the User Portability Utilities Option. 7

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

PAGER This variable shall be interpreted as a string representing an output filtering and/or pagination command for writing the output to the terminal. Any string acceptable as a *command_string* operand to the `sh -c` command (see 4.56) shall be valid. When standard output is a terminal device, the message output shall be piped through the command if the `mailx` internal variable `crt` is set to a value less the number of lines in the message; see 4.40.7.1. If the **PAGER** variable is null or not set, the paginator shall be either `more` (see 5.18) or another paginator utility documented in the system documentation. This standard specifies the effects of this variable only for systems supporting the User Portability Utilities Option. 7 8 8 7 7

SHELL The name of a preferred command interpreter. The default shall be `sh` (see 4.56). This standard specifies the effects of this variable only for systems supporting the User Portability Utilities Option. 7 7 7

TERM This variable shall be interpreted as the name of the terminal type, to determine in an unspecified manner, if the internal variable `screen` is not specified, the number of lines in a screenful of headers. If **TERM** is not set or is set to null, an unspecified default terminal type shall be used and the value of a screenful is unspecified. This standard specifies the effects of this variable only for systems supporting the User Portability Utilities Option. 7 7 7 7 7 7 7

VISUAL This variable shall be interpreted as a pathname of a utility to invoke when the `visual` command (see 4.40.7.2.38) or `~v` command-escape (see 4.40.7.3), is used. If this variable is null or not set, the full-screen editor shall be `vi` (see 5.35). This standard specifies the effects of this variable only for systems supporting the User Portability Utilities Option. 7 7 7

4.40.5.4 Asynchronous Events

When `mailx` is in Send Mode and standard input is not a terminal, it shall take the standard action for all signals; see 2.11.5.4.

In Receive Mode, or in Send Mode when standard input is a terminal, if a SIGINT signal is received:

(1) If in command mode, the current command, if there is one, shall be aborted, and a command-mode prompt shall be written.

(2) If in input mode:

(a) If `ignore` is set, the interrupt shall be ignored, and processing continues.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

(b) If the interrupt was received while sending mail, either when in Receive Mode or in Send Mode, a message shall be written, and another subsequent interrupt, with no other intervening characters typed, shall be required to abort the mail message. If in Receive Mode and another interrupt is received, a command-mode prompt shall be written. If in Send Mode and another interrupt is received, `mailx` shall terminate with a nonzero status.

In both cases listed in (b), if the message is not empty:

[1] If `save` is enabled and the file named by **DEAD** can be created, the message shall be written to the file named by **DEAD** . If the file exists, the message shall be written to replace the contents of the file.

[2] If `save` is not enabled, or the file named by **DEAD** cannot be created, the message shall not be saved.

The `mailx` utility shall take the standard action for all other signals; see 2.11.5.4.

4.40.6 External Effects

4.40.6.1 Standard Output

In command and input modes, all output, including prompts and messages, shall be written to standard output.

4.40.6.2 Standard Error

Used only for diagnostic messages.

4.40.6.3 Output Files

Various `mailx` commands and command escapes can create or add to files, including the *mbox* , the dead-letter file, and secondary mailboxes. When `mailx` is used as described in this standard, these files shall be text files, formatted as follows:

line beginning with `"From "`
[*header-line(s); see 4.40.7.2*]
empty line
[*body line(s)*
empty line]
[*line beginning with* `"From "...`]

where each message begins with the `"From "` line shown, preceded by the beginning of the file or an empty line. (The `"From "` line is considered to be part of the message header, but not one of the ‘‘header-lines’’ referred to in 4.40.7.2; thus, it is not affected by the `discard`, `ignore`, or `retain` commands). The formats of the remainder of the `"From "` line and any additional header lines are unspecified, except that none shall be empty. The format of a message body line

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

is also unspecified, except that none shall start with "From "; mailx shall modify any user-entered message body lines beginning with "From " by adding one or more characters to precede the F.

When a message from the system mailbox or entered by the user is not a text file, it is implementation defined how such a message is stored in files written by mailx.

4.40.7 Extended Description

The entire Extended Description subclause (4.40.7 through 4.40.7.3) shall apply only to implementations supporting the User Portability Utilities option.

When mailx is invoked using one of the Receive Mode synopsis forms, it shall write the page of header-summary lines (see below) containing the first new message (if −N is not specified), or the first unread message if there are no new messages, or the first message if there are no new or unread messages, followed by a prompt indicating mailx can accept regular commands (see 4.40.7.2); this is termed *command mode*. When mailx is invoked using the Send Mode synopsis and standard input is a terminal, if no subject is specified on the command line and the asksub variable is set, a prompt for the subject shall be written. At this point mailx is in *input mode*. This input mode is also entered when using one of the Receive Mode synopsis forms and a reply or new message is composed using the reply, Reply, or mail commands. When the message is typed and the end of message is encountered, the message shall be passed to the mail delivery software. Commands can be entered by beginning a line with the escape character [by default, tilde (~)] followed by a single command letter and optional arguments. See 4.40.7.3 for a summary of these commands.

NOTE: For notational convenience, this clause uses the default escape character, tilde, in all references and examples.

At any time, the behavior of mailx shall be governed by a set of environmental and internal variables. These are flags and valued parameters that can be set and cleared via the set and unset commands. See 4.40.5.3 and 4.40.7.1 for summaries of these parameters.

