LISP Machine

ZMail Manual

Richard M. Stallman

M.I.T. Artificial Intelligence Laboratory

ZMail Manual

First Edition, ZMail Version 50 (System 94)

April 1983

Richard Stallman

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research Contract number N00014-80-C-0505.

Summary Table of Contents

Table of Contents

Preface

This is a reference manual for ZMail, the Lisp machine system's facility for reading, managing and sending mail. It describes how to use ZMail, but not how to customize it, except for the profile options which you can set with ZMail's own profile editing mode.

Command names and names of keyboard characters are in bold face and capitalized, as in **Control** and **Select Referenced Msg**. Names of menu items are also in bold face, except that italics are used for menu items that appear on the screen in italics. Menu items are capitalized as they appear in the menu (or at least that is the intention).

A general knowledge of the conventions of operating the Lisp machine is necessary for understanding this manual. For the most part, you do not need to know anything about programming in Lisp. The exceptions are a few functions that it may be useful for your programs to call to interface with ZMail.

Many Lisp variables affecting ZMail's operation are documented in this manual (all in the **zwei:** package, though that is not stated explicitly below). These variables are ZMail profile options. While you can refer to or set them with Lisp code, generally you would set them with the ZMail profile editor (see chapter 18, page 57), which you can use without knowing how to program in Lisp. When a variable has a fixed set of alternative values, these are documented as they would appear in the ZMail profile editor. The actual Lisp values of the variables can be figured from the following rule: replace all imbedded spaces with hyphens, and intern the string in the keyword package. **Yes** and **No** are exceptions, represented by **t** and **nil**.

Any comments, suggestions, or criticisms will be welcomed. Please send Arpa network mail to BUG-LMMAN@MIT-OZ.

Those not on the Arpanet may send U.S. snail to

Richard M. Stallman
545 Technology Square, Room 791
Cambridge, Mass. 02139

Note

The Lisp Machine is a product of the efforts of many people too numerous to list here and of the former unique environment of the M.I.T. Artificial Intelligence Laboratory. The author of this manual believes that the commercialization of the computer industry hinders the further development of systems such as described herein. He considers proprietary software morally objectionable and plans to dedicate his career to promoting the sharing and free exchange of software. For more information on this issue, write to the address above.

ZMail was written primarily by Mike McMahon. Sarah Smith helped to correct this manual. Eugene C. Ciccarelli (author of Babyl) wrote the section on the Babyl file format, one day when the disks were not getting too many errors.

1. The Scheme of ZMail

There is always a ZMail window in your Lisp machine world. You can select it with the system menu Select option or by typing System M. Normally you would never officially "exit" ZMail but just switch to another window when you are finished using ZMail.

Another way to enter ZMail is to call the function **zmail**.

zmail &optional *file*
Selects the ZMail window, and reads in the mail file *file* (a pathname or string) if the argument is specified.

1.1 Simple Usage

If you want to use ZMail just for reading, deleting and answering your personal mail, you need to know only seven commands.

After you invoke ZMail, type G or click left on Get New Mail in the menu. This will read in your primary mail file (your old saved mail) and also its inbox file (your newly arrived mail). As soon as the first few messages have been read in, you can begin working with them while ZMail continues to read the rest. See chapter 4, page 13 for more information about this.

To move to the next message or the previous message, type N or P, or click left on Next or Previous (see section 5.2, page 17). When you are finished with a message and no longer wish to save it, type D or click left on Delete (see chapter 7, page 21). This marks the current message, the message being viewed, as deleted. It will really disappear the next time the mail file is saved.

To reply to a message, type R or click left on Reply (see section 10.7, page 37). You can then edit the text and headers of the reply. Send the reply by typing the End key or cancel it by typing the Abort key. The M command is similar; use it for originating a new message that is not a reply (see chapter 10, page 29).

When you are finished processing your mail, type S or click left on Save Files (see chapter 8, page 24). The deleted messages will actually disappear at this time, and your mail file on the file server will be updated.

While you are working, ZMail may tell you that more mail has arrived. You can type G again or click left on Get New Mail to bring it into ZMail. Note, however, that this may take some time.

1.2 The Screen Layout

When you are using ZMAIL, the screen normally contains three large windows: the *summary window* at the top, the *command menu* in the middle, and the *message window* at the bottom. Underneath are two small windows, the *mode line* and the *input line* or *echo area*. There are four top-level screen configurations you can choose from; see chapter 17, page 56.

The message window normally displays the selected message. This is how you read incoming mail. It can also display a message you are sending.

The summary window displays one line for each message in the selected ZMail buffer, to give you a general idea of what mail you have. See chapter 9, page 25 for more information on the summary window display, how to control it, and what you can do with the mouse on it.

The command menu is the primary way of giving commands to ZMail. Clicking the mouse on an item in the menu issues a command. It usually makes a difference what mouse button you click; the left button usually does the simplest or commonest form of the command, and the right button usually gives you a second menu of alternative forms of the command.

The mode line displays information about the selected message when you are at top level. When you are not at top level, it displays information about the particular mode you are in. It always starts with the word "ZMAIL". At top level, it displays

ZMAIL *buffername* *msgnumber/total* (*attributes*) {*keywords*}

See buffers (page 5), message numbers (page 7), attributes (section 3.2, page 11) and keywords (chapter 12, page 43). *total* is the total number of messages in the selected buffer, or "??" if the buffer is not fully read in yet.

The echo area is used for command prompting and for messages about the progress of activities such as reading or saving files. It is also the place where the *minibuffer* usually appears. The minibuffer is a small editing window in which you supply small pieces of input such as the name of a file to move a message into; you can edit the input with ordinary Zwei commands, and it is finialized when you type Return (in some cases End is used instead).

ZMail also uses another kind of minibuffer, the *pop-up minibuffer* (flavor zwei:temporary-mode-line-window), which appears temporarily when needed. This unusual minibuffer is used for input for commands in pop-up menus, and usually appears next to the pop-up menu.

At all times, the mouse documentation line at the bottom of the screen says what you can do using the mouse buttons at the current position of the mouse. You can explore the possibilities of using the mouse by moving the mouse around and watching the mouse documentation line.

1.3 Keyboard Commands

You can give commands to ZMail by typing on the keyboard as well as by clicking the mouse. Most of the commonest commands, in their simple forms, can be done either way. Since the keyboard command is usually a single character, it is usually easier to type it than to use the mouse.

Some less frequently used commands are *extended commands*. This means that they consist of the character X followed by a command name, terminated with a carriage return. Extended commands in ZMail are about the same thing as extended commands in the Zwei editor, except that you can type them with X instead of Meta-X (though Meta-X works too).

All ZMail commands have names as extended commands, including those that are available with single characters or through the command menu. For example, the Delete command can be typed as X Delete, which is the same as typing D or clicking on the Delete item in the command menu. What the command does when executed depends on whether it was invoked with the keyboard or the mouse, and on which mouse button was used to invoke it.

Keyboard commands can be given an argument simply by typing digits before entering the command. (Control, Meta, etc. digits also work, as in Zwei.) As in Zwei, if you delay in giving the command, the argument begins to echo.

1.3.1 Keyboard Help Commands

ZMail provides the same self-documentation facilities as the Zwei editor. To request help, type the Help key, followed by a character saying what kind of inquiry you are making (or by another Help, if you don't know what's available).

A few important help inquiries are:

Help C Type Help C followed by a keyboard character or clicking the mouse, to find out what that keyboard or mouse command does. If you click the mouse, be sure to click it in the place on the screen that you want to find out about.

Help D Type Help D followed by a ZMail extended command name, to find out what it does.

Help A Type Help A followed by a topic (such as Delete or Mail) to get a list of ZMail commands whose names contain the string you typed, and how to invoke them.

Help L Type Help L to find out the last 60 commands you typed, in case you forget what you did.

2. Mail Files and Buffers

The purpose of a mail editor is to operate on *mail files*: files stored permanently on a file server, containing *messages*. Each user has a *primary mail file* where his personal messages go. On ITS, your primary mail file is called either *directory*; *your-username* RMAIL or *directory*; *your-username* BABYL. On Tenex or Twenex, it is called <*your-directory*>*your-username*.BABYL. On Unix, it is called *your-directory*/mbox or *your-directory*/*your-username*.babyl.

You can create other mail files, and redistribute mail into them from your primary mail file. On certain file servers, you can cause mailing lists to forward mail to other mail files instead of your primary mail file; that is a system-dependent matter in which ZMail is not directly involved.

There are several standard formats for storing mail in mail files. Each operating system generally has a standard; for ITS, this is RMAIL format; for Tenex and Twenex, the so-called Tenex format; for Unix, the so-called Unix format. Then there is ZMail's own favorite format, called Babyl format, which is best because it allows more of ZMail's features to be used. Finally, there is the "text mail file" which is formatted so as to look good when printed out, and cannot be read back into ZMail (see section 13.2, page 47).

ZMail knows which formats of mail file are typically found on each file server operating system, and generally recognizes the format of a file automatically from the contents of the file. When it does not, you can use the command **X Select Arbitrary Format Mail File** to read in the file (see page 62).

Most ZMail commands do not work directly on a mail file, for two reasons: access to file servers is slow, and it is desirable not to make changes in the file stored on the file server until you have had a chance to reconsider them.

Instead, when you refer to a mail file for the first time, ZMail copies the contents of the mail file into the Lisp machine, creating something called a *buffer*. When you read a message, it comes from the buffer; when you delete a message, or edit its text, it is changed in the buffer, and so on. Eventually you give the **Save Files** command and the contents of the buffer are copied back to the file server, updating the mail file itself. Buffers that correspond to mail files are called *file buffers*.

Because reading a mail file from the file server can take a long time, ZMail lets you start working with a file buffer before the whole file has been read. Meanwhile, the ZMail *background process* continues to read in the rest of the file. While the file is being read in, you can only operate on the messages that have been read so far, and the total number of messages in the buffer will be displayed as "??". You will see additional messages appear in the summary, a few at a time, as the file is read. Finally the correct total number of messages will appear, indicating that read-in is complete.

Because reading in mail files takes time, you may not wish to enter ZMail and then wait. An alternative is to *preload* ZMail while using the Lisp machine for other things. Later on in the session, when ZMail is fully preloaded, you can enter it and work on your mail without delay.

zwei:preload-zmail &rest *files*

Asks the ZMail background process to load all the *files* into ZMail. Each argument should be a pathname or a string. The loading is done with low priority so as not to interfere with your other activities. If you enter ZMail and refer to one of the files before it is fully loaded, the rest of it will be loaded right away with high priority.

Some buffers do not correspond to files at all. They live and die inside ZMail. They are called *temporary buffers*, and their use is to record a particular set of messages taken from one or more mail files so that you can work with them as a group. Temporary buffers cannot be saved, and are lost when you end your session, but no mail is lost this way since each message must belong to a file buffer. Aside from this, temporary buffers can be used just like file buffers.

Each message resides in one file buffer and any number of temporary buffers. You can select the message through any of these buffers, and changes you make will be visible later no matter how you come across the message. Initially a message is found only in one buffer, a file buffer; you can move it into temporary buffers, or remove it from them.

The file buffer that the message is in is called the *owner* of the message. For any one message, the owning mail file never changes. When you "move" a message to another file buffer, in fact a copy is created there. The original message is deleted (this is actually optional), but still exists until its owning mail file buffer is expunged.

At any time, one buffer is *selected*. The summary window summarizes the messages in the selected buffer; the Next and Previous commands move through the selected buffer, and the Map Over command operates on all the messages in the selected buffer. In simple use of ZMail, the buffer for your primary mail file is selected.

zmail-startup-file-name Mail file read in at startup. *Variable*

The value is a pathname or string naming a mail file. When your ZMail init file is read in, this mail file will be read in automatically.

3. Messages

Each communication from another user to you is called a *message*. The data in a mail file is made up of messages. Within ZMail, the data object that represents one message in a mail file is also called a "message". Each message has a message number which is its position among all the messages in the buffer; this number is displayed in the mode line and in the summary window. If a message is in more than one buffer, it has a different number in each buffer.

A message contains *headers* and *text*. Also, various *attributes* may be attached to it, and *keywords* may be assigned to it, but these are not part of the contents of the message. See attributes (section 3.2, page 11) and keywords (chapter 12, page 43).

The headers of a message have been compared to the things written on the envelope of a letter, while the message text has been compared to what goes inside it. For example, the headers say who sent the message, who it was sent to (presumably including you, if you received it), and when it was sent. When a message is displayed in ZMail, the headers are at the top, followed by the text, with a blank line in between. For example:

```
Date: April 11, 1983 1:20 am
From: RMS @ MIT-OZ

Those were the headers; this is the text.
```

The headers are made of fields, each consisting of a field type and data that goes with it. There is a fixed set of field types, each with its own standard meaning and usage. For example, the person who sent the message is stated by the From field; the people sent to, by the To field; the date of mailing, by the Date field. By contrast, the text of the message has no standard interpretation or purpose except to be read by the people who receive the message.

When you use ZMail to look at a message you have received, the headers of the message appear at the top, followed by the text. Usually a blank line separates the headers from the text. Each header field begins with a word or multiple words, starting in the first column and terminated by a colon. This word or words are the field type ("From", "Date", etc.). Whatever comes after the colon is the date of the field (the author's name, the date, etc.). If the data of a field is more than one line long, the lines after the first are indented to show that they are not new fields. For example:

```
From: Richard M. Stallman <RMS@AI>
To: LOSER@AI, WINNER@AI,
     SOMEONE-ELSE@MC
Date: Not this week, sorry
Subject: Messages with invalid Date fields.
```

Some header fields are present for your information only. Some others are used automatically by ZMail at times. For example, the Reply command uses the From field to determine where to send the reply.

When you send a message, you can specify the headers as well as the text. You specify some header fields, such as Subject, simply to make them appear in the message. Others, such as To, have additional effects on how the message is sent. (The To field you specify controls who the message is sent to, as well as showing up in the headers that are sent.) ZMail puts some header fields automatically into messages you send. The Date field is always provided automatically, and the From field is provided if you do not specify one. The Subject field of a reply is initialized to be a copy of the subject of the message you are replying to, if it has one, but you can delete or change the subject before you send the reply.

3.1 Message Header Fields

BCC Blind CC. These recipients get a copy of the message but are not mentioned in the headers of the copies received by the ordinary (non-blind) recipients. There will only be a BCC field in a message you receive if you were one of the BCC recipients, or were forwarded the message by one of them. You put a BCC field in a message you are sending if you wish to specify BCC recipients for it.

