Laurel Manual

by Douglas K. Brotz

CSL-81-6 May 1981

Abstract: Laurel is an Alto-based, display-oriented, computer mail system interface. It provides facilities to retrieve mail and present it for delivery, and to display, forward, classify, file, edit and print messages. Additional features include facilities to read, write and copy files, run programs, and a whole lot more. Laurel is a component of a distributed message system that has been in operation for several years in the Xerox Research Internet.

This document is a description of the facilities contained in Laurel. Several tips on proper use of computer mail facilities in a social context are included.

Key words and phrases: Alto, Distributed computing, Electronic mail, Ethernet, Grapevine, Laurel, Text editor, User interface.

XEROX
PALO ALTO RESEARCH CENTER
3333 Coyote Hill Road
Palo Alto, California 94304

Table of Contents

1. Introduction

1.1. What is Laurel?

Laurel is an Alto-based, display-oriented, computer mail system interface. It provides facilities to retrieve mail and present it for delivery, and to display, forward, classify, file, and print messages. Additional features include facilities to read, write and copy files, run programs, and a whole lot more. Laurel is a component of a distributed message system that has been in operation for several years in the Xerox Research Internet. Laurel executes on an Alto (or Dolphin or Dorado emulating an Alto) and uses the display, keyboard and mouse in a fashion befitting Alto-based software.

1.2. What is this manual?

This manual is a reference document for Laurel. The Laurel team believes that most of the basic facilities of Laurel are self-explanatory, and that you can use Laurel quite competently after reading only this introductory section and taking the interactive tutorial using Laurel itself (see *How to obtain Laurel* below). Laurel also has features that are not immediately obvious, and after becoming acquainted with the system, you will want to read about those facilities in this manual.

Sections 3, 4, and 5 describe features of Laurel in detail. On first reading, you may read the introductory portion of each of these sections, quickly skim the rest, and return to them later to gain understanding of their finer points. Section 6 describes proper message system behavior, and should be read by every message system user, new or old. Section 7 contains advice that is useful after gaining some familiarity with Laurel through the earlier sections. Sections 8 and 9 may be skipped on first reading as their titles imply. Appendices A, B, and C give detailed information on a variety of topics, and may be read later as their information becomes relevant to your usage of Laurel. Appendix D has important information if you currently use MSG. Appendix E is a short summary of the differences between Laurel 6 and previous versions of Laurel.

If you have the time, it is perfectly all right to read this manual from beginning to end. Many of the functions provided by Laurel are interdependent; extensive cross-indexing is used throughout this manual to avoid repeating information in several places.

The Laurel team will be very happy to hear any suggestions you may have, and is particularly interested in your experiences in using the system. Comments should be sent to LaurelSupport.PA (using the facilities provided by Laurel!)

The version of Laurel described in this manual is Laurel 6. It replaces all previous versions of Laurel (those with smaller numbers and LaurelX). Previous versions of Laurel are no longer supported. It is in your best interest to update now to Laurel 6, as previous versions may stop working in the future due to changes in the message transport system.

1.3. How to obtain Laurel

We strongly recommend that new users obtain a Non-Programmer's disk to be used primarily for processing and managing messages (see section 7 for the reasons behind this recommendation).

Two Alto command files are available on most file servers for obtaining Laurel. If you are a new user, you should issue the following commands to the Alto Executive:

```
>Ftp FileServer Retrieve <Laurel>LaurelNewUser.cm CR
>@LaurelNewUser CR
```

where *FileServer* should be replaced by your local file server's name. This obtains Laurel and starts it up in a tutorial mode. If you are a new user, you should do this now, before reading the rest of this manual. What follows will be much easier to assimilate once you have a passing acquaintance with Laurel. After you have finished the tutorial, please do read the rest of this manual, especially section 6.

If you are already familiar with Laurel, you should issue the following commands to the Alto Executive:

```
>Ftp FileServer Retrieve <Laurel>Laurel.cm CR
>@Laurel CR
```

which merely retrieves the files necessary to run Laurel.

Fine point: Laurel 6 is a Mesa 6 program set up as a run file. It requires no other parts of the Mesa environment on the local disk. This one version of Laurel 6 adapts itself to your machine; no special extended memory version is necessary.

1.4. How to run Laurel

To invoke Laurel, type

```
>Laurel CR
```

to the Alto Executive. This is the default method of invoking Laurel. Command line options are discussed in Appendix A.