Regular commands are of the form:

[*command* **] [***msglist* **] [***argument* ... **]**

If no *command* is specified in command mode, print shall be assumed. In input mode, commands shall be recognized by the escape character, and lines not treated as commands shall be taken as input for the message.

In command mode, each message shall be assigned a sequential number, starting with 1.

All messages have a state that affects how they are displayed in the header summary and how they are retained or deleted upon termination of mailx. There is at any time the notion of a *current* message, marked by a > at the beginning of a line in the header summary. All messages are in one of the following states:

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

new The message is present in the system mailbox and has not been viewed by the user or moved to any other state. Messages in state *new* when `mailx` quits shall be retained in the system mailbox.

unread The message has been present in the system mailbox for more than one invocation of `mailx` and has not been viewed by the user or moved to any other state. Messages in state *unread* when `mailx` quits shall be retained in the system mailbox.

read The message has been processed by one of the following commands: `~f`, `~m`, `~F`, `~M`, `copy`, `mbox`, `next`, `pipe`, `print`, `Print`, `top`, `type`, `Type`, `undelete`. The `delete`, `dp`, and `dt` commands may also cause the next message to be marked as *read*, depending on the value of the `autoprint` variable. Messages that are in the system mailbox and in state *read* when `mailx` quits shall be saved in the *mbox*, unless the internal variable `hold` was set. 7
Messages that are in the *mbox* or in a secondary mailbox and in 7
state *read* when `mailx` quits shall be retained in their current location.

deleted The message has been processed by one of the following commands: `delete`, `dp`, `dt`. A message processed by `save` shall be 7
in state *deleted* unless the internal variable `keepsave` was set. 7
Messages in state *deleted* when `mailx` quits shall be deleted.

preserved The message has been processed by a `preserve` command. When `mailx` quits, the message shall be retained in its current location.

The header-summary line for each message shall indicate the state of the message.

Many commands take an optional list of messages (*msglist*) on which to operate, which defaults to the current message. A *msglist* is a list of message specifications separated by `<blank>`s, which can include:

n Message number *n*.

`+` The next undeleted message, or the next deleted message for the 7
`undelete` command. 7

`-` The next previous undeleted message, or the next previous 7
deleted message for the `undelete` command. 7

`.` The current message.

`^` The first undeleted message, or the first deleted message for the `undelete` command.

`$` The last message.

`*` All messages.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

n−*m*	An inclusive range of message numbers.
address	All messages from *address* ; any address as shown in a header summary shall be matchable in this form.
/*string*	All messages with *string* in the subject line (case ignored).
:*c*	All messages of type *c*, where *c* shall be one of:
	`d` Deleted messages
	`n` New messages
	`o` Old messages (any not in state *read* or *new*).
	`r` Read messages
	`u` Unread messages

Other commands take an optional message (*message*) on which to operate, which defaults to the current message. All of the forms allowed for *msglist* are also allowed for *message* , but if more than one message is specified, only the first shall be operated on.

Other arguments are usually arbitrary strings whose usage depends on the command involved.

At startup time, `mailx` shall take the following steps in sequence:

(1) Establish all variables at their stated default values.

(2) Process command-line options, overriding corresponding default values.

(3) Import any of the **DEAD** , **EDITOR** , **MBOX** , **LISTER** , **PAGER** , **SHELL** , or **VISUAL** variables that are present in the environment, overriding the corresponding default values.

(4) Read `mailx` commands from an unspecified system startup file, unless the `-n` option is given, to initialize any internal `mailx` variables and aliases.

(5) Process the startup file of `mailx` commands named in the user **MAILRC** variable.

Most regular `mailx` commands are valid inside startup files, the most common use being to set up initial display options and alias lists. The following commands shall be invalid in the startup file: `!`, `edit`, `hold`, `mail`, `preserve`, `reply`, `Reply`, `shell`, and `visual`. Any errors in the startup file shall either cause `mailx` to terminate with a diagnostic message and a nonzero status, or to continue after writing a diagnostic message, ignoring the remainder of the lines in the startup file.

A blank line in a startup file shall be ignored.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.1 `mailx` Internal Variables

The following variables are internal `mailx` variables. Each internal variable can be set via the `mailx set` command at any time. The `unset` and `set no`*name* commands can be used to erase variables.

In the following list, variables shown as

`variable`

represent Boolean values. Variables shown as

`variable=`*value*

shall be assigned string or numeric values. For string values, the rules in 4.40.7.2 concerning file names and quoting also apply.

The defaults specified here may be changed by the implementation-specific system startup file unless the user specifies the `-n` option.

`append` Append messages to the end of the *mbox* file upon termination instead of placing them at the beginning. The default shall be `noappend`. This variable shall not affect the `save` command when saving to the *mbox* .

`ask`

`asksub` Prompt for a subject line on outgoing mail if one is not specified on the command line with the `-s` option. The `ask` and `asksub` forms are synonyms; the system shall refer to `asksub` and `noasksub` in its messages, but shall accept `ask` and `noask` as user input to mean `asksub` and `noasksub`. It shall not be possible to set both `ask` and `noasksub`, or vice versa. The default shall be `asksub`, but no prompting shall be done if standard input is not a terminal.

`askbcc` Prompt for the Bcc list. The default shall be `noaskbcc`.

`askcc` Prompt for the Cc list. The default shall be `noaskcc`.

`autoprint` Enable automatic writing of messages after `delete` and `undelete` commands. The default shall be `noautoprint`.