BFCC Blind FCC. Like BCC except that the message is put into a mail file within ZMail rather than actually mailed. Like the BCC field, this field does not actually appear in the header of a message except in the copies sent to the BCC and BFCC recipients. You specify this field in a message you are sending if you wish to request delivery of the message into mail files within ZMail.

CC Carbon Copy. These recipients get copies of the message just like the To recipients. There are only two differences between To and CC: ZMail always insists on having at least one primary (To or FTo) recipient, and CC does not count for this; the human reader will interpret the message from a different point of view depending on whether he is a To recipient (the message is addressed to him), or a CC recipient (the message is merely being shown to him).

You can put a CC field in an outgoing message to specify CC recipients.

Date The date and time of origin of the message. This appears in every message. You never specify one when sending a message; it is filled in automatically.

Draft-composition-date
Indicates the date and time when a saved draft was written. It appears only in saved drafts. See page 29.

Expiration-date
Indicates the date and time after which the message is no longer useful. An optional ZMail feature deletes expired messages automatically (page 22). You can put an Expiration-date field in a message you are sending, and it may also be useful to put one into a message you have received with the X Set Expriration Date command.

FCC File Carbon Copy. The contents of the field consists of one or more pathnames separated by commas. You put an FCC field into a message you are sending to cause copies of it to be moved into files within ZMail. ZMail does this by reading the files into ZMail buffers and then adding a message to each one.

From The From field in a message received indicates who sent it, as an address to which replies can be sent. A From field appears in every message received. Normally the From field of a message you send is generated automatically based on your user name (the value of **user-id**), but you can specify one explicitly if you are not actually the person who is logged in and you want the recipients to know who really originated the message.

FTo File To. This is the same as FCC except that the presence of an FTo satisfies ZMail's demand for at least one To recipient in each message sent. If you wish to send a message to one of your mail files directly within ZMail, bypassing the mail system of your file server host, you can use this.

In-reply-to This field contains references to messages which this message is intended as a reply to. It can be used by the recipients to match up related messages. (In ZMail, this is done with the conversation commands; see chapter 14, page 48.) A properly formatted In-reply-to field can be added to a reply you are sending with the command **Meta-X Add In Reply To Field** during editing of the draft (page 32); a profile option causes them to be added automatically to every reply (page 39).

Mail-from This header field is put into messages received over the network by certain file server hosts.

Message-ID This header field is meant as a standard way of uniquely identifying one message among all the message sent on any machine in all human history. At least, we hope there will never be a pair of messages with the same message-id.

Message-id fields may appear in messages received if the originating hosts and mail composition programs chose to generate them. If you wish to have message ids in your outgoing mail, a ZMail profile option causes them to be generated automatically.

Redistributed-by
Redistributed-date
Redistributed-to

Redistributing a message with ZMail or various other programs adds one of each of these fields to the header (section 10.5, page 36). They state who redistributed the message, when, and to whom (presumably including you). You do not write these fields yourself.

Remailed-by
Remailed-date
Remailed-to These mean the same thing as Redistributed-by, -date and -to. Different mail processing programs use one or the other.

Resent-by
Resent-date
Resent-to Yet another series of names for the same thing; these are the official names adopted in 1982 for use in Internet.

Resent-CC
Resent-reply-to
Resent-sender

These may be used with the previous three. Resent-reply-to and Resent-sender are to Resent-by as Reply-to and Sender are to From, I guess.

References This field identifies messages whose contents are related to this message. Each line of the field identifies one other message, either by message-id or by author and date. The command **Meta-X Add References Field** can be used to add a properly formatted References field to an outgoing message (page 32). The conversation commands use the References fields of messages in your ZMail buffers to match up related messages.

A References field looks like and is used like an In-reply-to field. The only difference is in what they signify to a human reader, and in the commands that generate the fields in outgoing messages.

Reply-to This field says who a reply should be sent to, overriding the From field which is what usually supplies this information. The ZMail **Reply** command knows how to recognize both Reply-to and From fields in the message being replied to.

You can put a Reply-to field in an outgoing message, if you like, to direct replies elsewhere. The effect of this is about the same as the effect of specifying your own From field, but the meaning to a human recipient is different. Putting in From: JDoe@HERE says "I, JDoe, am the person really sending this", whereas putting in Reply-to: JDoe@HERE says "I am not JDoe, but reply to him rather than to me."

Return-path This field gives information on how to get a message back to the ultimate originating host of the message, in case that host is not known to the destination host. Return-path fields are added automatically as a message moves from one network to another. You do not specify them yourself in outgoing mail.

Sender This field identifies the actual login name of whoever sent the message, and is present only if that is different from what the From field says. ZMail inserts a Sender field automatically in an outgoing message if you specify the From field yourself.

Subject This field contains a line of text whose purpose is to tell humans briefly what the message is about. Most of the space in the summary window line for a message is devoted to displaying the message's subject, if it has one. The ZMail **Reply** command provides a default Subject field copied from the message you are replying to.

To This field names the primary recipients of a message. You must specify a To recipient (or else an FTo file) in every message you send. Other kinds of recipients are specified with CC (or FCC for files) and BCC (or BFCC for files); these recipients are optional.

Header fields on messages in ZMail files are parsed by ZMail once and for all, and from then on are represented as properties with names in the keyword package on the **msg-status** of the message. Thus,

```
(get (locf (zwei:msg-status msg)) ':subject)
```

to get the Subject field.

For fields such as To and From, whose values are one or more mailing addresses, the value obtained in that way is a list of address-descriptions, each of which is a list of alternating properties and values. The properties found in an address-description include

:host Followed by a list of strings, a path of hosts to the target host.

:name Followed by a string, the mailbox name at that host.

:personal-name
Followed by a string, the user's "personal name". This is the "Winner" in "Winner <WNR@FOO>".

:interval Followed by a list of two ZWEI buffer-pointers to the beginning and end of the text for this mailing address.

:distribution-list or :bracketed-list
These indicate an address which is is composed of a collection of other addresses. It is followed by a string that names the collection. The :inferiors property will also be present to describe the contents. A distribution list looks like "*name: address, address...;*" and a bracketed list is "*name <address, address...>*" with more than one address inside.

:inferiors Followed by a list of more address-descriptions for the addresses in the distribution list or bracketed list.

:include or :postal
Followed by a string, these represent ":include:" or ":postal:" addresses.

3.2 Message Attributes

Attributes are properties assigned to a message automatically by ZMail as a result of the ZMail commands you perform on the message. There are a fixed set of attributes, each of which has a defined meaning: At top level in ZMail, the attributes of the selected message are displayed in the mode line in parentheses.

To contrast attributes with keywords (see chapter 12, page 43): keywords are assigned to messages only by explicit commands from the user, any name can be used as a keyword, and it is up to the user to decide what the keyword signifies.

Answered A reply has been sent for this message by means of the Reply command. (ZMail cannot tell whether you have sent a message which a human would consider a "reply").

Bad-header This message has a header field that is syntactically invalid.

Deleted This message has been deleted; it will go away when its owning mail file is expunged.

Filed A copy of this message has been moved into another mail file with Move to File.

Forwarded This message has been forwarded with the Forward command.

Last This is the last message in the selected ZMail buffer. (This attribute is somewhat funny, as it comes and goes on a message when you switch buffers.)

Recent	This message has been obtained as new mail since the last expunge of its owning mail file.
Redistributed	This message has been remailed with the Redistribute command.
Unseen	This message has never been selected.

Attributes are represented as properties with names in the zwei package on the msg-status of the message. Thus,

```
(not (null (get (locf (zwei:msg-status msg)) 'zwei:filed)))
```

to see whether a message is filed. Exception: the Bad-header attribute is represented by zwei:losing-headers.

4. Getting New Mail; Inboxes

Your mail file is never the same file that the host computer delivers mail in. Instead, the file mail is delivered in is called the *inbox* file. Two different files are used so that you will not have problems of losing mail that arrives while you are running ZMail.

ZMail moves the new mail from the inbox file to the mail file's buffer in an operation known as *getting new mail*. Each mail file for which mail can be delivered has its own inbox file; ZMail knows automatically how to find a mail file's inbox. Mail files which are filled only with ZMail (by moving messages from other mail files) and do not receive any mail directly do not need inbox files.

ZMail knows the inbox file for your primary mail file because it knows file server's operating system's convention for where to deliver mail for a user. Mail files other than your primary mail file must be Babyl files in order to have inboxes. A Babyl format mail file can explicitly record the name of the corresponding inbox file with the Mail file option (see section 20.3, page 65). You specify this option when you create the mail file, or with the File Options box in the profile editor (see section 18.5, page 61). The other mail file formats do not provide for this feature, so they have no way to record a name for the inbox.

A consequence of this is that when documentation for your host operating system (Twenex, Unix, etc.) speaks of "your mail file", the file it speaks of is really the inbox file. Your primary mail file itself may not be known to any programs running on the host computer, except those which use the same strategy of moving mail between files that ZMail uses, such as BABYL on ITS and Twenex, and RMAIL on ITS.

To get new mail for your primary mail file, click left on Get New Mail, or type G. To get new mail for some other mail file, click right on Get New Mail. You then type the file name. You can specify a file not read into ZMail yet, and ZMail will read it in; you can also specify a nonexistent file, and ZMail will create it.

Getting new mail for a mail file works by renaming the inbox (so that additional mail will not be added to it while ZMail is reading it) and reading the contents of the old inbox into the mail file buffer. Meanwhile, any arriving mail goes into a new inbox file, not the one ZMail has renamed. Getting new mail does not in itself change the mail file, only the file buffer, but normally ZMail begins to save the buffer for you "in the background" while letting you continue to type ZMail commands.

The renamed inbox file is not deleted until your mail file is saved. This is to make sure you do not lose any messages if your Lisp machine crashes. If you save the mail file after getting new mail, then the new messages are all safe in your mail file (unless you explicitly got rid of them), so they will not be lost. If you fail to save the mail file, the renamed inbox waits for you to run ZMail again and get new mail. Then, ZMail will merge in the old, renamed inbox a second time. The old inbox is only deleted once it is safe to do so.

While you are running ZMail, it will check from time to time whether there is anything new in your primary mail file's inbox. If it finds anything, a message "You have new mail" will appear in the echo area.

inhibit-background-mail-checks Don't check for new mail in the background. *Variable*
Yes or No, with No (do check!) as the default.

If you do get additional new mail, you can do Get New Mail again to bring it into ZMail. However, the mail file must have been saved first. This is because the old, renamed inbox cannot be deleted until you save the file, and the new inbox cannot be renamed until the old renamed one is deleted. Since normally ZMail begins to save the mail file whenever you get new mail, if you ask to get new mail again, it just waits for the save to be finished (or it may have finished already). If you turn off the automatic saving, doing Get New Mail again will begin the save if necessary.

inhibit-background-saves Do not automatically save after get new mail. *Variable*
Yes or No, with No (do save!) as the default.

If the mail file has been saved once since last getting new mail, it does not have to be saved again, even if you have changed it since saving it. The important point is that the new mail you got last time has to be safe on the file server. If you have deleted some of it since then, saving the deletions or not is your choice.

always-jump-after-get-new-mail Move to first message even when no new mail. *Variable*
When you get new mail, the first new message becomes selected. If this option is Yes, when you try to get new mail and there is none to get, the first old message is selected. Yes or No, with No as the default.

ZMail knows that some mail files store mail in forward order of receipt, while others store it in reverse order. Forward order is called "appending", a term which we usually take to mean "adding at the end". Reverse order is called "prepending", or "not appending". System-depending mail file formats usually have a system-imposed convention; for example, Tenex-format mail files always store mail in forward order. In a Babyl or RMAIL format file, you can specify the order. ZMail will automatically add new mail to the front or end of the file to maintain the ordering that the file is supposed to have. In a Babyl file, you can also specify whether to reverse the order of the messages when they are transferred from the inbox file to the mail file, in case those two files use opposite orderings. This is controlled by the Reverse New Mail file option.

Each Babyl or RMAIL file's ordering (appending or not) is controlled by the Append file option which you can specify with the profile editor (see section 18.5, page 61). In addition, this profile option controls what will be done when you create a new Babyl or RMAIL file:

new-mail-file-append-p Should newly created mail files append? *Variable*
This controls whether a newly created mail file is initialized to add messages at the beginning ("prepending") or at the end ("appending"). The alternatives are Append, meaning new mail files should append; Prepend, meaning new mail files should prepend; Sticky, meaning a new mail file should copy the setting of the mail file that is current when you create it; and Ask, meaning you should always be asked whether to append or prepend each time you create a mail file. The default is Sticky.

4.1 GMSGS: Reading System Announcements

Some file servers (ITS and Twenex at present) have the ability to record general announcements and deliver them to you as mail.

Rather than putting a copy of each announcement into your inbox, which would waste a lot of disk space when there are hundreds of users, the announcement system (GMSGS) remembers which announcements each user has seen, and can give you the new announcements when you are ready to read them, in ZMail.

You can get new announcements at any time with the extended command X Gmsgs. This is a process nearly the same as getting new mail. The file server puts the new announcements for you into a file that is like an inbox, and ZMail merges them into your file buffer.

You can also have GMSGS done for you automatically, under control of an option in your profile.

run-gmsgs-p Run GMSGS before getting new mail. *Variable*
Values are Yes, No and Once only, with No as the default. Once only means run GMSGS the first time new mail is read in, but not on subsequent occasions.

gmsgs-other-switches Other switches to supply to GMSGS server. *Variable*
The value is a string containing zero or more switches for GMSGS. It is significant only when GMSGS is run. The default is /Z.

Normally the GMSGS server is invoked on the host on which the mail file is stored (the mail file that announcements are going into). A Babyl mail file can have a GMSGS-Host file option which specifies a different host. These options are set with the profile editor (see section 18.5, page 61).

5. Selection

At any time one ZMail buffer is the selected buffer and one of its messages is the selected message—except when there are no buffers at all. The selected message appears in the message window and is the subject of most ZMail commands. The selected buffer is described in the summary window, and is used for the basic ZMail motion commands.

5.1 Selecting and Creating ZMail Buffers

To select a buffer, click the right button on the Select File menu item. This gets a menu which includes all the buffers you have ever selected, any additional mail files that appear in your profile (see chapter 18, page 57), plus four special items: *Find File*, *Mark Summary*, *Subset* and *Abort* (the last is the way to get out of the menu).