To invoke the Laurel tutorial (after obtaining it with @LaurelNewUser), type

```
>Laurel Tutorial CR
```

1.5. Acknowledgments

The Laurel program and its associated mail transport facilities are the product of the combined efforts of many individuals. The members of the Laurel team have included at various times (in alphabetical order): Richard M. Barth, Douglas K. Brotz, James J. Horning, Robert Kierr, J. Timothy Korb, Roy Levin, Roger Needham, Michael D. Schroeder, Jay Spitzen, and Ben Wegbreit. Persons who have provided supporting facilities included in Laurel include Hal G. Murray, James E. White, and the entire Mesa Group. The mail transport facilities are due to Andrew D. Birrell, David R. Boggs, Stephen C. Butterfield, Roy Levin, Michael D. Schroeder, and Edward A. Taft. Inspiration and managerial support were provided by Jerome I. Elkind, Robert M. Metcalfe, and Robert W. Taylor.

For an interactive program to become polished and meet the needs of its user population, extensive testing and accumulation of user experience are all important. Special thanks are due to the dedicated Alpha-testers of Laurel throughout its development, whose collective suggestions and patient error tracking have helped to shape this system into its present form.

Last, but not least, none of this would have been possible without the existence of the Alto, the Ethernet, and, of course, the Xerox Corporation.

2. Basic notions and facilities

Laurel is a display-based, interactive program that manipulates a particular class of files, called *mail files*. In essence, a mail file is just a sequence of *messages*, each of which is a text string formatted according to certain conventions. The details of these conventions are not of major interest to most users suffice it to say that messages have a *header* and a *body*; a header contains (at least) a *sender*, a *subject*, one or more *recipients*, and a *date*. For each mail file, Laurel constructs and maintains a *table-of-contents* that summarizes the messages residing in the file. Mail files are private to each user, meaning that your mail files reside on your own Alto disk(s). Laurel provides facilities for receiving messages, manipulating mail files, examining and responding to messages, composing and sending new messages, and cataloging, filing, and printing messages.

Laurel is an interface program that allows access to the facilities provided by the *Message Transport System*. This system provides users with *in-boxes*, *mail drops* and a *name data base*. An *in-box* is a place where messages sent to a message system user are stored until that user requests those messages. The in-box resides on a mail server or file server. A *mail drop* is a service to which messages addressed to any user(s) of the message system may be sent. The Message Transport System sorts out the recipients and delivers the messages to their individual in-boxes. The *name data base* holds the names of registered users of the message system along with named lists of such users.

Each user’s message system name consists of two parts separated by a period. The second part is the *registry*; the first part is an identifier for a person within that registry. Registries exist to partition the number of names understood by the message system and to help the message system determine on which mail server the in-box for a particular user will be found. For example, in the actual message system name "LaurelSupport.PA", the registry is PA and the user name within that registry is LaurelSupport.

The Message Transport System is presently implemented by two distinct systems, MTP and Grapevine. For the most part, Laurel hides the differences between these two implementations so that you need not be aware of the particular system to which you are connected. However, there are some functions of the Message Transport System, particularly with regard to public distribution lists, that are different depending on which implementation is supported in your registry. These differences are noted in this manual.

MTP (for Message Transport Protocol) is the older system, and is currently used on IFS file servers and on Maxc. Grapevine (or GV) is the newer system, providing distributed functions throughout the Xerox Internet and more powerful operations to its clients. These two systems interface to each other so that users of either system are connected to the entire message system community. At the time of this writing, the PA and ES registries are served by Grapevine; all others are served by MTP. As time goes on, Grapevine will gradually replace MTP throughout the Xerox Internet.

2.1. The user interface

Laurel is a highly interactive system in which economy and clarity of expression are essential. In striving to provide a convenient user interface, Laurel borrows a number of conventions familiar to users of other Alto subsystems. This section describes the principles that underlie the Laurel user interface.

2.1.1. Command invocation

Most Laurel facilities are invoked by use of the mouse, with the keyboard being used almost exclusively for text entry. Commands are generally represented by words or phrases on the display screen. You invoke a screen command by moving the mouse until the cursor points at the desired command, then clicking a mouse button (usually RED). If you hold down RED and move the cursor to point at a command, the command will appear *inverted* (i.e., white letters on black background). When you release the button, the command name will appear *grayed* (i.e., black letters on a gray halftone background). If, while holding down the mouse button, you change your mind about the command you intend to select, simply move the cursor until the inverted name is restored to its normal state, then release the button. A fine point: position the cursor so that it *points at* the command, not so that it rests on top of it.

Normally, you use RED to invoke screen commands. Some commands have such a significant effect on the current state that you must confirm them explicitly, and in such cases Laurel will prompt you. If you are certain in advance, you may invoke the command by clicking BLUE. Laurel will then suppress the prompt for confirmation and execute the selected command immediately. Some commands require extra confirmation even when invoked with BLUE.