`bang` Enable the special-case treatment of exclamation-marks (`!`) in escape command lines; see 4.40.7.2.41 and 4.40.7.3. The default shall be `nobang`, disabling the expansion of `!` in the *command* argument to the `~!` command and the `~<!`*command* escape.

`cmd=`*command*
Set the default command to be invoked by the `pipe` command; see 4.40.7.2.21. The default shall be `nocmd`.

`crt=`*number*
Pipe messages having more than *number* lines through the command specified by the value of the **PAGER** variable. The default shall be `nocrt`. If it is set to null, the value used is implementation defined.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

`dot` When `dot` is set, a period on a line by itself during message input from a terminal shall also signify end-of-file (in addition to normal end-of-file). The default shall be `nodot`. If `ignoreeof` is set (see below), a setting of `nodot` shall be ignored and the period shall be the only method to terminate input mode.

`escape=`*c* Set the command escape character to be the character *c*. By default, the command escape character shall be tilde. If `escape` is unset, tilde shall be used; if it is set to null, command escaping shall be disabled.

`flipr` Reverse the meanings of the `R` and `r` commands; see 4.40.7.2.25 and 4.40.7.2.26. The default shall be `noflipr`.

`folder=`*directory*
The default directory for saving mail files. User-specified filenames beginning with a plus-sign (+) shall be expanded by preceding the filename with this directory name to obtain the real pathname. If *directory* does not start with a slash (/), the contents of **HOME** shall be prefixed to it. The default shall be `nofolder`. If `folder` is unset or set to null, user-specified filenames beginning with "`+`" shall refer to files in the current directory that begin with the literal `+` character. See also `outfolder` below. The `folder` value need not affect the processing of the files named in **MBOX** and **DEAD** .

`header` Enable writing of the header summary when entering `mailx` in Receive Mode. The default shall be `header`.

`hold` Preserve all messages that are read in the system mailbox instead of putting them in the *mbox* save file. The default shall be `nohold`.

`ignore` Ignore interrupts while entering messages. The default shall be `noignore`.

`ignoreeof` Ignore normal end-of-file during message input. Input can be terminated only by entering a period (`.`) on a line by itself or by the `~.` command escape. The default shall be `noignoreeof`. See also `dot` above.

`indentprefix=`*string*
A string that shall be prefixed to each line that is inserted into the message by the `~m` command escape. This variable shall default to one `<tab>` character.

`keep` When a system mailbox, secondary mailbox, or *mbox* is empty, truncate it to zero length instead of removing it. The default shall be `nokeep`.

`keepsave` Keep messages that have been saved in other files in the system mailbox instead of deleting them. The default shall be `nokeepsave`.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

`metoo` Suppress the deletion of the login name of the user from the recipient list when replying to a message or sending to a group. The default shall be `nometoo`.

`outfolder` Cause the files used to record outgoing messages to be located in the directory specified by the `folder` variable unless the pathname is absolute. The default shall be `nooutfolder`. See the `record` variable.

`page` Insert a `<form-feed>` after each message sent through the pipe created by the `pipe` command. The default shall be `nopage`.

`prompt=`*string*
Set the command-mode prompt to *string* . If *string* is null or if `noprompt` is set, no prompting shall occur. The default shall be to prompt with the string `"? "`.

`quiet` Refrain from writing the opening message and version when entering `mailx`. The default shall be `noquiet`.

`record=`*file*
Record all outgoing mail in the file with the pathname *file* . The default shall be `norecord`. See also `outfolder` above.

`save` Enable saving of messages in the dead-letter file on interrupt or delivery error. See the variable **DEAD** for the location of the dead-letter file. The default shall be `save`.

`screen=`*number*
Set the number of lines in a screenful of headers for the `headers` and `z` commands. If `screen` is not specified, a value based on the terminal type identified by the **TERM** environment variable, the window size, and/or the baud rate shall be used.

`showto` When the sender of the message was the user who is invoking `mailx`, write the information from the `"To:"` line instead of the `"From:"` line in the header summary. The default shall be `noshowto`.

`sign=`*string*
Set the variable inserted into the text of a message when the `~a` command escape is given. The default shall be `nosign`. The character sequences `\t` and `\n` shall be recognized in the variable as `<tab>` and `<newline>`, respectively. (See also `~i` in 4.40.7.3).

`Sign=`*string*
Set the variable inserted into the text of a message when the `~A` command escape is given. The default shall be `noSign`. The character sequences `\t` and `\n` shall be recognized in the variable as `<tab>` and `<newline>`, respectively. (See also `~i` in 4.40.7.3).

`toplines=`*number*
Set the number of lines of the message to write with the `top` command. The default shall be 5.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2 `mailx` Commands

The following `mailx` commands shall be implemented. In the following list, ''header'' refers to lines from the message header, as shown in 4.40.6.3. ''Header-line'' refers to lines within the header that begin with one or more non-white-space characters, immediately followed by a colon and white space and continuing until the next line beginning with a non-white-space character or an empty line. ''Header-field'' refers to the portion of a header line prior to the first colon in that line.

For each of the commands listed below, the command can be entered as the abbreviation (those characters in the Synopsis command word preceding the **[**), the full command (all characters shown for the command word, omitting the **[** and **]**), or any truncation of the full command down to the abbreviation. For example, the `exit` command (shown as `ex[it]` in the Synopsis) can be entered as `ex`, `exi`, or `exit`.

The arguments to commands can be quoted, using the following methods:

— An argument can be enclosed between paired double-quotes (`" "`) or single-quotes (`' '`); any white space, shell word expansion, or backslash characters within the quotes shall be treated literally as part of the argument. A double-quote shall be treated literally within single-quotes and vice versa. These special properties of the quote marks shall occur only when they are paired at the beginning and end of the argument.