If you click on a buffer listed in the menu, that buffer is reselected.

Click on *Find File* to select a mail file. You use a minibuffer to type the mail file's pathname. Then the mail file is read into a file buffer, which is selected. If you specify a file that does not exist, ZMail creates it, after first asking you to specify the file options (see section 18.5, page 61).

Click on *Mark Summary* to create a new buffer and fill it with messages chosen from the selected buffer. After you type the name for the new temporary buffer, move the mouse to the summary window and click on the lines for the messages you want to mark. An X appears on the lines you have selected. To remove the X from a line, click on it again. You can scroll the summary window if necessary to find all the messages you want to mark. When you are finished marking messages, type the End key and the new buffer will be selected. Alternatively, type the Abort key and all will be canceled.

Click on *Subset* to create a new subset buffer (click middle on Select File does this too). The predicate selection window appears, a frame containing several menus and boxes that you use to specify a universe and a predicate. When you click on the Done box, a new temporary buffer is filled with all the messages in that universe that match the predicate. See chapter 15, page 49 (predicates) for more about these matters.

Clicking left on Select File is a quick way to switch to the previously selected buffer. Repeating this alternates between two buffers.

The extended command X List Buffers prints a list of all buffers in ZMail, and all additional mail files ZMail knows about but has not read in. You can click on one to select it, or flush the display by typing a space.

5.2 Selecting a Message

Once you have selected a buffer, you can move between messages in it. The Next, Previous and Jump commands are used for this. In addition, you can select a message by using the mouse in the summary window.

The Next command is used for moving forward in the selected buffer. If you type N or click left on Next, you move to the next non-deleted message in the buffer. That is, a new selected message is found by looking forward from the message that was already selected.

If you type N with a numeric argument, you move forward as many messages as you specify.

If you click right on Next, you get a menu of six different ways of moving forward. These alternatives are:

Next Undeleted
: Move to the next message in the buffer that is not deleted. This is what you get from clicking left on Next.

Next Unseen
: Move to the next message in the buffer that has not been seen yet. A message is marked as "seen" when its contents are displayed in the message window. Merely showing up in the summary does not constitute being seen.

Next
: Move to the next message in the buffer, including even deleted messages.

Last Undeleted
: Move to the last (that is, final) message in the buffer that is not deleted. This is what you get by default from clicking middle on Next, but you can make that do any of these six things by means of a profile option.

Last Unseen
: Move to the last message in the buffer that has not been seen yet.

Last
: Move to the last message in the buffer, including even deleted messages.

next-middle-mode Middle button on Next command. *Variable*
This controls what happens when you click the middle button on the Next menu item. The alternatives are: Next Undeleted, Next Unseen, Next, Last Undeleted, Last Unseen, and Last. The default is Last Undeleted.

Typing Control-N moves to the next message, including deleted messages.

The menu item Previous is just like Next but moves backwards instead of forwards. Clicking right on Previous gets you a menu of Previous Undeleted, Previous Unseen, Previous, First Undeleted, First Unseen and First. The keyboard commands P and Control-P move to the previous undeleted message, and to the previous message including deleted messages, respectively.

previous-middle-mode Middle button on Previous command. *Variable*
This controls what happens when you click the middle button on the Previous menu item. The alternatives are: Previous Undeleted, Previous Unseen, Previous, First Undeleted, First Unseen, and First. The default is First Undeleted.

The J command selects the first undeleted message in the selected buffer. A numeric argument is interpreted as a message number; you would generally find the message number of the message you want to select from the summary window.

You can also select any message in the buffer by finding it in the summary window and clicking left there. Scroll the summary window if necessary to find the message you want.

The command Control-F searches through the rest of the selected buffer, starting at the cursor in the selected message, for a search string which it reads from the keyboard. The next message which contains the string is selected.

5.2.1 The Point Pdl

The *point pdl* remembers the last several messages that have been selected. Whenever you select a message, except for sequential motion with the Next and Previous commands, the previous selected message is remembered on the point pdl.

The command Control-Meta-Space selects the last message on the point pdl. The message selected before the Control-Meta-Space goes on the point pdl in its stead. As a result, repeating this command alternates between two messages. When given a numeric argument *n*, it cycles through the last *n* messages (the currently selected one, and *n*-1 from the point pdl). A negative argument cycles in the other direction.

Clicking middle on Jump lets you choose a message from the point pdl with the mouse. It prints summary lines in the summary window for all the messages in the point pdl. Click on a line to select the message.

6. Viewing and Editing the Selected Message

If a message is too long to fit in the message window all at once, you must scroll through it. The mode line tells you where you are in the message by displaying one of these three strings:

--More Below--
: The beginning of the message is visible in the window, but the end is not.

--More Above--
: The end of the message is visible in the window, but the beginning is not.

--More Above and Below--
: Neither the beginning nor the end is in the window now.

The character Period moves to the beginning of the selected message. Space scrolls forward one screenful, and Overstrike (that is, backspace) scrolls backward. Control-V and Meta-V also scroll, just as in Zwei. Control-L redisplays with the current line at the center of the window if possible.

You can also scroll with the mouse, in three different ways. If you move the mouse off the left edge of the message window, you get a standard scroll bar. You can also brush the mouse against the top or bottom of the window, near the right hand edge. Watch for the mouse cursor to change shape to a fat arrow pointing either up or down; then you will have found the right spot.

Also, if you put the mouse on the --More Below-- or whatever string in the mode line, click left scrolls forward, click middle scrolls backward, and click right gets a menu of Forward, Backward, Beginning and End.

6.1 Editing the Selected Message

To edit the text of the selected message, type Control-R or click the mouse on the message window. You will enter the editor, and can use the ordinary editing commands to edit the text of the message. When you are finished, type End to return to ZMail top level.

The commands to move across mailing addresses (recipient or mailbox names) are available when editing a message, just as they are when composing a draft. You could use them for moving within the headers of the message. See page 31.

There are two ways of merging the text of two messages together, both extended commands. X Yank Msg inserts the text of some other message into the selected message. By default it uses the most recently deleted message (from whatever buffer), but you can also specify a numeric argument that is a message number in the current buffer. X Concatenate inserts the text of the selected message into some other message, which you choose by clicking on a line in the summary window. It deletes the selected message after copying its text.

The extended command X Occur prints all the lines in the selected message that contain a string that you specify with the minibuffer.

6.2 Header Reformatting

ZMail provides the option of reformatting headers or messages for display (in Babyl files only). How this is done is up to you. Most commonly users choose to suppress the display of certain uninteresting header fields such as Return-path.

To request reformatting, define a mail template (see section 10.3, page 32) and make its name be the value of the ZMail option *default-reformatting-template*. Reformatting is done by invoking this template on each message of a Babyl file when the file is read in. Both the original and the reformatted header are saved in the mail file, so each message is reformatted only once.

default-reformatting-template Template for reformatting headers. *Variable*

The value is a symbol, the name of a template defined with zwei:define-mail-template. nil as a value means do not reformat messages; this is the default. Otherwise, the template is invoked on each message parsed in a Babyl file being read in, unless the message has been reformatted already or the Babyl file disables reformatting with the **No Reformation** file option.

7. Deleting Messages

When you no longer need a message, you delete it with the Delete command.

Deleting a message in ZMail does not mean that it disappears immediately. Instead, the message is flagged as "deleted". This shows up as a "D" after the message number in the summary window.

Deleted messages can be operated on just like any others, though many commands including the simple variety of Next and Previous will skip over them by default. In fact, you can select the message and *undelete* it with the Undelete command. A message that has been undeleted is as good as one that was never deleted.

If a message exists in more than one buffer, then if you mark it as deleted, it will show up as deleted in all of the buffers.

Deleted messages do disappear eventually. When you *expunge* a buffer all deleted messages are removed from it. If you expunge a file buffer, all deleted message in it are permanently discarded from ZMail; they are removed from the file buffer and from any other buffers they may be in. (Recall that each message must always belong to exactly one file buffer.) Normally file buffers are expunged just before they are saved.

7.1 How to Delete, and Undelete

To delete the selected message, type D or click on the Delete menu item. If you type D, or click the left mouse button, the next undeleted message in the selected buffer is selected instead of the message you deleted. Type Control-D or (unless you redefine its meaning) click middle on Delete to move to the previous undeleted message instead. If you click the right button on Delete, you get a menu of options for what message to select:

Forward	Select the next undeleted message in the selected buffer. This is what you get if you click left.
Backward	Select the previous undeleted message in the buffer.
No	Leave the message just deleted selected anyway.
Remove	Actually remove the message from the buffer now. Allowed only when the selected buffer is a temporary buffer. This is the same as the X Remove command.

Every time you do a deletion using the menu, you also change the action of the middle button for the future.

You can set the direction to move after click left, and the initial direction to move after click middle, with the profile editor.

next-after-delete Direction to move after Delete. *Variable*
The alternatives are Forward, Backward, No and Remove. The default is Forward.

delete-middle-mode Direction to move for click middle on Delete. *Variable*
The possible values are the same as for *next-after-delete*.

To undelete the selected message, type U or click on the Undelete menu item. If the selected message is not deleted, this will undelete the last previous deleted message; therefore, a U will undo the effect of a D if you have not redefined the way the D command moves.

You can delete any message by typing D with the message number as a numeric argument. You can undelete any message in a similar fashion.

You can also delete and undelete messages using the summary window. If you click right on a line in the summary, the menu of operations you get includes both Delete and Undelete. In addition, the default action of the middle button in the summary is to delete a message if it is not deleted, or undelete a message if it is deleted.

The extended command X Delete Duplicate Msgs finds all duplicate messages in the selected buffer and deletes all but one of each set of duplicates.

7.2 How to Expunge Buffers

You can expunge the selected buffer by clicking middle on the Save Files menu item, or by typing E.

To expunge other buffers without having to select them, click right on Save Files. This gives you the "expunge/save/kill menu". With it, you can specify which buffers you wish to save, which you wish to kill, and of course which you wish to expunge. Finally, click on the Do It choice box and the specified operations will occur. You can also click on Abort to cancel them all and leave the menu.

When you expunge a temporary buffer, deleted messages which are in that buffer are removed from that buffer only. When you expunge a file buffer, all the deleted messages in that buffer are eliminated completely from ZMail, removed from all the buffers they occur in.

query-before-expunge Show headers and ask before expunging deleted messages. *Variable*
If Yes, expunging prints a list of all the messages in the buffer that are about to be discarded, and then asks you to confirm the expunging of them. Yes or No, with No as the default.

A message can have an *expiration date* after which it is no longer useful. Normally only system announcements (see section 4.1, page 15) have expiration dates, but you can give the selected message an expiration date with the extended command X Set Expiration Date. Expunging a file buffer optionally expunges all expired messages.

delete-expired-msgs Automatically delete expired messages. *Variable*
Alternatives are Yes, No, Ask and Per File. The last is the default, and means that each file specifies whether expired messages in it should be deleted (with a file option; see section 20.3, page 65). Ask means that you should be asked explicitly what to do, each time a buffer is expunged.

The extended command X Remove removes the selected message from the selected buffer, which must be a temporary buffer. The buffer will no longer contain this message. Removing the message does *not* delete it; it does not cause the message to be discarded from its owning file buffer.

8. Saving Files and Exiting ZMail

When you have finished editing your mail files with ZMail, you must save the buffers back on the file server if you do not want your changes to be lost.

Usually you will want to expunge any deleted messages from the files before they are saved. You can expunge and save all mail files by clicking left on **Save Files** or typing S. (Mail files that have not changed will not be saved.)

For more explicit control over saving and expunging, click right on **Save Files**. You get a menu with three boxes for each ZMail buffer: an Expunge box, a Save box, and a Kill box. If the box is filled in, the buffer will be expunged, saved, or killed. Click on the boxes you wish to fill on (or clear out), and then click on the **Do It** box below. Alternatively, click on the **Abort** box to leave the menu and not expunge, save or kill anything.

When you have saved your files, you might wish to reboot the machine if you are finished with it, or use the **System** or **Terminal** key to go to another program. ZMail has a **Quit** command, but it isn't really needed; you might as well just use the **Save Files** command and then select whatever other program you want.

The keyboard command **Meta-~** marks the selected mail file as "not changed, not needing to be saved". ZMail will no longer attempt to save it for you, unless you change it again.

To change the name of a mail file, you use the extended command **X Rename Buffer** and then save the file under its new name. **X Rename Buffer** on a file buffer asks for a new filename. It is also allowed on temporary buffers; then it asks for an arbitrary string for the new name.

9. The Summary Window

The *summary window* occupies the top half or so of the screen, when you are at top level in ZMail. It displays one line for each message in the selected buffer. You can do various things to a message by clicking the mouse on the line for that message. If there are too many messages to fit on the screen at once, you can scroll through the list.

To scroll the summary window, bump the mouse against the top or the bottom where it says *More Above* or *More Below*. Or you can move the mouse off the left edge of the summary window to get a standard scroll bar, where all three buttons become scrolling commands. In addition, the keyboard command Control-Meta-V scrolls the summary window a whole screenful or by the specified number of lines (which may be negative).

Each line of the summary window describes one message.

default-summary-template Pattern for lines in summary window. *Variable*

The value of this variable controls how a summary line is calculated for a message. The value should be a list of alternating keywords and arguments. Each keyword specifies one printer-function which prints something about a message. The argument following the keyword is passed to that printer-function.

nil and t are also allowed as values, and treated specially. nil is interpreted as an abbreviation for

```
(:size 5 :recipients 23. :keywords t :subject t)
```

and t for

```
(:size 5 :date :date :recipients 23. :keywords t :subject t)
```

The default value is t.

Every Babyl format mail file has a Summary Window Format option (see section 20.3, page 65). Its value controls display of the summary when that file is the selected buffer, overriding this ZMail option. The possible values of the file option are the same as those of the ZMail option.

The defined keywords (all of which appear in the example) are:

:size — Prints the size of the message. The argument (5, above) is the number of columns of space to use for the printout.

:date — Prints the date of origin, time of origin or both. The argument is :date, :time or :date-and-time, to specify which.

:recipients — Prints the sender and recipients of the message. Your own name is omitted to save space; certain host names identified as "local" by your site's site configuration are also omitted. The argument is the number of columns to use.