Laurel prompts you for confirmation by displaying the message
Type ESC to confirm, DEL to cancel command.
in the feedback region at the bottom of the screen. It also displays a large flashing question mark in the cursor. You may confirm by striking ESC, CR, Y, or space, or by clicking YELLOW. If you strike DEL or N, the command will be aborted. Ultra-fine point: if you change your mind after pressing YELLOW, but before releasing it, clicking RED will turn it into a DEL!

2.1.2. Editing conventions

Laurel supports two distinct editor styles, *modal* and *modeless*. By default, Laurel will use the modal editor; to change this you must place an entry in your Laurel profile (see section 5). The modal editor is similar to the Laurel 5.1 editor and to Bravo 7 and 8. The modeless editor is similar to the LaurelX editor (previously distributed on a limited basis) and to BravoX. When either editor allows you to supply text, it displays a blinking caret at the appropriate place on the screen. Striking BS or CTRL-A deletes the character to the left of the caret; CTRL-W deletes the word to the left of the caret.

Shifted selection is available as an alternative to typing in either editor. A shifted selection is made by holding down either SHIFT key on the keyboard, making a selection with the SHIFT key still down, and finally lifting up on the SHIFT key. At this point, the selected text will be copied to the point indicated by the blinking caret. Shifted selection may be used at any time during type-in, not just immediately after issuing an editor command.

In the modal editor, type-in must be terminated by typing ESC; in the modeless editor no explicit termination is required.

The Laurel editor is described in detail in section 4.

2.1.3. Brackets

Several Laurel commands require text arguments. For instance, commands that manipulate a file need to know the name of the file on which they are to act. Text arguments appear in **{***brackets***}** following the command name on the screen.

When you invoke such a command by clicking RED, Laurel prompts you to fill in the brackets by displaying a blinking caret within the brackets. If there is already text within the brackets, the caret follows the text. In addition to the type-in conventions described above (including shifted selection), the following rules apply: If you strike ESC, CR, TAB, or space, or click YELLOW on the original command, the caret disappears and the contents of the brackets remain unchanged. If you strike BS or CTRL-W the contents of the brackets will be edited as if you had just typed that text. If you type anything else, it replaces the complete text within the brackets. Type-in is terminated by ESC, CR, TAB, space, or YELLOW click on the original command. Striking DEL cancels the entire command. If the text is too long to be displayed in the brackets then the middle of the text is represented as ". . .".

If instead of typing to fill in brackets for several commands that require filename arguments, you click RED on the original command again, Laurel will search your Alto file directory for an appropriate filename. Holding RED down on the original command will cause Laurel to repeatedly fill in the brackets with successive appropriate filenames from your directory. The most recently used files will appear first. The meaning of "appropriate" varies with the particular command and is discussed in the section for each command.

If you invoke the command with BLUE rather than RED, Laurel executes the command immediately, using the text argument already contained within the brackets, with no further action on your part. This is true of all commands that have text arguments in brackets, with the exception of the **User** command.

2.2. The display

While reading this section, it will be useful to have Laurel running on on Alto so as to refer to its screen, or to have Figure 1, The Laurel Screen handy.

Laurel maintains four regions on the display screen. From top to bottom they are: the *table-of-contents region*, the *message display region*, the *composition region*, and the *feedback region*. The table-of-contents region holds a directory of the messages residing in the current mail file. The message display and composition regions are used to examine messages received and to compose messages to be sent, respectively. The feedback region displays various information (frequently error notifications) appropriate to particular Laurel commands. Each region will be discussed in detail in subsequent sections.

The display also has three *command menus*. The topmost menu, just above the table-of-contents region, contains a number of miscellaneous commands and status information. The commands, like all Laurel screen commands, appear in **bold-face** type; the status information is in normal type. We call this simply the *upper menu*. Below the table-of-contents region and above the message display region is the *middle menu*. The commands in this menu are used to manipulate selected entries in the table-of-contents. Below the message display region and above the composition region is the *lower menu*. This menu contains commands that facilitate the composition of new messages, responses to old ones, and manipulation of files. The composition region is separated from the feedback region by a single horizontal line.

2.2.1. Scrolling and thumbing

The table-of-contents region, the message display region, and the composition region each have a *scroll bar* in the margin to their extreme left. Within this scroll bar, the RED and BLUE mouse button clicks cause scrolling actions as in Bravo. Holding RED or BLUE down for some time will produce continuous scrolling, upwards or downwards respectively, by scrolling one line at a time until the mouse button is released. The cursor will appear as a double-headed arrow when positioned in the scroll bar and will change to point up or down when RED or BLUE is pressed.