— A backslash outside of the enclosing quotes shall be discarded and the following character treated literally as part of the argument.

— An unquoted backslash at the end of a command line shall be discarded and the next line shall continue the command.

File names, where expected, shall be subjected to the process of shell word expansions (see 3.6); if more than a single pathname results and the command is expecting one file, the effects are unspecified. If the file name begins with an unquoted plus-sign, it shall not be expanded, but treated as the named file (less the leading plus) in the `folder` directory. (See the `folder` variable.)

4.40.7.2.1 Declare aliases

Synopsis : `a[lias]` [*alias* [*address* ...]]
Synopsis : `g[roup]` [*alias* [*address* ...]]

Add the given addresses to the alias specified by *alias* . The names shall be substituted when *alias* is used as a recipient address specified by the user in an outgoing message (i.e., other recipients addressed indirectly through the `reply` command shall not be substituted in this manner). Mail address alias substitution shall apply only when the alias string is used as a full address; e.g., when `hlj` is an alias, `hlj@posix.com` does not trigger the alias substitution. If no arguments are given, write a listing of the current aliases to standard output. If only an *alias* argument is given, write a listing of the specified alias to standard output. These listings need not reflect the same order of addresses that were entered.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2.2 Declare alternates

Synopsis : `alt[ernates]` *name ...*

Declare a list of alternate names for the user's login. When responding to a message, these names shall be removed from the list of recipients for the response. The comparison of names shall be in a case-insensitive manner. With no arguments, `alternates` shall write the current list of alternate names.

4.40.7.2.3 Change current directory

Synopsis : `cd` [*directory*]
Synopsis : `ch[dir]` [*directory*]

Change directory. If *directory* is not specified, the contents of **HOME** shall be used.

4.40.7.2.4 Copy messages

Synopsis : `c[opy]` [*file*]
Synopsis : `c[opy]` [*msglist*] *file*

Copy messages to the file named by the pathname *file* without marking the messages as saved. Otherwise, it shall be equivalent to the `save` command.

4.40.7.2.5 Delete messages

Synopsis : `d[elete]` [*msglist*]

Mark messages for deletion from the mailbox. The deletions shall not occur until `mailx` quits (see 4.40.7.2.24) or changes mailboxes (see 4.40.7.2.10). If `autoprint` is set, the next message after the last one deleted shall be written (see 4.40.7.1); if there is no subsequent message, the previous message, if it exists, shall be written. In the case of no subsequent or previous message, or when `noautoprint` is set, the `mailx` prompt shall be written.

4.40.7.2.6 Discard header fields

Synopsis : `di[scard]` [*header-field ...*]
Synopsis : `ig[nore]` [*header-field ...*]

Suppress the specified header fields when writing messages. Specified *header-fields* shall be added to the list of suppressed header fields. Examples of header fields to ignore are `status` and `cc`. The fields shall be included when the message is saved. The `Print` and `Type` commands shall override this command. The comparison of header fields shall be in a case-insensitive manner. If no arguments are specified, write a list of the currently suppressed header fields to standard output; the listing need not reflect the same order of header fields that were entered.

If both `retain` (see 4.40.7.2.27) and `discard` commands are given, `discard` commands shall be ignored.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2.7 Delete messages and display

Synopsis : `dp` [*msglist*]
Synopsis : `dt` [*msglist*]

Delete the specified messages from the mailbox and write the next message after the last one deleted. If there is no subsequent message, the `mailx` prompt shall be written.

4.40.7.2.8 Edit messages

Synopsis : `e[dit]` [*msglist*]

Edit the given messages. The messages shall be placed in a temporary file and the utility named by the **EDITOR** variable (see 4.40.5.3) shall be invoked to edit the file. The default editor is unspecified.

4.40.7.2.9 Exit

Synopsis : `ex[it]`
Synopsis : `x[it]`

Exit from `mailx` without changing the mailbox. No messages shall be saved in the *mbox* (see also `quit`).

4.40.7.2.10 Change folder

Synopsis : `fi[le]` [*file*]
Synopsis : `fold[er]` [*file*]

Quit (see 4.40.7.2.24) from the current file of messages and read in the file named by the pathname *file* . If no argument is given, the name and status of the current mailbox shall be written.

Several unquoted special characters shall be recognized when used as *file* names, with the following substitutions:

%	The system mailbox for the invoking user.
%*user*	The system mailbox for *user* .
#	The previous file.
&	The current *mbox* .
+*file*	The named file in the `folder` directory. (See the `folder` variable.)

The default file shall be the current mailbox.

4.40.7.2.11 Display list of folders

Synopsis : `folders`

Write the names of the files in the directory set by the `folder` variable (see 4.40.5.3).

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2.12 Display header summary for specified messages

Synopsis : `f[rom]` [*msglist*]

Write the header summary for the specified messages.

4.40.7.2.13 Display header summary

Synopsis : `h[eaders]` [*message*]

Write the page of headers that includes the message specified. The `screen` variable sets the number of headers per page. See also the `z` command.

4.40.7.2.14 Help

Synopsis : `hel[p]`
Synopsis : `?`

Write a summary of commands.