:keywords — Prints a description of the keywords of the message, inside of braces (see chapter 12, page 43). The argument is ignored.

:subject — Prints the subject field of the message, or, if there is no subject, the first

nonblank line of text. The argument is ignored.

default-summary-template does not control the printing of the message number and the message status. These always appear at the beginning of each line. The status is a character that follows the message number, and is one of these four:

D	This message is deleted. This overrides the following conditions.
A	A reply has been sent for this message.
-	This message is unseen. It has never been the selected message.
:	This message has been seen but not replied to.

9.1 Mouse Commands on the Summary Window

Clicking left on a summary line selects the message on that line. This means that the message you chose will be displayed in the bottom half of the screen, and most commands will apply to it.

Clicking right on a summary line gets a menu of several things you can do to the message on that line. The alternatives are:

Keywords	Change keywords on this message. This works just like clicking on Keywords in the command menu, including depending on which button you click, but applies to the message you chose in the summary rather than the selected message.
Delete	Delete this message.
Reply	Reply to this message. This works just like clicking on Reply in the command menu, including depending on which button you click, but applies to the message you chose in the summary rather than the selected message.
Move	Move this message to another buffer. This works just like clicking on Move in the command menu, including depending on which button you click, but applies to the message you chose in the summary rather than the selected message.
Append	Append this message to another message. If you click the left mouse button, it is appended to the selected message. If you click the right mouse button, you get to choose the message to append to by clicking on its line in the summary.
Filter	Create a temporary buffer of messages from the selected buffer that resemble this message in some way. The first thing that happens when you click the mouse is to print a list of various characteristics of the message. Then you click on one of the lines in the list with the mouse, and ZMail finds all the messages which share the chosen characteristic and makes a temporary buffer containing them.

Clicking middle on a summary line either deletes or undeletes the message. If it is deleted to begin with, it is undeleted; otherwise, it is deleted. You can redefine the operation of click middle to be any of the alternatives available in the menu for click right, plus a couple of other alternatives; use the profile editor.

summary-mouse-middle-mode Middle button on summary window. *Variable*
Controls what happens when you click the middle button on the line for a message in the summary window. The alternatives are Keywords, Delete, Undelete, Remove, Reply, Move, Append, Filter, and two that are not in the menu for click right: Delete/Undelete, delete the message if not deleted, undelete if already deleted; Delete/Remove, delete if from a file buffer, or remove if from a temporary buffer. The default is Delete/Undelete.

There are a few other options that affect the display of the summary window:

summary-window-fraction Fraction of the frame occupied by the summary. *Variable*
This value is in effect in the normal mail editing mode. By default, it is 0.45s0.

filter-summary-window-fraction Fraction for the summary in filter mode. *Variable*
Either nil, meaning that the summary window should not appear in filter definition mode, or a flonum between zero and one. The default value is nil.

summary-scroll-fraction Amount by which to glitch summary window. *Variable*
The value is a flonum that specifies a fraction of the summary window's size. The summary window "glitches" (scrolls by a few lines) when a message just outside of the displayed part of the summary is selected.

10. Sending Mail

You can send mail by clicking left on the menu item Mail, or by typing M. You then get to edit the text and headers of the message to be sent. When you are finished, type End to send the message, or type Abort to go back to the top level of ZMail without sending the message.

Beginning to send a message creates a ZMail object called a *draft*. Aborting sending a message is not final. You can resume editing the draft with the Continue command. Then you again have the choice of sending it or aborting. Even once a draft has been sent, you can resume editing it and send it again, with changes in the text or the recipients.

Click left on Continue or type C to resume editing the last draft you were editing. Click middle on Continue to resume editing the last draft that has not been sent. To resume any other draft, click right on Continue. This gives you a menu including all the drafts that exist—all the messages you have sent or begun to compose. Click on one to resume editing it.

While editing a draft, you can save it in a separate file, or as a message in a mail file in ZMail. There are commands for saving the draft:

Control-X Control-W
: Saves the draft in a file, whose name you specify.

Control-X Control-S
: Saves the draft in a file, the same file last saved in.

Control-X Control-Meta-S
: Saves the draft as a message in a mail file loaded into ZMail.

To restore a draft, click right on Continue. The menu you get includes two special items, in italics, as well as the existing drafts. Use the menu item *Restore Draft File* to restore a draft saved in a separate file. Use the menu item *Restore Draft Message* to restore a draft saved as a message. Clicking left restores the selected message as a draft. Clicking right asks you to click on a line in the summary window to specify the message to be restored as a draft.

You can also restore a file containing a saved draft while composing some other draft, using the editing command Control-X Control-R. It reads a pathname with the minibuffer.

mail-file-for-drafts Mail file to store drafts in. *Variable*
: This is the mail file in which drafts saved as messages should go. The value should be a pathname (or string), or nil which means your primary mail file or else some other mail file present in ZMail. A value of nil appears in the profile editor as an empty line.

default-draft-file-name Default file for saving drafts of messages in. *Variable*
: This is the default pathname for Control-X Control-W. The value should be a pathname (or string) or nil. A value of nil appears in the profile editor as an empty line.

10.1 Mail Composition Window Configurations

There are three window configurations that you can use for composing outgoing mail. Profile options control which configurations the Mail and Reply commands use. Once you are composing mail, you can switch to any of the configurations with editor commands.

"One window" mode displays the headers and text of the draft together in the message window, the same window that normally displays the selected message being read. There are actually three variants of this mode, since each of the three top-level window configurations that includes the message window could be used (see chapter 17, page 56). The headers are separated from the text by a delimiter line which you should not change; by default this line contains

```
--Text Follows This Line--
```

but there is a profile option to change it (some people like the delimiter line to be blank, as it is in messages you have received). When you are editing a draft, the editor command Control-X 1 switches to this mode.

"Two window" mode displays the headers and text each in its own window. These two windows use up the entire ZMail frame; the message window, summary window and command menu do not appear. The editor command Control-X 2 switches to this mode.

"Three window" mode displays the separate headers and text windows just like two window mode, and in addition displayes the message(s) being replied to in a third window at the top of the frame. This mode is never entered automatically except by the Reply command, but you can always switch to it if you like. If there is no message "being replied to", the selected one is displayed. The editor command Control-X 3 switches to this mode.

The initial choice of configuration for all the mail sending commands (except for Reply) is controlled by the following profile option:

default-mail-window-configuration Default window configuration when mailing. *Variable*
The default value is Normal, which means not to switch window configurations for composing mail. The window which normally displays the selected message you are reading displays the draft instead. This is one of the variants of "One window" mode; which variant is used depends on what window configuration you were using to read your mail. This is the default because it makes entering and exiting mail composition fast.

The other alternatives are Send, which specifies "Two window" mode (headers and text in two separate windows), and the specific variants of "One window" mode: Both, meaning show the summary window and a mail (headers and text) window; Message only, meaning show just one window containing headers and text; and Experimental, which uses the experimental configuration of ordinary mail reading mode, with the message being sent instead of a message being read.

mail-header-delimiter Line between headers and text in one-window mode. *Variable*
The value is a string which becomes the contents of the line separating the headers and text of a draft. If the string is empty, the line will be blank. The default is "--Text Follows This Line--".

The Continue command to resume editing a draft goes back to the same window configuration that was in use the last time you were editing the same draft.

The Reply command has its own set of profile options and arguments to control the window configuration, but it uses the same set of window configurations and the same editor commands to switch between them.

10.2 Editing Commands for Message Headers

Special editing commands that understand the syntax of message headers are available when you are composing mail. Many are also available in the ZMACS Control-X M and Meta-X Bug commands, and in the functions mail and bug, though these are not strictly speaking part of ZMail.

These commands are for motion across mailing addresses (names of mailboxes), and are available when editing received messages as well as when editing drafts.

Hyper-F Forward Address. Moves forward to end of this or following address.

Hyper-B Backward Address. Moves backward to end of this or following address.

Hyper-K

Hyper-Rubout
Kill Address and Backward Kill Address. Kill forward to end or backwards to beginning of an address.

Hyper-T Exchange Addresses. Exchanges like Meta-T and Control-Meta-T, but operates on mailing addresses rather than words or s-expressions.

These commands are available only when editing drafts (outgoing mail).

Control-X A Add more text. If the text and headers are in separate windows, it moves to the text window. Otherwise it moves point to the end of the draft.

Control-X C Add CC recipient. Creates a CC field in the headers of this draft message, if there isn't one already, and positions at the end of it, adding a comma if there is already something in the field.

WIth a negative argument, deletes any CC field that exists. With a zero argument, just moves to the start of the CC field (creating the field if necessary).

Control-X T Add To Field. Like Control-X C but operates on To fields rather than CC fields.

Control-X S Add Subject field. Creates an empty Subject header field, or deletes the contents of an existing Subject field, and positions point there.

With a negative argument, deletes any Subject field outright. With a zero argument, positions to the beginning of the Subject field, creating one if necessary, but not deleting any existing subject.

Meta-X Add From Field
Like Control-X S but operates on From fields. Normally the From field of a message is generated for you. You add an explicit one if you, the person sending

the message, are not the person who is logged in.

Meta-X Add In Reply To Field
Inserts an In-reply-to field whose contents describe the message(s) you are now replying to.

Meta-X Add References Field
Inserts a References field which has describes all the messages in the selected buffer. It has one line per message, containing either the message's message-id if it has one, or else the date and sender of the message.

There are two techniques for using this: select a temporary buffer containing only the messages you have in mind and then compose a draft and use this command, or use the command with a mail file selected and delete the lines that describe messages other than the ones you want to refer to.

Meta-X Add FCC Field
Meta-X Add FTo Field
Inserts an FCC or FTo field naming a mail file which you specify with a menu like the one for the **Select** command. The menu includes all the mail files known to ZMail, and items for reading in/creating additional files.

Meta-X Add Expiration Date Field
Inserts an Expiration-date field containing a date you specify (see page 22).

Meta-X Change Subject Pronouns
Changes all forms of "I" to forms of "you" and vice versa, in the subject field. This is useful when the subject field has been copied from another message that you are replying to.

10.3 Mail Templates

ZMail templates are a means of specifying header fields automatically in messages you are sending. Each template you define has a name and its definition specifies various header fields to initialize. You can invoke a template explicitly, and you can also have various default templates that are invoked automatically in certain circumstances.

zwei:define-mail-template *Macro*

A template is defined with zwei:define-mail-template, as follows:

```
(zwei:define-mail-template name
  documentation
  body...)
```

body is Lisp code which uses convenient functions described below to modify header fields as you desire. *documentation* is a string whose first line is brief documentation (a complete sentence) and whose entire text is the full documentation. The purpose of the documentation is to say what this template is to be used for. *name* is the template's name, a symbol.

An editor command is automatically constructed from the template. Its name is created by appending "COM-" and *name*. The resulting command is made available through Meta-X when you are composing mail in Zmacs or ZMail, and when you are editing a received message in ZMail. Thus, a template named MORE-INFO would produce a command com-more-info which would be accessed using Meta-X More Info.

These are the functions recommended for use in the body of a template:

zwei:add-field *type contents*
Adds a *type* field to the header, putting in contents *contents*. Existing *type* fields are left alone. *type* should be a keyword identifying a header field, such as :cc or :reply-to. *contents* should be a string.

zwei:default-field *type contents*
Similar to zwei:add-field but does nothing if a field of type *type* already exists with nonempty contents. If an empty field exists, it is filled in from *contents*; otherwise a new field is created and filled with *contents*.

zwei:delete-field *type*
Deletes any field(s) of type *type*.

zwei:find-field *type*
Returns a buffer pointer to the first *type* field, or nil if there are none. This is for sophisticated alteration of header fields.

zwei:add-text-start *string*
Adds *string* to the text of the message, at the beginning of the text.

zwei:add-text-end *string*
Adds *string* to the text of the message, after any existing text.

The arguments to these functions can be calculated in an arbitrary fashion. Actually, templates are not limited to calling these functions. They can run arbitrary Lisp code. In particular, they can use any of the primitives that editor commands use. zwei:*interval* will be bound to an interval containing the message being operated on. In addition, if you are replying to messages or forwarding messages in ZMail, the variable zwei:*msgs* will be bound to a list of those messages (otherwise, it is nil). You can therefore easily examine those messages, such as by using zwei:msg-fits-filter-p on them to see if they match one of your ZMail filters, and make your header alterations conditional on the answer.

In addition to invoking templates explicitly with Meta-X, you can have templates that are run automatically when you begin to send a message. The ZMail options zwei:*default-forwarding-template*, zwei:*default-bug-template*, zwei:*default-reply-template*, and zwei:*default-mail-template* may have as their values the names of templates (as specified in the template definitions) that are to be run automatically when you start to compose, respectively, a forwarding of other messages, a bug report, a reply, or anything else. The variables may also be nil to specify that no template is to be invoked; this is the default. Sending with Zmacs uses, instead, the variables zwei:*default-zmacs-mail-template* and zwei:*default-zmacs-bug-template*. They too should have as values either template names or nil.

Here is an example of a template definition:

```
(zwei:define-mail-template more-info
        "Set up request for more info on a bug."
  (zwei:add-field ':cc "BUG-LISPM")
  (zwei:default-field ':subject "Your bug report.")
  (zwei:add-text-start
    (if (bit-test 1 (random))
        "I need more information in order to track
down the bug that you reported."
      "The most important thing a bug report should contain
is the precise sequence of actions that will reproduce the bug.
It should be written so that no imagination is required
to follow the directions.  If I have to imagine what to do,
there is no guarantee that what I imagine will behave the same
way as what you did.  All filenames should be given exactly.
All sequences of characters, code, etc. should be given exactly.
Never describe anything that you could present exactly.
Every description is based on a theory of what is relevant;
leaving out what you consider irrelevant cannot help me,
and can hinder me if your theory is wrong.
")))
```

Template definitions can be put in your ZMail init file, by editing the text, or they can go in your LISPM init file (since you don't need to be using ZMail to use templates). If you ask for a template to be invoked automatically, make sure you arrange to have the template defined when that occurs!

Templates are also used for reformatting headers for display. See section 6.2, page 20.

10.4 Profile Options for Sending Mail

default-fcc-list Default initial FCC list. *Variable*

Whenever you start to send a message, it is initialized to send copies to the files in this list. The copies go to the files using local mail (see below). The list is empty by default.

default-cc-list Default initial CC list. *Variable*

Whenever you start to send a message, it is initialized to send copies to the recipients in this list. The list is empty by default.