4.40.7.2.15 Hold messages

Synopsis : `ho[ld]` [*msglist*]
Synopsis : `pre[serve]` [*msglist*]

Mark the messages in *msglist* to be retained in the mailbox when `mailx` terminates. This shall override any commands that might previously have marked the messages to be deleted. During the current invocation of `mailx`, only the `delete`, `dp`, or `dt` commands shall remove the *preserve* marking of a message.

4.40.7.2.16 Execute commands conditionally

Synopsis : `i[f] s|r`
mail-command s
`el[se]`
mail-command s
`en[dif]`

Execute commands conditionally, where `if s` shall execute the following *mail-command* s, up to an `else` or `endif`, if the program is in Send Mode, and `if r` shall cause the *mail-command* s to be executed only in Receive Mode.

4.40.7.2.17 List available commands

Synopsis : `l[ist]`

Write a list of all commands available. No explanation shall be given.

4.40.7.2.18 Mail a message

Synopsis : `m[ail]` *address* ...

Mail a message to the specified addresses or aliases.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2.19 Direct messages to mbox

Synopsis : `mb[ox]` [*msglist*]

Arrange for the given messages to end up in the *mbox* save file when `mailx` terminates normally. (See **MBOX** in 4.40.5.3 for a description of this file.) See also the `exit` and `quit` commands.

4.40.7.2.20 Process next specified message

Synopsis : `n[ext]` [*message*]

Go to the next message matching *message* .

4.40.7.2.21 Pipe message

Synopsis : `pi[pe]` [[*msglist*] *command*]
Synopsis : `|` [[*msglist*] *command*]

Pipe the messages through the given *command* by invoking the command interpreter specified by **SHELL** (see 4.40.5.3) with two arguments: `−c` and *command* . (See also `sh −c` in 4.56.) The command shall be given as a single argument. Quoting, described previously, can be used to accomplish this. If no arguments are given, the current message shall be piped through the command specified by the value of the `cmd` variable. If the `page` variable is set, a `<form-feed>` character shall be inserted after each message.

4.40.7.2.22 Display message with headers

Synopsis : `P[rint]` [*msglist*]
Synopsis : `T[ype]` [*msglist*]

Write the specified messages, including all header lines, to standard output. Override suppression of lines by the `discard`, `ignore`, and `retain` commands. If `crt` is set, the messages longer than the number of lines specified by the `crt` variable shall be paged through the command specified by the **PAGER** environment variable. (See 4.40.5.3).

4.40.7.2.23 Display message

Synopsis : `p[rint]` [*msglist*]
Synopsis : `t[ype]` [*msglist*]

Write the specified messages to standard output. If `crt` is set, the messages longer than the number of lines specified by the `crt` variable shall be paged through the command specified by the **PAGER** environment variable. (See 4.40.5.3).

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2.24 Quit

Synopsis : `q[uit]`
Synopsis : *end-of-file*

Terminate `mailx`, storing messages that were read in *mbox* (if the current mailbox is the system mailbox and unless `hold` is set), deleting messages that have been explicitly saved (unless `keepsave` is set), discarding messages that have been deleted, and saving all remaining messages in the mailbox.

4.40.7.2.25 Reply to a message list

Synopsis : `R[eply]` [*msglist*]
Synopsis : `R[espond]` [*msglist*]

Mail a reply message to the sender of each message in the *msglist* . The subject line shall be formed by concatenating `"Re: "` (unless it already begins with that string) and the subject from the first message. If `record` is set to a file name, the response shall be saved at the end of that file. 7

See also the `flipr` variable in 4.40.7.1. 7

4.40.7.2.26 Reply to a message

Synopsis : `r[eply]` [*message*]
Synopsis : `r[espond]` [*message*]

Mail a reply message to all recipients included in the header of the message. The subject line shall be formed by concatenating `"Re: "` (unless it already begins with that string) and the subject from the message. If `record` is set to a file name, the response shall be saved at the end of that file. 7

See also the `flipr` variable in 4.40.7.1. 7

4.40.7.2.27 Retain header fields

Synopsis : `ret[ain]` [*header-field* ...]

Retain the specified header fields when writing messages. This command shall override all `discard` and `ignore` commands. The comparison of header fields shall be in a case-insensitive manner. 7 7

4.40.7.2.28 Save messages

Synopsis : `s[ave]` [*file*]
Synopsis : `s[ave]` [*msglist*] *file*

Save the specified messages in the file named by the pathname *file* , or the *mbox* if the *file* argument is omitted. The file shall be created if it does not exist; otherwise, the messages shall be appended to the file. The message shall be deleted from the mailbox when `mailx` terminates unless `keepsave` is set.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2.29 Set variables

Synopsis : `se[t]` [*name* [=[*string*]] ...] [*name* =*number* ...] [`no`*name* ...]

Define one or more variables called *name* . The variable can be given a null, string, or numeric value. Quoting and backslash escapes can occur anywhere in *string* , as described previously, as if the *string* portion of the argument were the entire argument. The forms *name* and *name*= shall be equivalent to *name* `=""` for variables that take string values. The `set` command without arguments shall write a list of all defined variables and their values. The `no`*name* form shall be equivalent to '`unset` *name* ''.

4.40.7.2.30 Invoke a shell

Synopsis : `sh[ell]`

Invoke an interactive command interpreter (see also **SHELL** in 4.40.5.3).

4.40.7.2.31 Display message size

Synopsis : `si[ze]` [*msglist*]

Write the size in bytes of each of the specified messages.

4.40.7.2.32 Read `mailx` commands from a file

Synopsis : `so[urce]` *file*

Read and execute commands from the file named by the pathname *file* and return to command mode.