You could get the effect of *default-cc-list* and *default-fcc-list* with a suitable template set up to run by default, but the variables are kept around since they already existed, and are simpler than a template if they are all you want. If you do have a default template, the default CC's have already been put in the header when the default template is run.

mail-sending-mode How to transmit outgoing messages. *Variable*

The alternatives are COMSAT, Chaos, and Chaos Direct. Chaos is the default; it means that one of the hosts at your site which has a mail server is asked to transmit the mail to all the recipients. Chaos Direct is similar, except that each recipient on a host on your local Chaosnet is taken care of by contacting that host's own mail server directly. Thus, if there are recipients on several local hosts, the Lisp machine will contact each of them. Recipients on hosts that are reached through other networks, or whose hosts are not responding, are handled by forwarding through one of the hosts that did respond.

COMSAT is the mailer demon on ITS, and COMSAT mail sending mode means that mail is transmitted by writing a file in COMSAT's input format on a suitable ITS, using ordinary file output. This mode is only available at MIT.

default-header-force Default header force (via COMSAT). *Variable*

This option is only meaningful when you are using COMSAT for sending mail, which is possible only at MIT.

The alternatives are: None, meaning COMSAT can choose the header format; RFC733, meaning force an RFC 733 format header; Network, meaning force a typical Arpanet format header; ITS, meaning force an ITS-style single line header. The default is None.

require-subjects Require subjects on outgoing messages. *Variable*

The alternatives are Yes, No, On bug reports, and Initial but not required. Yes does not actually forbid you from sending a message with no subject. If you try to do so, ZMail will ask for a subject with a minibuffer, but if you just type Return, the message will go without a subject. The default is No.

send-header-format Format of headers in outgoing mail. *Variable*

Format of recipients in headers sent (except via COMSAT). This controls how mailing addresses are formatted in the actual outgoing header, based on information obtained from parsing the headers you specify or on any other information ZMail has. For example, in automatically generated From fields, ZMail always has available your personal name as determined from your user-name when you logged in; the question is whether to include it in the actual header.

The alternatives are: Short, use "@" to separate user and host, and omit personal names; Long, use " at " to separate user and host, and omit personal names; and Include Personal, include the user's personal name if any. The default is Include Personal.

10.5 Forwarding and Redistribution

Redistributing means sending the text of a message you have received to a new bunch of recipients.

Forwarding is a generalization of redistributing. The text of a message you have received is inserted into an editing buffer, but you can then modify it or insert additional text before you send it.

You can do any of these things by clicking right on the menu item Mail and selecting the item Forward or Redistribute from the menu that appears next. Clicking middle on Mail does one of these things also; which one is controlled by a variable in your profile:

mail-middle-mode Middle button on Mail command. *Variable*
The alternatives are Mail, Bug, Forward, Redistribute and Local. Bug is described in the following section. Local is an obsolete feature not documented because there are better ways to do what it does. The default is Bug.

When you redistribute a message, you specify only the recipients. The original message is mailed to them, with only some added header information saying who redistributed it. Redistribution does not create a draft. The commands X Redistribute All and X Redistribute Msg are also available for redistributing all the messages in the selected buffer, or just the selected message.

Forwarding a message works just like mailing a new message, except that the text is initialized to containing a copy of the headers and text of the original message. A subject field summarizing the original message is also provided. Using the Map Over command, you can forward more than message at once. The F command is a convenient way to forward the selected message.

forwarded-message-begin Format line before forwarded messages. *Variable*
This supplies the contents of the line that is put before the first message in a bunch of messages being forwarded. By default, this string is empty, so the line will be blank.

forwarded-message-end Format line after forwarded messages. *Variable*
This supplies the contents of the line that is put after the last message in a bunch of messages being forwarded. By default, this string is empty, so the line will be blank.

forwarded-message-separator Format line between forwarded messages. *Variable*
This supplies the contents of the lines that go between messages in a bunch of messages being forwarded. By default, this string is empty, so the line will be blank.

forwarded-add-subject Forwarded messages are supplied with a subject. *Variable*
Yes or No, with Yes as the default.

10.6 Sending Bug Reports

Reporting a bug is mailing a message to a bug report mailing list.

You send a bug report with the command X Bug, from the Mail-right menu, or from middle on Mail. You get a menu of the known bug report topics, which include

ZWEI	Report a bug in the Lisp machine editing software.
ZMail	Report a bug in ZMail.
LISPM	Report a bug in any other part of the Lisp machine software system.
LMMAN	Report an inaccuracy, omission or unclarity in the Lisp machine manual.
Hardware	Report a malfunction in a particular Lisp machine.
Other	If you select this, you can type in the name of the bug report topic.

The function zwei:add-bug-recipient can be used to add entries to the list of bug report topics. For example, ZWEI could have been added with

```
(zwei:add-bug-recipient "ZWEI"
    "Report a bug in the Lisp machine editing software.")
```

In addition, any system defined with defsystem can specify a bug report mailing list to go in the menu, with (:bug-reports *topic-name documentation*). The two arguments are strings.

The actual address to which the bug report is sent is made by concatenating "BUG-" and the name of the bug report topic; thus, BUG-LISPM or BUG-Hardware. The host is determined by a site option, :host-for-bug-reports, specified for your site in SYS: SITE; SITE LISP.

The bug report is automatically initialized to contain the version numbers of the software you are running and the name of the machine you are using. This information is very helpful to the person who will investigate the bug (that is, me).

10.7 Replying

Most messages you send are probably replies to messages you have received, so ZMail has a special command with its own features for replying to the selected message. You can start sending a reply by typing the character R, or by clicking on the menu item Reply.

The recipients of the reply are initialized to be the sender and recipients of the message you are replying to. Some other header fields, such as the subject, may also be initialized for you.

You can then edit the text of your reply. You can insert the text of the original message into the reply using the C-M-Y command. You can also edit the headers. When you are finished, type End to send the reply or Abort to cancel it. You can resume editing an aborted reply with the Continue command.

10.8 Reply Mode Options

While the concept of replying is simple, there are many alternative ways to initialize the header of the reply, and various window configurations that can be used for editing the reply. You can set your reply mode parameters with the profile editor; in addition, you can do one reply with different parameters by clicking the right mouse button on the Reply menu item. This gives you a menu with which you can specify the parameters for this reply only.

There are two options that pertain to replying: how to initialize the recipients, and the window configuration.

The initialization of the recipients is a matter of who should receive the reply, and which recipients should be listed as "CC" rather than "To" (admittedly, this is a subtle matter). The potential recipients are the sender of the original message, the "To" recipients of the original message, and the "CC" recipients of the original message. You can specify what to do with each of them.

The alternatives are:

All	To Sender and original To, CC original CC.
All-CC	To Sender, CC original To and CC.
Cc-All	To original To, CC Sender and original CC.
To	To Sender and original To.
To-CC	To Sender, CC original To.
CC-To	To original To, CC Sender.
Sender	To Sender.

The window configurations available for reply include displaying both the original window and the reply (Show Original), treating this as ordinary mail (Like Mail), and displaying the text of the reply with the original message inserted into it (Yank). If you do not choose to have the original message inserted to start with, you can insert it later using the Control-Meta-Y command.

Control-Meta-Y
: Yanks in the text of the message(s) you are replying to, indenting all the lines. If you specify a numeric argument, the lines are not indented.

Control-X Y
: "Prunes" headers yanked by Control-Meta-Y (or in any fashion), deleting all header fields except the sender and the date.

one-window-after-yank Just show headers and text after yanking in message. *Variable*
: Should Control-Meta-Y remove the window containing the original message from the window configuration? Yes or No, with Yes as the default.

prune-headers-after-yanking Prune headers of yanked messages. *Variable*

If this option is Yes, yanked headers are pruned automatically when they are yanked. Otherwise, you must prune them explicitly with the Control-X Y command. Yes or No, with No as the default.

If you click right on Reply, you can choose both of these options. If you click left on Reply or type R, the options are controlled by two variables, *reply-mode* and *reply-window-mode*, which you can set with the profile editor. The defaults are All and Like Mail.

Users often have a few combinations of options which they find useful. In addition to the default combination which you get with left click or with R, you can have a second combination that you get with middle click on Reply, and a third combination that you get by typing 1R (The R command with argument 1). Many users make these two the same.

Middle click on Reply is controlled by the two variables *middle-reply-mode* and *middle-reply-window-mode*, which default to Sender and Show Original. R with argument 1 is controlled by the variable *1r-reply-mode*, which defaults to Sender, and by *reply-window-mode* (so you cannot set this by itself). You can specify these variables with the profile editor.

10.9 Reply Profile Variables

generate-in-reply-to-field Automatically generate In-reply-to fields. *Variable*

Yes or No, with No as the default.

dont-reply-to People not to reply to. *Variable*

This is a list of strings. When you reply to a message, any recipients of the original message whose names begin with any of these strings will be omitted from the recipients of the reply. The default is the list ("INFO-*"), which matches all informational lists.

reply-header-format Format of recipients inserted for Reply. *Variable*

This controls how the recipients as obtained from the original message are formatted when inserted into the draft headers for you to edit. Contrast it with *send-header-format*, which has the same alternatives and controls a similar formatting operation performed when the message is actually *sent* (page 35).

The alternatives are: Short, use "@" to separate user and host, and omit personal names; Long, use " at " to separate user and host, and omit personal names; Include Personal, include the user's personal name if any; and Use Original, meaning format the name as it is formatted in the message you are replying to. The default is Short.

reply-mode Default reply to. *Variable*

Controls which recipients will receive the reply, when you click on the Reply menu option with the left button, or type the R command with no argument. The alternatives are the same as are available in the menu you get when you click right on Reply, and the default is All.

reply-window-mode Default reply window setup. *Variable*

Controls the window configuration used when you click on the Reply menu option with the left button, or type the R command with no argument. The alternatives are the same as are available in the menu you get when you click right on Reply, and the default is Like Mail.

1r-reply-mode Default reply with argument of 1 to. *Variable*

Controls which recipients will receive the reply, when you issue the R command with argument 1. The alternatives are the same as are available in the menu you get when you click right on Reply, and the default is Sender.

middle-reply-mode Default reply to for middle button. *Variable*

Controls which recipients will receive the reply, when you click on the menu option Reply with the middle button. The alternatives are the same as are available in the menu you get when you click right on Reply, and the default is Sender.

middle-reply-window-mode Default reply window setup for middle button. *Variable*

Controls the window configuration used when you click on the menu option Reply with the middle button. The alternatives are the same as are available in the menu you get when you click right on Reply, and the default is Show Original.

11. Sorting Messages

You can sort the messages in the selected buffer in many different orderings using the Sort command. In addition, any Babyl format mail file can be given a *sort predicate* as a file option; then any new messages added to the file will be inserted in the correct position according to the sort predicate. (This implies that the file will always be sorted *provided* you do not alter existing messages.)

Sorting is controlled by two options: what part of the message to compare, and whether to sort forwards or backwards. For example, you can sort forwards by date (earlier messages first), or you can sort backwards by sender (alphabetical order, with senders at the end of the alphabet first).

Here are the message characteristics you can sort by:

Date The dates (and times) at which messages were sent are compared. "Forward" means earlier dates come first.

To The recipients of the messages are compared alphabetically. "Forward" means that names at the beginning of the alphabet come first, in this and all the other alphabetical sorting options

From The recipients of the messages are compared alphabetically.

Subject The subjects of the messages are compared alphabetically.

Text The texts of the messages are compared alphabetically.

Keywords The lists of keywords of the messages are compared alphabetically (see chapter 12, page 43). Each Babyl format mail file remembers the list of keywords used in it, and each message's keywords are ordered in the same order as that list. So keywords that come earlier in the file's list will have a higher priority for determining where a message will go. You can edit the file's list using the Mail Files box in the profile editor (see section 18.3, page 60).

Length The lengths of the messages are compared. Forward means that shorter messages come first.

Position This means that messages are ordered first alphabetically by the name of the mail files they belong to; messages from the same mail file are ordered by position in that mail file. Note that a message can belong to any number of buffers, but only one mail file.

Clicking left on the **Sort** menu item does the default kind of sort. Initially, this is sorting forwards by date. Clicking right on **Sort** gets a menu with which you can choose the type and direction of sort. The type that you choose becomes the new default.

Clicking middle on the **Sort** menu item sorts the selected buffer according to its own sort predicate. This does not set the default for click left.

A Babyl file's sort predicate is the value of the Sort file option (see section 20.3, page 65). Set this with the File Options box in the profile editor (see section 18.5, page 61). The Sort option controls only which characteristic the sort is done with; the direction of sort is controlled by the Append file option, the one which normally controls which end of the file new mail is added to. You can think of this as a generalization of its usual meaning.

12. Keywords

You can assign keywords to messages to classify them for later use. For example, you might give all messages that are about bugs the keyword **bug**, and all messages about ZMail the keyword **zmail**. Later you would be able to read through or operate on all messages with the keyword **bug**, or on all messages that have the keyword **zmail** and not the keyword **bug**.

You assign keywords to messages by hand using ZMail commands. A program with less than human intelligence cannot tell from the text of a message whether it pertains to bugs, so you tell ZMail once, and it remembers from then on. Keywords are the mechanism by which ZMail remembers these things.

You can assign keywords to any message, but Babyl format is the only mail file format that can remember keywords. If you are using any other format of mail file, the keywords will not be there when you read the file into ZMail in the next session. This is one of the reasons why Babyl format is preferred.

Sometimes you can figure out simple rules that are good guides for classifying a message. For example, any message sent to BUG-ZMail is probably about ZMail. You can tell ZMail such rules and use them to provide default assignments of keywords, which you can override if you do not like what they did.

The basic way to change the keywords of a message is to click right on **Keywords**. You get a multiple highlighting menu listing all the keywords ZMail has seen in this session, with the ones assigned to the selected message highlighted. You can click on a keyword to turn it on or off. Finally, make the changes take effect by clicking on the **Do It** box, or cancel them by clicking on the **Abort** box. To specify a new keyword that has not been used before, click on the **New** box. You will then be asked to type the keyword. When you have done so, the keyword will appear in the menu with the others, and you can turn it on or off for this message by clicking on it.