4.40.7.2.33 Display beginning of messages

Synopsis : `to[p]` [*msglist*]

Write the top few lines of each of the specified messages. If the `toplines` variable is set, it is taken as the number of lines to write. The default shall be 5.

4.40.7.2.34 Touch messages

Synopsis : `tou[ch]` [*msglist*]

Touch the specified messages. If any message in *msglist* is not specifically deleted nor saved in a file, it shall be placed in the *mbox* upon normal termination. See `exit` and `quit`.

4.40.7.2.35 Delete aliases

Synopsis : `una[lias]` [*alias*] ...

Delete the specified alias names. If a specified alias does not exist, the results are unspecified.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2.36 Undelete messages

Synopsis : `u[ndelete]` [*msglist*]

Remove the deleted markings from the specified messages. If `autoprint` is set, the last message of those restored shall be written. If *msglist* is not specified, it shall default to the first deleted message following the current message that has not been undeleted if there is one, or the last deleted message preceding the current message that has not been undeleted otherwise.

4.40.7.2.37 Unset variables

Synopsis : `uns[et]` *name* ...

Cause the specified variables to be erased.

4.40.7.2.38 Edit message with full-screen editor

Synopsis : `v[isual]` [*msglist*]

Edit the given messages with a screen editor. The messages shall be placed in a temporary file, and the utility named by the **VISUAL** variable (see 4.40.5.3) shall be invoked to edit the file. The default editor shall be `vi`.

4.40.7.2.39 Write messages to a file

Synopsis : `w[rite]` [*msglist*] *file*

Write the given messages to the file specified by the pathname *file*, minus the message header. Otherwise, it shall be equivalent to the `save` command.

4.40.7.2.40 Scroll header display

Synopsis : `z[+|–]`

Scroll the header display forward (if `+` is specified or if no option is specified) or backward (if `-` is specified) one screenful. The number of headers written shall be set by the `screen` variable.

4.40.7.2.41 Invoke shell command

Synopsis : `!`*command*

Invoke the command interpreter specified by **SHELL** (see 4.40.5.3) with two arguments: `–c` and *command* . (See also `sh –c` in 4.56.) If the `bang` variable is set, each unescaped occurrence of `!` in command shall be replaced with the command executed by the previous `!` command or `~!` command escape.

4.40.7.2.42 Null command

Synopsis : `#` *comment*

This null command (comment) shall be ignored by `mailx`.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.7.2.43 Display current message number

Synopsis : `=`

Write the current message number.

4.40.7.3 `mailx` Command Escapes

The following commands can be entered only from input mode, by beginning a line with the escape character [by default, tilde (~)]. See the `escape` variable description for changing this special character. The format for the commands is:

<*ESC* ><*command-char*><*separator*> [<*argument(s)*>]

where the <*separator*> can be zero or more <`blank`>s.

In the following descriptions, the argument *command* (but not *mailx-command)* shall be a shell command string. Any string acceptable to the command interpreter specified by the **SHELL** variable (see 4.40.5.3) when it is invoked as `-c` *command_string* (see 4.56 for a description of `sh -c`) shall be valid. The command can be presented as multiple arguments (i.e., quoting is not required).

Command escapes that are listed with *msglist* or *mailx-command* arguments are invalid in Send Mode and produce unspecified results.

`~!` *command* — Invoke the command interpreter specified by **SHELL** (see 4.40.5.3) with two arguments: `-c` and *command* ; and then return to input mode. If the `bang` variable is set, each unescaped occurrence of `!` in *command* shall be replaced with the command executed by the previous `!` command or `~!` command escape.

`~.` — Simulate end-of-file (terminate message input).

`~:` *mailx-command*
`~_` *mailx-command*
— Perform the command-level request.

`~?` — Write a summary of command escapes.

`~A` — This shall be equivalent to `~i Sign`.

`~a` — This shall be equivalent to `~i sign`.

`~b` *name* ... — Add the *name* s to the blind carbon copy (Bcc) list.

`~c` *name* ... — Add the *name* s to the carbon copy (Cc) list.

`~d` — Read in the dead-letter file. See **DEAD** (4.40.5.3) for a description of this file.

`~e` — Invoke the editor, as specified by the **EDITOR** environment variable (see 4.40.5.3), on the partial message.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

~f [*msglist*] Forward the specified messages. The specified messages shall be inserted into the current message without alteration. This command escape shall also insert message headers into the message with field selection affected by the `discard`, `ignore`, and `retain` commands.

~F [*msglist*] This shall be the equivalent of the `~f` command escape, except that all headers shall be included in the message, regardless of previous `discard`, `ignore`, and `retain` commands.

~h If standard input is a terminal, prompt for a Subject line and the To, Cc, and Bcc lists. Other implementation-defined headers may also be presented for editing. If the field is written with an initial value, it can be edited as if it had just been typed.

~i *string* Insert the value of the named variable, followed by a `<newline>`, into the text of the message. If the string is unset or null, the message shall not be changed. 7

~m [*msglist*] Insert the specified messages into the message, prefixing nonempty lines with the string in the `indentprefix` variable. This command escape shall also insert message headers into the message, with field selection affected by the `discard`, `ignore`, and `retain` commands.

~M [*msglist*] This shall be the equivalent of the `~m` command escape, except that all headers shall be included in the message, regardless of previous `discard`, `ignore`, and `retain` commands.

~p Write the message being entered. If the message is longer than `crt` lines (see 4.40.7.1), the output shall be paginated as described for the **PAGER** variable (see 4.40.5.3). 7 7 7

~q Quit (see 4.40.7.2.24) from input mode by simulating an interrupt; see 4.40.5.4. If the body of the message is not empty, the partial message shall be saved in the dead-letter file. See **DEAD** (4.40.5.3) for a description of this file.

~r *file*
~< *file*
~< !*command* Read in the file specified by the pathname *file*. If the argument begins with an exclamation-mark (`!`), the rest of the string shall be taken as an arbitrary system command; the command interpreter specified by **SHELL** (see 4.40.5.3) shall be invoked with two arguments: `−c` and *command* . (See also `sh −c` in 4.56.) The standard output of *command* shall be inserted into the message. 7 7 7 7 7

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

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

940 `~t` *name* ... Add the given *name* s to the To list.

941 `~v` Invoke the full-screen editor, as specified by the **VISUAL**
942 environment variable (see 4.40.5.3), on the partial message.

943 `~w` *file* Write the partial message, without the header, onto the file
944 named by the pathname *file*. The file shall be created or the
945 message shall be appended to it if the file exists.

946 `~x` Exit as with `~q`, except the message shall not be saved in the
947 dead-letter file.

948 `~|` *command* Pipe the body of the message through the given *command* by 7
949 invoking the command interpreter specified by **SHELL** (see 7
950 4.40.5.3) with two arguments: `-c` and *command* . If the *com-* 7
951 *mand* returns a successful exit status, the standard output 7
952 of the command shall replace the message. Otherwise the
953 message shall remain unchanged. If the *command* fails, an
954 error message giving the exit status shall be written.

955 ### 4.40.8 Exit Status

956 When the `-e` option is specified, the `mailx` utility shall exit with one of the follow-
957 ing values:

958 0 Mail was found.

959 >0 Mail was not found or an error occurred.

960 Otherwise, `mailx` utility shall exit with one of the following values:

961 0 Successful completion.

962 >0 An error occurred.

963 ### 4.40.9 Consequences of Errors

964 When in input mode (Receive Mode) or Send Mode: 7

965 — If an error is encountered processing a command escape (see 4.40.7.3), a 7
966 diagnostic message shall be written to standard error, and the message 7
967 being composed may be modified, but this condition shall not prevent the 7
968 message from being sent. 7

969 — Other errors shall prevent the sending of the message. 7

970 When in command mode:

971 — Default.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

4.40.10 Rationale. *(This subclause is not a part of P1003.2a)*

Usage, Examples

The format of the system mailbox is intentionally unspecified. Not all systems will implement system mailboxes as flat files, particularly with the advent of multimedia mail messages. Some system mailboxes may be multiple files, others records in a database. The internal format of the messages themselves are specified with the historical format from Version 7, but only after they have been saved in some file other than the system mailbox. This was done so that many historical applications expecting text-file mailboxes will not be broken.

Some new formats for messages can be expected in the future, probably including binary data, bit maps, and various multimedia objects. As described here, `mailx` is not prohibited from handling such messages, but it must store them as text files in secondary mailboxes (unless some extension, such as a variable or command-line option, is used to change the stored format). Its method of doing so is implementation defined and might include translating the data into text-file-compatible or readable form or omitting certain portions of the message from the stored output.

Some historical implementations support the following internal variable:

`debug` Enable verbose diagnostics for debugging. Messages shall not be delivered. The default shall be `nodebug`.

The `edit` and `visual` commands merely edit the specified messages in a temporary file. They do not modify the contents of those messages in the mailbox; such a capability could be added as an extension, such as by using different command names.

The `discard/ignore` and `retain` commands are not inverses of each other. The `retain` command discards all header-fields except those explicitly retained. The `discard` command keeps all header-fields except those explicitly discarded. After a `retain` command is specified, `discard` and `ignore` commands are ignored.

The restriction on a subject line being {LINE_MAX} – 10 bytes is based on the historical format that consumes 10 bytes for `"Subject: "` and the trailing `<newline>`. Many historical mailers that a message may encounter on other systems will not be able to handle lines that long, however.

History of Decisions Made

Any references to networking have been removed as being outside the scope of the document.

The developers of the standard felt strongly that a method for applications to send messages to specific users was necessary. The obvious example is a batch utility, running noninteractively, that wishes to communicate errors or results to a user. However, the actual format, delivery mechanism, and method of reading the message are clearly beyond the scope of this standard.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

The intent of this command is to provide a simple, portable interface for sending messages noninteractively. It merely defines a ''front-end'' to the historical mail system. It is suggested that implementations explicitly denote the sender and recipient in the body of the delivered message. Further specification of formats for either the message envelope or the message itself were deliberately not made, as the industry is in the midst of changing from the current standards to a more internationalized standard and it is probably incorrect, at this time, to require either one.

Implementations are encouraged to conform to the various delivery mechanisms desribed in the CCITT X.400 standards or to the equivalent Internet Standards described in Internet ''Request for Comment'' (RFC) documents RFC 819, RFC 822, RFC 920, RFC 921, and RFC 1123. Copies of these Internet RFCs can be obtained from the DDN Network Information Center, SRI International, Menlo Park, CA, USA 94025. *Editor's Note: This paragraph will be reworked editorially to use a bibliographic entry after the two standards are merged. Doing so now is rather a headache.*

The standard does not place any restrictions on the length of messages handled by `mailx`, and for delivery of local messages the only limitations should be the normal problems of available disk space for the target mail file. When sending messages to external machines, applications are advised to limit messages to less than 50 kilobytes because many mail gateways impose message-length restrictions. (Note that this is usually an administrative issue based on the amount of mail traffic and disk space available on the gateways. Therefore, there is no way for this standard to require implementations to guarantee delivery of long messages to remote systems.)