Once you have turned keywords on or off with click right, you can repeat the operation on other messages with click left. Click left on **Keywords** turns on the same keywords that you turned on the last time you used the menu, and turns off the same keywords that you turned off the last time you used the menu.

The keywords of the selected message appear in the mode line within braces, after the message flags that follow the message number. You can click on them too; it is the same as clicking on **Keywords**. Also, each line in the summary window lists the keywords of the message within braces at the beginning of the Subject or Text field.

Each Babyl file has a list of keywords, which you can set with the profile editor (see section 18.4, page 61). The purpose of putting keywords in this list is so that they will appear in the menu of possible keywords regardless of whether ZMail has noticed any messages using them.

12.1 Filter-Keyword Associations

You can tell ZMail rules for how to assign keywords; these are called *filter-keyword associations*. To create one, you must first define a filter, which is a named predicate that some messages will satisfy (see section 15.2, page 50). For example, there could be a filter named bug-zmail-recipient which is satisfied by any message which has BUG-ZMail as a recipient. Then you associate the filter with the desired set of keywords. Both of these operations can be done with the Filters box in the profile editor (see section 18.2, page 59).

Having established the rules, you tell ZMail to apply them to the selected message by typing K or clicking middle on Keywords. ZMail looks through the list of associations and applies each filter to the message. Each time the message satisfies a filter, the associated keywords are given to the message. Any other keywords the massage already had remain on.

13. Moving Messages

When you work with more than one mail file, you need to move messages from one file to another. Some users divide move most of their mail out of their primary mail files and into other mail files according to their topics. The ZMail command **Move to File** is used for this, and for some related operations.

The simplest way to move a message to another mail file is to type O (for Output). ZMail reads the filename from the keyboard, reads in or creates the file if necessary, and moves the selected message to that file's buffer. The default filename is the last name you used, so if you just type **Return**, you will move to the same file as last time.

You can also move to the default place by clicking left on **Move to File** in the command menu.

Clicking right on **Move to File** gets you a menu of places to move to. These include all the buffers in ZMail and six special items, which appear in italics.

By Filters
By Individual Filters

Figure out which mail file to move to by matching the message against the mail-file-filter associations (see below). *By Filters* appears when you are moving a single message, and *By Individual Filters* appears when you are moving several messages, such as with **Map Over** of **Move**, to emphasize that each message will be match individually against the filters to determine where to put that message.

New Temporary
Generated Temporary

Create a new temporary buffer and move the message into it. *New Temporary* asks for a name from the keyboard, while *Generated Temporary* uses the name Temp, or Temp-1 if there is already a Temp, and so on.

The next time you click right on **Move to File**, the temporary buffer will have its own item in the menu. You do not use *New Temporary* or *Generated Temporary* to move a second message into the same buffer.

Find File Create a new file buffer, reading in the mail file if it exists, and move the message into it. The filename is read from the keyboard with the minibuffer.

Text Mail File Create a new write-only text mail file (see below), and move the message into it.

Hardcopy "Moving the message to hardcopy" is how you print the message on a printer. Click left to use the default printer and default options (you can change the defaults in the profile editor to save them in your init file); click right to specify them with a menu. The available hardcopy options are documented under profile editing; see section 18.6, page 62. Any changes made to the hardcopy options remain in effect for the rest of the session.

Abort Cancels the **Move to File** operation. This is the only way to get out of the menu.

Whichever you specify with click right becomes the new default for click left.

If you follow the practice of moving much of your mail into non-primary mail files, it is convenient to have those files appear in the Move to File-right menu even if you have not read them in yet. Your profile can contain a list of files to include in the menu, which you can edit with the Mail Files box in the profile editor (see section 18.3, page 60).

default-move-mail-file-name Default filename for files to move messages to. *Variable*
The value specified for this variable in your profile provides an initial meaning for click left on Move to File to use. The value is also used as the default pathname when ZMail reads the name of a file to move to. Every time you specify a file buffer to move to, this variable is updated.

move-file-name-sticky-fn2 Take filename type from default. *Variable*
Take file type for moving to a new file from default. In other words, if the default is FOO.BAR and you type FOXX, the file will be FOXX.BAR. Yes or No, with Yes as the default.

A message has only one owning mail file buffer, which cannot be changed. When you move a message into a temporary buffer, the same message appears in that buffer in addition to whatever other buffers it belongs to. But when you "move" a message into a file buffer, including text mail files, it actually works by copying the message. Normally, the original is deleted in this case (but it does not actually disappear until expunged).

delete-after-move-to-file Delete message when moved to file. *Variable*
Yes or No, with Yes as the default. If Yes, moving a message into another mail file buffer, which actually copies the message, marks the original as deleted. If no, the original is not deleted.

13.1 Mail-File—Filter Associations

It is also useful to tell ZMail about rules for how to decide where to move a message. These rules are called *mail-file-filter associations*, made up of pairs of one mail file and one filter, and they are set up using the Mail Files or Filters box in the profile editor (see section 18.2, page 59). The rules are used only when you tell ZMail to use them. You do this by selecting *By Filters* or *By Individual Filters* in the Move to File menu, or by clicking middle on Move to File. Then ZMail takes each message to be moved and matches it against the filters in the associations, one by one, until a filter matches the message. The mail file associated with that filter is the one the message is moved to.

For example, you might define a filter about-zmail which matches messages which have "zmail" in their subjects, and associate this with the mail file ZMAIL-BUGS.XMAIL. Then Map Over of Move to File of *By Individual Filters* would (among other things) move all such messages into that file.

When you are moving multiple messages, messages that do not match any association filter are not moved. Messages are normally deleted when moved into another mail file buffer (see the preceding section), and in this case the messages actually moved are deleted while those not

moved because they matched no filter are not deleted. A useful technique for dividing up all new mail automatically is to get new mail, make a subset buffer of unseen messages (see page 49), then do Map Over of Move to File of *By Individual Filters*.

13.2 Text Mail Files

A *text mail file* is a file of messages written to be read directly by humans.

Other formats of mail file are designed to be used as data bases by ZMail and other mail editing programs. They contain special delimiters to enable the programs to parse out the messages reliably no matter what their contents. Text mail files are designed to look good when printed or typed on a terminal. Messages are separated only by blank lines, or some other string you specify. As a result, it is impossible to parse one once it has been written. Text mail files are produced as output from ZMail but cannot be used as input.

The only way to create a text mail file buffer is in the Move to File command, since there is no other way for it to have any messages. It always starts out empty except for the messages you move in.

The text mail file buffer is just like any other buffer as long as you are in ZMail. You can select the buffer, operate on the messages, delete them, reply to them. You can even change the mail file's format, using the File Options box in the profile editor. But if you save the file in text mail file format, that file cannot be read back in by ZMail. Indeed, ZMail does not even look for an existing file when it creates a text mail file buffer. The buffer starts out empty, and saving it will produce a new file version.

text-mail-file-separator Line between messages in text mail file. *Variable*
This string supplies the contents of the lines written between each pair of messages in a text mail file. The default is the empty string, making the line that separates messages a blank line.

14. References Between Messages

Some messages refer to other messages with In-reply-to or References fields. You can add an In-reply-to field to an outgoing message with the editing command Meta-X Add In Reply To Field, and ZMail provides a profile option (*add-in-reply-to-fields*) for adding them automatically to all outgoing messages. You can also insert a References field, either manually or with Meta-X Add References Field (page 32). If other users use ZMail for their replies, or use some other program which does likewise, it is possible to trace out all the messages that make up one conversation by looking at the inter-message references. ZMail provides commands to do this.

The extended command X Select Referenced Msg finds the message referenced by the selected message and selects that one. X Delete Referenced Msgs deletes all messages referenced by the selected message.

The extended command X Select References creates and selects a temporary buffer containing all the messages referenced directly or indirectly by the selected message. This means the messages that the selected message refers to, and the messages they refer to, and so on.

To trace references both forward and backward, use X Select Conversation by References. This selects a temporary buffer containing all the messages connected by references *in either direction* with the selected message. Messages that reference the selected message, directly or indirectly, are included as well as those referenced by it. The connection between the selected message and the other message need not always run in the same direction; a message that references the same one that the selected message references will be included. There is also X Delete Conversation by References, which deletes the same set of messages that the other command would select.

X Append to Referenced Msg appends the selected message into the text of the message it references, then deletes the selected message. The result is that the two related messages are textually tied together.,

X Move In Place of Referenced Msg moves the selected message to occupy the place of the message it references. It is moved as in the Move command, into that message's file, at a position just before it. Then that other message is deleted.

When searching for the referenced message, ZMail first tries the selected mail file (a numeric argument to any of these commands inhibits this). Next it uses the filter-universe associations to try to find a universe: it applies each filter in turn to the original message, and if a filter matches, the associated universe is searched. If nothing is found this way, the user is asked to specify a universe to search. The filter-universe associations are set using the Filters box or the Universes box in the profile editor (see section 18.2, page 59).

15. Message Predicates

A *message predicate* is a way of distinguishing some messages from the rest according to their contents (headers and text). Message predicates can test such things as whether a message has been read, deleted, replied to, or moved to another file; for a particular sender or recipient, a particular subject, or a particular keyword.

The normal command menu contains three commands that work with message predicates. Each command uses both a predicate and a *universe*. A universe is a buffer or a union or intersection of buffers (see universes, section 15.4, page 52). The predicate is used to select some of the messages from the universe. For example, you might choose as the predicate "messages with RMS as a recipient" and as the universe "either FOO.BABYL or Temp-2"; then you will get all messages in either FOO.BABYL or the temporary buffer Temp-2 which have RMS as a recipient. The messages that match are the ones that any of these three commands will operate on. The commands differ in what they *do* to the messages thus found.

Survey Print a list, in the summary window, of all the messages matching a predicate.

Jump Select, one by one, the messages matching a predicate. Click right on **Jump** to specify a predicate; then you will move to the next message in the selected buffer that satisfies the predicate (you can also specify a universe to move through; but the default is "the rest of the selected buffer"). Once you have done this, you can continue moving through the same set of messages by clicking left on **Jump**. This uses the same predicate and universe as last time.

Select File

Click middle on **Select File** to create and select a *subset buffer* consisting of all messages which match the predicate; then you can use the **Next** and **Previous** commands to move around through them, or use **Map Over** to operate on all of them. The subset buffer is given a name that looks like [*universe*]<*predicate*>, as in [RMS.BABYL]<deleted>.

15.1 Built-in Message Predicates

Whenever you are asked to choose a message predicate, no matter what it will be used for, the choice is always made with the *predicate selection window* (flavor **zwei:filter-selection-frame**). Here are the predicates you can choose from:

All Checks nothing at all; this is the trivial predicate, which every message satisfies.

These predicates check for message attributes:

Deleted Checks for a deleted message.

Unseen Checks for a message that has never been selected in ZMail ("seen" in full by the user).

Recent Checks for a message that is "recent"; that is, was obtained as new mail since the last time its mail file was expunged.

Answered Checks for a message that you have replied to using the **Reply** command.

Filed Checks for a message that you have moved into another buffer using the **Move to File** command.

These check the headers or the text:

Search Checks for a message whose text contains a given string. You are asked to specify the string to search for with the minibuffer.

From/To Checks for a message that has a specified address as its sender or as a recipient. You are asked to specify the address to look for with the minibuffer.

Subject Checks for a message whose subject contains a specified substring. You are asked to specify the string to search for with the minibuffer; alternatively, you can click on a message in the summary window and its subject will be used as the string to check for.

These check the message keywords:

a keyword Checks for a message that has this keyword.

Any Checks for a message that has any keywords at all.

These can check for derived predicates or combinations of things.

a filter Each defined filter (see section 15.2, page 50) appears in the predicate selection window. Click on a filter name to use that filter.

New Filter Click on this to define a new filter and use that one (unless you abort).

Not Negates the condition specified by whatever else you choose. Clicking on **Not** does not specify a predicate and does not exit the selection window. Each click on **Not** turns it on or off, as you can see by the highlighting of the box. When you select one of the other predicates, if **Not** is on at the time, the meaning of the selected predicate is inverted.

15.2 Filters

A ZMail *filter* is a message predicate that you define in terms of other predicates. Once defined, these user-defined predicates can be used just like the built-in ones in the **Survey**, **Jump** and **Select File** commands. In addition, filters can be associated with mail files, keywords or universes. These associations enable ZMail to figure out automatically which mail file, universe or keywords to use in certain commands by checking the associated filters against the message to be operated on. Associations are set up with the profile editor (see section 18.2, page 59). Filter-keyword associations provide rules for the **Keywords** command (see section 12.1, page 44). Filter-universe associations provide rules for the conversation commands (see page 48). Filter-mail file associations provide rules for the **Move to File** command (see section 13.1, page 46).

You use a filter as a predicate by clicking on its name in the predicate selection window. When editing the associations of filters, you specify a filter with a menu. Both situations provide a box to click on to define a new filter instead of using one of the listed existing filters.

15.3 Defining Filters

You can ask to define a new filter almost any time you are supposed to choose a filter to use. What happens then is the same no matter what you were going to use the filter for. The window configuration changes to present the filter definition frame (flavor zwei:zmail-filter-frame). This frame contains an editor window that displays the definition as written so far, as Lisp code, and boxes that you can use to specify the definition. You can edit the Lisp code directly if you know what the code should look like. You can also create the filter definition by clicking on the other boxes in the frame to specify various primitive predicates and logical operations (And, Or and Not) for combining them. As you click on the boxes, the code displayed in the editor window changes automatically to show the effect of what you have done.

Above the editor window is a box that is used to specify the name of the filter to be defined. Initially the name is something like Noname-1; click on the box to specify a different name. When you have done so, the name will appear both in the box and in the editor window (in the proper place in the definition).

The remaining boxes are on the left hand side. The bottom two rows are for specifying primitive predicates. The top row of boxes are for specifying combinations of primitive predicates. The second row is used for testing the filter and for exiting filter definition.

Here are the boxes that specify primitive predicates:

Deleted, Unseen, Recent, Answered, Filed, Subject
: These are the same as the primitive predicates described in the preceding section.

To
: Checks for a message whose To recipients include a particular one. You are asked to specify with the minibuffer the recipient to check for.

To/CC
: Checks for a message whose recipients of all kinds include a particular one. This is a more general condition than the preceding one; it is always true if that one is. You are asked to specify with the minibuffer the recipient to check for.

From
: Checks for a message from a certain sender. You are asked to specify with the minibuffer the sender to check for.

Subject
: Checks for a message whose subject includes a particular string, which you specify in the minibuffer.

Other
: Checks for a message containing a particular string within a particular header field. You specify first the header field name and then the string, with the minibuffer. If the header field is one which contains recipients, such as To or From, the specified string is looked for as one of the recipients. Otherwise it is looked for as a textual substring.

Before
: Checks for a message dated before a date which you specify.

On
: Checks for a message dated on a date which you specify.

After
: Checks for a message dated after a date which you specify.

Keywords
: The box labeled "keywords" has an item for each keyword that appears on any message, and another item named *Any* (in italics). *Any* checks for a message that has any keywords. The item for a particular keyword checks for a message that

has that particular keyword.

Filters The box labeled "Filters" has an item for each filter already defined. The item specifies a predicate that checks for a message which satisfies the predefined filter. Thus, one filter can be used as a subroutine of another filter definition.

A simple filter definition can be completed simply by selecting one primitive predicate with the boxes listed above. A more complex filter definition would involve combinations of primitive predicates using **And**, **Or** and **Not**. These definitions are written using the boxes labeled thus in the first row. To specify a conjunction of predicates, first click on **And**, then click on the predicates in the conjunction, and finally click on **Close** to finish the conjunction. **Or** and **Not** are done the same way (but you may only have one predicate inside the **Not**). Clicking on **And**, **Or** or **Not** inserts a pair of parentheses, which you can see in the editor window, and positions point between them. **Close** moves point over the close parenthesis. Thus, you can always see in the editor window the results of any complicated structures you have specified with the boxes.

You can test the definition you have made so far by clicking on **Sample**. This finds all the messages in the selected buffer that match the definition and displays them in the summary window that appears at the top of the screen.

If you are satisfied, click on **Done** to actually define the filter and return to your previous activity. Click on **Abort** to return there without defining a filter.

Filter definitions can be saved in your ZMail init file, but this does not happen automatically. You must use the profile editor to specify which filters should be saved. See section 18.2, page 59.

15.4 Universes

A *universe* is a specification of a space of messages to search through. Any single buffer can be used as a universe; so can a union or intersection of buffers or other universes. A universe can also be defined as the complement of a buffer or universe; that is, all messages *not* in that buffer or universe. There are also a few built-in special universes:

Selected Buffer
This universe refers to whichever buffer that is selected at the time the universe is used.

Rest of Selected Buffer
This universe refers to all of the selected buffer following the selected message. This is the default for the **Jump** command, which is why repeated use goes through all the messages in the buffer that satisfy the chosen predicate.

Beginning of Selected Buffer
This universe refers to all of the selected buffer before the selected message.

Loaded Mail Files
All messages actually loaded into ZMail as of the moment the universe is used. Recall that ZMail allows you to work with a file before the whole file is read in. This universe specifically ignores the messages that have not been read in yet.

All All messages in all mail files known to ZMail. If not all the mail files are loaded, use of this universe will wait until loading is complete.

A universe is usually used together with a message predicate by some sort of multiple-message operation. The operation will act on all the messages in the given universe which fit the given predicate. For these applications, you will specify a universe with the predicate selection window.

If you click on the box that asks for a universe, you get a menu that includes each of the built-in special universes, each of the user-defined universes, and each ZMail buffer (since they can serve as universes). There is also an additional item for defining a new universe. If you click on this, the universe definition window appears.

The universe definition window contains menus listing some built-in special universes, all the existing user-defined universes, and all the ZMail buffers. There are also boxes for the set operations **Union**, **Intersection** and **Not**. You use these to specify a combination of the existing universes; they work like **And**, **Or** and **Not** in filter definition mode. **Not** of a universe expression means all loaded messages that would not be in that universe. You can see the Lisp code for the universe definition accumulate in the editor window as you click on the boxes; you can also click on that window and edit it directly.

Click on the box above the editor window to specify a name for the universe, and finally click on **Done** to define it or **Abort** to exit without defining a universe.

Universe definitions can be saved in your ZMail init file, but this does not happen automatically. You must use the profile editor to specify which universes should be saved. See section 18.3, page 60.

One other use of universes is in telling certain searching commands the bounds of where to search. See page 48. You can specify the universe manually, or give ZMail rules in advance from which it can figure out automatically which universe to use. Such rules are called the filter-universe associations, and you specify them in the profile editor (see section 18.2, page 59).

16. Map Over

The Map Over menu command allows you to perform an operation on all the messages in the selected buffer at once. For example, you can delete them all, add keywords to them all, or forward them all (at once, in a single message). Map Over is most useful together with temporary buffers, when you create a temporary buffer so that you can map over it. The various ways of creating temporary buffers thus become ways of specifying which messages to delete, forward, etc.

Clicking right on Map Over brings up a menu of operations that can be mapped. The alternatives are: Delete, Undelete, Type, Find String, Keywords (that is, add keywords to a message), Unkeywords (that is, remove keywords), Move to buffer, Forward, Redistribute, Reply, and Concatenate.

Type means that all the messages are printed in succession. Find String means that you type a string in the minibuffer and each message is searched for that string; messages which contain it are listed in the typeout window, and you can click on a line to select that message. Concatenate means that all the messages are stuck together into a single message. The other options do the same thing as the like-named top level ZMail commands.

Mapping the Forward command over multiple messages creates one draft only, which contains all of the messages to be forwarded. Mapping the Reply command, however, starts one draft for each message to be replied to.

Clicking left on Map Over repeats the last map operation. Clicking middle performs one of the map operations described above; a profile option specifies which one:

map-middle-mode Middle button on Map command. *Variable*
This controls what happens when you click the middle button on the Map menu item. The alternatives are the same as those in the menu of mapping operations.

16.1 Experimental Window Configuration

The experimental window configuration allows you to apply any of the commands in the command menu to a set of messages instead of a single message. You get into the experimental configuration by clicking right on Configure in the command menu and then selecting Experimental in the menu that follows; you can also make it the default with a profile option (see chapter 17, page 56).

In the experimental configuration, the command menu is split by a pane containing two boxes, one specifying a universe and the other a predicate. By clicking on the boxes, you can specify a different universe or predicate; these then apply to the next command you click on in the command menu. The commands for which the universe and filter are relevant are those in the part of the command menu below the boxes.

The box for specifying a universe appears on the left and normally says **Just current message**. If you click right on this, you get a menu of built-in and user-defined universes, with which you can select one. The box will change to display the name of the universe you have selected.

Similarly, the predicate selection box normally says **All**, which means that the predicate used by default is a trivial one that accepts all messages in the specified universe. If you click right on the box, you get to the predicate selection window, in which you can select a built-in message predicate or a user-defined filter, or define and use a new filter. The box will change to display an indication of the predicate or filter you chose.

Thus, to delete all the messages in all loaded files, click right on the universe box and select **All** in the menu, then click on **Delete** in the command menu.

The boxes always display the name of the universe and filter that will be used for the next command. Normally the universe box says **Just current message** and the filter box says **All**, informing you that commands will apply to the selected message alone. If you choose a different universe or filter, the box changes. After the next command, the boxes revert back to their normal states, so specifying a universe or predicate applies only to one command. However, the last universe you specified is recorded and you can specify it again just by clicking the middle button on the universe box; similarly, the last predicate you specified is recorded and you can specify it again by clicking middle on the predicate box.

17. Window Configurations

There are four window configurations you can use for normal mail-reading with ZMail. The normal default is called Both; it displays both the summary window and the message window, with the command menu in between. Two other configurations are Message only and Summary only; these eliminate either the summary window or the message window (The command menu and mode line are always present in top-level configurations). The remaining configuration, Experimental, presents the same set of windows as Both but provides additional features relating to universes and filters (see section 16.1, page 54).

You can switch to a different configuration at any time while you are at top level by clicking on Configure. Clicking left selects the Both configuration. Clicking right gets a menu of all four configurations. In addition, the following profile option allows you to specify which configuration ZMail should start up in.

default-initial-window-configuration Default startup window setup. *Variable*
Controls the window configuration used initially in normal mail reading mode. The alternatives are Both, Summary only, Message only, and Experimental. The default is Both.

(ZMail also has other window configurations, such as the one used for editing the profile and those used for composing mail, but they are used automatically by the commands they are intended for and are not alternatives at this level.)

18. Editing Your Profile

ZMail has many flags and parameters for the user to set. The ZMail *profile editor* makes it easy for you to find out what options are available, set them, and store the settings in your "ZMail init file" so that they will be in effect every time you use ZMail.

To enter the profile editor, click on the menu item **Profile**. The window configuration will change to display a list of option names and values at the top, and the text of your ZMail init file at the bottom. (There will also be several boxes with words in them, which you can click on to enter other aspects of the profile editor).

The list of options displays one option per line, together with its current setting or the possible alternatives. If the option has a fixed set of alternatives, they are all displayed, with the selected alternative in boldface. To select an alternative, click on it with the left mouse button. If the option allows an arbitrary number, string or filename, the current setting is displayed. To change it, click on it with the left mouse button. A cursor will appear instead of the current setting, and you can then type in the new setting. End it with **Return**.

There are too many options to fit on the screen at once, so you can scroll through the list, either with the scroll bar by moving the mouse off the left of the list, or by pushing the mouse against the top or bottom of the list where it says **more above** or **more below**.

You can also edit the text form of the information by clicking on the window containing the text with the left mouse button. You then use the ordinary editor commands to do your editing. To exit, type **End**.

The list of option settings and the text are duplicate representations of the same information. You can edit either one, and the other will be changed to match (eventually, not instantaneously). The list of options at the top is easiest for a beginning user to edit, because it shows you what options are available and what kinds of values they can have. Editing the text form is useful only for more sophisticated tasks. The text form is maintained because that is the way your profile is stored in the init file.

18.1 The Bottom Row of Boxes

The five boxes underneath the displayed list of options are used to control the profile editor.

The **Done** box exits the profile editor. This does not save your init file on disk; use the **Save** box for that. However, you can go back to using ZMail with the option settings you have established. If you decide you like them, you can reenter the profile editor later and save your init file on disk then.

The **Reset** box resets all ZMail user options to the values they have in your init file on disk. You are offered the choice of reading the text back in from the disk file as well, or updating it as is usually done when option values are changed.

The **Defaults** box resets all ZMail options to their system default values—what they would be if you had no init file. You are asked whether to update the textual form as well. If you say Yes, it is updated by deleting everything that sets an option parameter.

The **Edit** box selects the textual form of the init file for editing with all the usual editor commands. This is the same as clicking the mouse on the window where the text is displayed. The text is updated to match your latest option settings before you begin editing it. You can exit with **End**, whereupon the text as you have edited it will be read and the options list will be updated according to the changes you have made.

The **Save** box controls the updating of the textual form of the profile and of the init file on disk.

Clicking left on the **Save** box is a simple way to update them both. The text is updated to correspond to the option settings that you now have, and then saved on disk in your init file. Finally, if you have created a compiled init file, that too is updated.

Clicking Right on the **Save** box gets you a menu containing operations that let you control the updating more carefully. The menu items are **Save file**, **Make init file compiled**, **Insert changes**, **Reap file**, and **Recompile file**.

Insert changes updates the text version of the profile, as stored inside ZMail and displayed in the bottom half of the screen. It does not change any disk files.

Save file writes the text version of the profile, as displayed on the screen, into your init file on disk. It does not update the text first, and it does not update the compiled version of your init file, just the text version.

Recompile file recompiles your init file, updating the compiled version. Use this only if you already have a compiled init file.

Clicking left on the **Save** box is equivalent to doing **Insert changes**, **Save file** and then perhaps **Recompile file**.

Make init file compiled compiles the text of your init file. This is useful if the text is long and contains lots of filter definitions, because it will load faster if compiled. It does not make ZMail run faster once the file is loaded; for example, filter definitions are always compiled no matter how they are input. For simple init files, there is no advantage in compilation.

When you have a compiled init file, the actual init file which ZMail loads is the compiled file. The source is stored in a different file. When you perform the **Make init file compiled** operation, you are asked for the filename to use for the non-compiled version.

Reap file offers to delete old versions of the text of your init file.

18.2 The Filters Box

ZMail can record filter definitions in your init file so that they are available in future sessions. It can also record associations between filters and mail files. The **Filters** box is used to specify which filter definitions and associations should be recorded permanently.

If you click left on the **Filters** box, you get a multiple menu which lists all the filters you have defined. Those which are recorded in the init file are highlighted.

You can change which filters are recorded by clicking on the filters you wish to change. Clicking on a highlighted filter makes it unhighlighted, and vice versa. To finish, click on the **Do It** choice box at the bottom of the menu. The text in the profile editor window does not actually change until this time, and until then you can cancel your changes by clicking on the **Abort** choice box instead.

The remaining choice box, **New Filter**, allows you to define new filters while in the middle of this. Clicking on this choice box puts you into filter definition mode (see section 15.3, page 51). When you exit that mode, having possibly defined a new filter, you are back in the menu.

Click middle on the **Filters** box to edit the associations of a filter. First you are presented with a menu listing all filter names. Use this to specify which filter's associations to edit. The menu also contains the item **New**, which you can use to define a new filter and then return to the menu.

After selecting a filter, you receive another menu with which you specify what kind of associations to edit. This menu contains three items: **Keywords**, **Mail files**, and **Universes**. Filter-keyword associations provide rules for the **Keywords** command (see section 12.1, page 44). Filter-universe associations provide rules for the conversation commands (see page 48). Filter-mail file associations provide rules for the **Move to File** command (see section 13.1, page 46).

If you click on **Keywords**, you then get to edit the keyword associations of the filter. A filter can have any number of associated keywords. You get a multiple highlighting menu listing all known keywords, where keywords associated with this filter are highlighted. Click on a keyword to add it to the list or remove it. Use the **Do It** choice box to finalize the changes, or **Abort** to cancel them. The **New Keyword** choice box allows you to define new keywords on the spot and add them to the menu.

If you click on **Mail files**, you then get to edit the mail file association of the filter. The filter can have only one associated mail file, so you get an ordinary menu listing all known mail files, **New** and **None**. Click on the one you wish to select. **None** means leave the filter with no associated mail file, and **New** lets you specify a mail file not already known. If you decide not to change the association, move the mouse away from the menu to abort it.

If you click on **Universes**, you then get to edit the universe association of the filter. The filter can have only one associated universe. so this menu works just like the one for mail files.