Like the utilities `logger` and `lp`, `mailx` is admittedly difficult to test. This was not deemed sufficient justification to exclude these utilities from the standard. It is also arguable that they are, in fact, testable, but that the tests themselves are not portable.

When `mailx` is being used by an application that wishes to receive the results as if none of the User Portability Utilities Option features were supported, the **DEAD** environment variable must be set to `/dev/null`. Otherwise, it may be subject to the file creations described in 4.40.5.4. Similarly, if the **MAILRC** environment variable is not set to `/dev/null`, historical versions of `mailx` and `Mail` read initialization commands from a file before processing begins. Since the initialization that a user specifies could alter the contents of messages an application is trying to send, such applications must set **MAILRC** to `/dev/null`.

The description of **LC_TIME** uses ''may affect'' because many historical implementations do not or cannot manipulate the date and time strings in the incoming mail headers. Some headers found in incoming mail do not have enough information to determine the time zone in which the mail originated, and, therefore, `mailx` cannot convert the date and time strings into the internal form that can then be parsed by routines like *strftime* () that can take **LC_TIME** settings into account. Changing all these times to a user-specified format is allowed, but not

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

required.

The paginator selected when **PAGER** is null or unset is partially unspecified to allow System V historical practice of using `pg` as the default. Bypassing the pagination function, such as by declaring that `cat` is the paginator, would not meet with the intended meaning of this description. However, any ‘‘portable user’’ would have to set **PAGER** explicitly to get his or her preferred paginator on all systems. The paginator choice was made partially unspecified, unlike the **VISUAL** editor choice (mandated to be `vi`) because most historical pagers follow a common theme of user input, whereas editors differ dramatically.

Options to specify addresses as ‘‘cc’’(carbon-copy) or ‘‘bcc’’(blind-carbon-copy) were considered to be format details and were omitted.

A zero exit status implies that all messages were *sent* , but it gives no assurances that any of them were actually *delivered* . The reliability of the delivery mechanism is unspecified and is an appropriate marketing distinction between systems.

In order to conform to the utility syntax guidelines, a solution was required to the optional *file* option-argument to `-f`. By making *file* an operand, the guidelines are satisfied and users remain portable. However, it does force implementations to support usage such as:

```
mailx -fin mymail.box
```

The `-F` was incorrectly stated in previous drafts to apply to Receive Mode, but it has only been in `mailx` Send Mode historically. It, along with the `Copy`, `Save`, `followup`, and `Followup` commands, has been omitted. Historical systems used these to save messages in a file whose name is ‘‘derived from the author (or recipient) of the message.’’ Because of some unusual network mail addresses currently in use, and because POSIX does not specify anything about the content of the mail headers, it is not possible for this standard to identify the algorithm used for this derivation; since files would therefore be created in an unpredictable manner, this is unacceptable for portable usage.

The `no`*name* method of unsetting variables is not present in all historical systems, but it is in System V and provides a logical set of commands corresponding to the format of the display of options from the `mailx set` command without arguments.

The `ask` and `asksub` variables are the names selected by BSD and System V, respectively, for the same feature. They are synonyms here.

The `mailx echo` command was not documented in the BSD version and has been omitted here because it is not obviously useful for interactive users.

The default prompt on the System V `mailx` is a question mark, on BSD `Mail` an ampersand. Since POSIX. 2 chose the `mailx` name, it kept the System V default, assuming that BSD users would not have difficulty with this minor incompatibility (that they can override anyway).

The meanings of `r` and `R` are reversed between System V `mailx` and Sun OS `Mail`. Once again, since POSIX. 2 chose the `mailx` name, it kept the System V default, but allows the Sun OS user to achieve the desired results using `flipr`, an internal

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

variable in System V `mailx`, although it has not been documented in the *SVID*. 7

The `indentprefix` variable, the `retain` and `unalias` commands, and the `~F` and `~M` command escapes were adopted from 4.3 BSD `Mail`. 7

The `version` command was not included because no sufficiently general specification of the version information could be devised that would still be useful to a portable user. This command name should be used by suppliers who wish to provide version information about the `mailx` command.

The ''implementation-specific (unspecified) system startup file'' has been traditionally named `/etc/mailx.rc`, but this specific name and location are not required. 7

4.52 `read` — Read a line from standard input

⇒ **4.52.2 `read` Description.** *In subclause 4.52, add the following to the end of the second paragraph in the Description subclause:*

If standard input is a terminal device and the invoking shell is interactive, `read` shall prompt for a continuation line when:

— The shell reads an input line ending with a backslash, unless the `-r` option is specified.

— A here-document is not terminated after a `<newline>` is entered.

⇒ **4.52.5.3 `read` Environment Variables.** *In subclause 4.52, add the following environment variable in the correct sorted order to the Environment Variables subclause.*

PS 2 This variable shall contain the prompt string that an interactive shell shall write to standard error when a line ending with a backslash is read and the `-r` option was not specified, or if a here-document is not terminated after a `<newline>` is entered.

⇒ **4.51.6.2 `read` Standard Error.** *In subclause 4.52, replace the Standard Error subclause with the following.*

Used for diagnostic messages and prompts for continued input.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.