18.2.1 Pitfalls of Associations

Any changes you make to the associations of a filter are recorded in the init file, if you save the init file in that session. Associations record filters by their names. If you do not direct ZMail to record the associated filters in your init file, and fail to define appropriately named filters in your next session, you will probably get Lisp errors when ZMail tries to use the missing ones.

For the case of filter-universe associations, the same caveat applies to the universes.

It is possible to associate a filter with a temporary buffer instead of a mail file. If you do so, and you save the init file later in the session, the init file will record the name of the temporary buffer. In a later session, if you make use of the mail-file-filter associations, you must have a buffer by that name to avoid difficulties.

18.3 The Universes and Mail Files Boxes

The Universes box does for universes what the Filters box does for filters. Clicking left, you can specify which of your defined universes should be recorded in your init file. Clicking middle, you can alter the associations between filters and universes.

Clicking left on the Universes box gives you a multiple highlighting menu listing all defined universes. The ones recorded in your init file are highlighted. Click on a universe to add it to the list or remove it. Finally, click on the Do It choice box to finalize your changes, or click on the Abort choice box to cancel them. The New Universe choice box allows you to define a new universe on the spot and go back to the menu.

Clicking middle on the Universes box gives you a menu of universes, to select the one whose associations will be edited. There is also an item New, which allows you to define a new universe and proceed to edit its associations. The only associations a universe has are filters. Once you select a universe, you get a multiple highlighting menu listing all filters, with the ones associated with the given universe highlighted. Click on a filter to add it to the list or remove it. Finally, click on the Do It choice box to finalize your changes, or click on the Abort choice box to cancel them. The New Filter choice box allows you to define a new universe on the spot and go back to the menu of associations.

Be sure to read section 18.2.1, page 60 before using associations.

The associations between filters and universes are bidirectional. It does not matter whether you select a filter and change its associated universe, or select a universe and change its associated filters; the same set of bidirectional associations is being edited in either case. There is one asymmetry, however: a filter can have only one associated universe, but a universe can have any number of associated filters. If you associate a filter with one universe, no matter how, you automatically remove any association between that filter and any other universes. For the uses of filter-universe associations, see page 48.

The Mail Files box does for mail files what the Filters box does for filters. It works just like the Universes box. Clicking left, you can specify which mail files should be recorded in your init file. Clicking middle, you can alter the associations between filters and mail files.

The associations between filters and mail files are bidirectional, just like those between filters and universes. They provide defaulting rules for the **Move to File** command (see section 13.1, page 46).

18.4 The Keywords Box

The **Keywords** box lets you edit the list of keywords stored in a mail file or the associations between keywords and filters or. Click left to do the former or middle for the latter.

Clicking left on the **Keywords** box allows you to edit the lists of keywords stored in Babyl format mail files. These lists exist to make keywords appear in the menu for the **Keywords** command so you can conveniently add them to messages. An editor window appears containing a list of Babyl files and their keywords. You can edit this list, and finish by typing **End** to make your changes take effect or **Abort** to cancel them.

Editing the keyword-filter associations by clicking middle on **Keywords** works just like editing the universe-filter associations by clicking middle on **Universes**. You get a menu of all known keywords (including **New**, which lets you specify a keyword not previously known), and then a multiple highlighting menu with which you select the filters to associate with the keyword.

Keyword-filter associations are bidirectional, like mail file-filter and universe-filter associations, and you can use either the **Filters** box or the **Keywords** box to change them. Unlike universes and mail files, more than one keyword may be associated with a single filter. These associations provide defaulting rules for the **Keywords** command (see section 12.1, page 44).

Be sure to read section 18.2.1, page 60 before using associations.

18.5 The File Options Box

The **File Options** box lets you change the parameters of any mail file. You can change the mail file from one format to another this way, or change which end new messages are added at. In addition, some formats of mail file have additional subsidiary options that you can set.

You first get a menu listing all known mail files. Click on the mail file whose options you wish to edit.

Then you get a window displaying the options for the mail file. You get the same sort of window every time you create a mail file, so that you can specify its format and options. This window is like the one which displays the ZMail user options; it contains one option per line, together with its current value or the possible alternatives. If the current value is displayed, click left on it to change it; if alternatives are displayed, the current alternative is in bold face, and click left on an alternative to select it.

The first option listed is the format of the mail file. Each file server provides its own set of formats; some formats exist on more than one type of server, while others belong to one particular operating system. For example, Twenex file servers offer the formats **Tenex**, **Babyl** and **Text**. Each mail file format has its own set of options, and if you change format, the set options listed will change.

The available mail file formats are **RMAIL**, **Babyl**, **Tenex**, **Unix** and **Text**. The Babyl format is a flexible format that provides the most features. The Tenex and Unix formats are the standard formats of those operating systems. The RMAIL format is the standard format of ITS. The Text format contains no special formatting information, and can only be used for output from ZMail, because such files cannot be parsed back into messages.

VMS and Multics users should note that ZMail is not presently capable of reading the mail files found on those systems.

A mail file must be read in in the proper format in order for ZMail to parse it properly. When ZMail reads in a mail file, it normally determines the format of the file automatically, based on knowing which possibilities to expect on a particular file server, plus some heuristics. For example, on a Twenex file server, the Tenex and Babyl formats are the only possibilities expected (Text format files cannot be read), and there is a special program in ZMail to decide between those two. If you have copied a Unix mail file onto the Twenex and want to read it from there, ZMail will read it wrong.

Changing the format with the **File Options** box determines how ZMail will *save* the file; it is available only after the file has been read into a buffer, and that is too late to be useful in reading the file properly. For this purpose, the command **X Select Arbitrary Format Mail File** is provided. This command asks for a mail file format as well as a filename, and then reads the file parsing for that format. For example, you could read in the Unix format file this way, specifying Unix as the format. Later you could use **File Options** to change the format to Tenex or Babyl, and save the file in that format.

Strictly speaking, the parameters of a mail file are not part of your profile, because they often affect the way the mail file is formatted, and in any case affect all users who edit it, not just you. But setting mail file options and altering your profile are similar sorts of operations, and that is why the **File Options** box is found in the profile editor.

18.6 The Hardcopy Box

The **Hardcopy** box lets you edit the options for printing hardcopy from messages.

Clicking on the **Hardcopy** box gets you a list of the hardcopy options, together with the alternatives for them. This list works just like the regular ZMail options list: click on an alternative to select it, and the selected alternative is in displayed bold face; if any number or string is allowed, the current setting is displayed, and you change it by clicking on the current value and then typing the new value. Finally, click on **Do It** to make the changes take effect, or **Abort** to cancel them.

Unlike the main ZMail options list, the hardcopy options list is short and is all displayed at once. There is no scrolling.

The options are:

Output Device The only device available at MIT is **Dover**.

Font For output devices which can print more than one font, this specifies the font to use. The default at MIT is **LPT8**.

Number of copies

This is the number of copies to print. The default is **1**.

Spool through MC

This is relevant to Dover output only. Alternatives are **Yes** and **No**. **No** means send directly to the Dover.

Include Summary

This says whether the hardcopy should include a summary of the messages in it. The alternatives are **Yes**, **No** and **Just Summary**.

Print each message on a separate page

The alternatives are **Yes** and **No**.

Line between messages

This is a string used to make the lines separating messages, if you specify that each message should not be a separate page. By default, this is the empty string, meaning that messages are separated by blank lines.

19. "Other" Commands

The Other item in the command menu is a way at getting at various other ZMail commands with the mouse.

Click right on Other gets a menu of "other" commands. Initially there are two commands on the "other" command list: X Whois and X View File. The former prints the verbose information about a specified user at a specified site. The latter displays the contents of a file, and lets you scroll through it. You can add any ZMail extended command to the "other" menu using the function zwei:add-other-command, as in

```
(zwei:add-other-command zwei:com-zmail-yank-msg)
```

This is a macro and its arguments are not evaluated.

Notice how the command name X Yank Msg becomes zwei:com-zmail-yank-msg. In general, change hyphens to spaces and add "zwei:com-zmail-" to the front, to get the symbol for nearly any ZMail command.

Click left on Other repeats the last Other command used.

20. Babyl File Format

Unless otherwise stated, an Uparrow *character* is to be read as Control-*character*, e.g. ↑L is a Control-L.

20.1 Versions

First, note that each Babyl file contains in its Babyl Options section the version for the Babyl file—each version change signals a change in the format of Babyl files. This file describes Version 5 format.

20.2 Overall Babyl File Structure

A Babyl file consists of a Babyl Options section followed by zero or more message sections. The Babyl Options section starts with the line `BABYL OPTIONS:`. Message sections start with Control-Underscore Control-L Return Linefeed. Each section ends with a Control-Underscore. (That is also the first character of the starter for the next section, if any.) Thus, a three message Babyl file looks like:

```
BABYL OPTIONS:
contents of the Babyl Options section.
↑_↑L
contents of the 1st message section
↑_↑L
contents of the 2nd message section
↑_↑L
contents of the last message section
↑_
```

Babyl is tolerant about some whitespace at the end of the file—the file may end with the final ↑_ or it may have some whitespace, e.g. a Return Linefeed, after it.

20.3 The Babyl Options Section

Each Babyl option is specified on one line (thus restricting string values these options can currently have). Values are either numbers or strings. The format is name, colon, and the value, with whitespace after the colon ignored, e.g.:

```
Owner: ECC
Append: 3
Mail: PS:<ECC>ECC-TEST.MAIL.1
```

Here, the Owner option is the string "ECC", the Append option is the number 3, and the Mail option is the string "PS:<ECC>ECC-TEST.MAIL.1".

If just an option appears with no colon or anything else following it on the line, that signifies a value of t for boolean options, 1 for numeric options. A missing option has a default value of 0 for most numeric options, or nil for most other options (boolean or otherwise).

Different mail-reading programs may have their own options set of options, so all programs should be prepared to ignore and preserve any options that they do not recognize.

These options can be set in ZMail with the File Options box in the profile editor (see section 18.5, page 61).

Mail The value is a string, the name of the inbox file(s) for this Babyl file, separated by commas if there are more than one filename in the string. This is converted by ZMail into a list of pathname objects.

Version The value is a number, the Babyl file format version.

Owner A string which is the name of the user who "owns" this file. If any other user tries to save the file, he will be asked to confirm. ZMail regards an empty string or missing option as nil, no owner.

Append The value is a number, either 0, 1, 2 or 3. The number is interpreted as a pair of bit-flags. The low bit is 1 if new mail should be added at the end of the file. The second bit is 1 if the inbox file should be reversed before it is added. ZMail presents this value to the user as two separate file options, the Append option and the Reverse-New-Mail option.

No Reformation The value is boolean, so this option either appears with no value (t) or does not appear (nil). t means that message headers should not be reformatted. This turns off ZMail's or Babyl's reformatting feature (see section 6.2, page 20). The default is nil.

Labels The value is a string, made up of names of keywords separated by commas. This represents what is referred to above as the list of keywords stored in a Babyl file (see section 18.4, page 61).

GMSGS-Host The value is a string, the name of the host to find a GMSGS server on when doing GMSGS to get system announcements for this file. The default is an empty string which is taken as meaning nil. This means to use the host the file is stored on, a reasonable thing to do if that host supports a GMSGS server. See section 4.1, page 15.

Summary-Window-Format The value is a Lisp object in printed form. If it is not nil, it is used instead of the value of *default-summary-template* when this file is displayed in the summary window. See that variable, page 25, for possible values (aside from nil, which in this context means "not specified by this file").

Sort The value is the window's sort predicate, a symbol. See sorting, chapter 11, page 41. The option appears in the file as a string naming the type of sort.

Delete-expired The value is a string, Yes or No, which is taken as meaning t or nil. t means delete expired messages automatically when expunging the file. (Just the option

name with no colon is also accepted as meaning t, and if the option is not present the value is nil).

Babyl uses certain other options that are not used by ZMail. They are described here for completeness.

XMail — The value is a filename, used by Babyl on Tenex/Twenex systems. When Babyl adds the mail file to the Babyl file, it will also add it to this XMail file if specified. Note that this XMail file is in standard Tenex/Twenex mail-file format, not Babyl file format. The default is no XMail file specified.

XMail Append — The value is a number, zero or one, used by Babyl on Tenex/Twenex systems. This controls how Babyl adds new mail to the XMail file, mentioned above. It is a boolean: one means append to the XMail file, zero means prepend. The default is to prepend.

No Original — The value is a number, zero or one, used by Babyl. A value of one specifies that when header reformation occurs, the original header should be discarded, to save space in the Babyl file. The default is to keep the original header.

20.4 Message Sections

A message section contains one message and information associated with it. The first line is the "status line", which contains a bit (0 or 1 character) saying whether the message has been reformatted yet, a list of the attributes of the message, and a list of keywords. Attributes are built-in properties with built-in-meanings. Some, such as Answered or Unseen, are assigned automatically as byproducts. Others, such as Deleted, appear in response to special commands. Keywords are user-defined and user-assigned (with the Keywords command). The entries in the status line are separated by commas, with an empty entry separating the attributes from the keywords. For example, consider

```
1, answered,, zval, bug,
```

The 1 means this message has been reformatted. The only attribute is Answered, and the keywords are zval and bug. A message with no attributes or labels would have a status line like

```
1,,
```

Or, if it had two attributes, Answered and Deleted, it would look like:

```
1, answered, deleted,, zval, bug,
```

After the status line comes the original header if the header has been reformatted for display. Following that is the EOOH line, which contains exactly the characters

```
*** EOOH ***
```

(which stands for "end of original header"). Note that the original header, if a network format header, includes the trailing CRLF. And finally, following the EOOH line is the visible message, header and text as it is displayed for the user by Babyl or ZMail. For example, here is a complete message section, starting with the message starter, and ending with the terminator:

```
↑_↑L
1,, wordab, eccmacs,
Date: 11 May 1982 21:40-EDT
From: Eugene C. Ciccarelli <ECC at MIT-AI>
Subject: notes
To: ECC at MIT-AI

*** EOOH ***
Date: Tuesday, 11 May 1982  21:40-EDT
From: Eugene C. Ciccarelli <ECC>
To:   ECC
Re:   notes

Remember to pickup check at cashier's office, and deposit it
soon.  Pay rent.
↑_
```

The header reformatting shown is that done by Babyl. Reformatting in ZMail is done by a user-defined mail template, if desired.

Concept Index

Variable Index

Function Index