Thumbing differs substantially from the Bravo style. At the top of each of the scrollable regions is a horizontal bar (called the *thumb bar*) that separates the region from the menu above. When the cursor is positioned just below the thumb bar and YELLOW is depressed and held, the cursor changes shape to a short vertical line segment. The position in the text to which you want to thumb is indicated by the position of this cursor on the thumb bar. The left edge of the bar corresponds to the beginning of the text, the middle of the bar to the middle of the text, the right edge of the bar to the end of the text, and all points in between to the corresponding relative positions in the text. When YELLOW is released while the cursor is still a short vertical line, the indicated position in the text is brought to the top of the region.

A portion of the thumb bar appears as a dashed line, whose position and length correspond to the position and length of the portion of text displayed in the window relative to the entire text. Thus, positioning the line segment at the left edge of the dashed line indicates the beginning of the currently displayed text. (Releasing YELLOW at this point naturally has no effect on the display.)

As you will see shortly, it is possible to *select* entities within the table-of-contents and composition regions. The position of the first selected entity in these regions is indicated on its associated thumbing bar by a short vertical line segment. This segment is always present, and should not be confused with the cursor shape that appears when you depress YELLOW. You can therefore obtain the effect of Bravo's Normalize command by pressing down on YELLOW and positioning the cursor so that it coincides with the permanent vertical segment on the thumbing bar. Releasing the mouse button then causes the selected item to be moved to the top of the region.

2.2.2. Adjusting region sizes

You can adjust the boundaries of the three major regions using the small squares at the upper right-hand corner of the middle and lower menus. Point the cursor at the desired box, then press down and hold YELLOW. By moving the mouse up or down, you drag the box with you to a new position on the screen. When you release the mouse button, the menu will move to the new position, and the contents of the adjacent regions will be adjusted accordingly. A fine point: you may find it more convenient to hold down YELLOW and move the cursor to the vicinity of the box. The cursor will capture the box when it comes sufficiently close.

2.3. The table-of-contents region

This region provides an index to the messages in the current mail file. Each entry in the index is numbered and contains the date sent, the sender, and the subject. It is also possible for each entry to have a *mark character* for classification purposes. Laurel does not permit you to modify the information in the table-of-contents window (except the mark character).

When Laurel begins execution, it normally gets the default mail file and displays its table-of-contents in the top region on the screen. It also places a *selection pointer* (which appears as a small black triangle pointing to the right) next to the last entry in the index (or first unexamined entry, if there is one). These pointers indicate messages to be manipulated.

2.3.1. Selecting messages

Entries in the table-of-contents region are selected in a manner similar to the selection of lines of text in Bravo. Position the cursor to the right of the scroll bar next to the entry you wish to select. The cursor will appear as a right-pointing arrow when it is properly positioned to change the selection pointer. Click RED. Any existing selection pointers will be removed, and a new one will be placed at the indicated entry.

If you click YELLOW instead of RED, then existing selection pointers will not be removed. YELLOW adds a message to an existing selection. If you click SHIFT-YELLOW (hold down either SHIFT key, click YELLOW, raise the SHIFT key), then the indicated selection pointer will be removed, without affecting any other selection pointers.

If you click BLUE, then selection pointers are *extended* from some existing selection pointer through the indicated entry. If you click BLUE at an entry whose number is less than the lowest numbered entry already selected, then selection pointers will be placed next to the indicated entry plus all entries between the indicated entry and the lowest numbered selected entry. If you click BLUE at an entry whose number is higher than the lowest numbered entry already selected, then selection pointers will be placed next to the indicated entry plus all entries between the indicated entry and the next lower numbered selected entry.

When making table-of-contents selections, think

RED:	NEW selection
YELLOW:	ADD to the selection
SHIFT-YELLOW:	REMOVE from the selection
BLUE:	EXTEND the selection.

2.3.2. Mark characters

You may wish to classify messages within a particular file. Laurel provides a primitive marking system by allowing you to supply a single *mark character* for any table-of-contents entry. To set a mark character, point the cursor at that mark character (to the left of the desired entry but to the right of the line bar and selection pointers) and click RED. A blinking caret will appear, inviting you to type a single character. This character will be retained in the table-of-contents entry and displayed whenever the entry is in the region. It will also be retained in the message if the message is moved to another file (using **Move to**, section 3.2.4). Some mark characters ("?" and "m") have additional semantics assigned to them by Laurel.

You may change a mark character by pointing the cursor at that mark and repeating the preceding steps. The proper way to remove a mark character is to replace it with a blank.

2.4. Status information in the upper menu

The upper menu contains some obvious status information: the version of Laurel you are running, the date and time, the status of your in-box(es), and the amount of free space remaining on your Alto disk. The remaining items are screen commands, which are discussed in section 3.

2.4.1. Laurel version

The upper left corner of the Laurel screen identifies the version of Laurel that you are running. Several versions of Laurel have been released in the past. Laurel 6 unifies and supersedes all previous versions and should be the version displayed in this area. Future maintenance releases of Laurel will add a fractional part to this version number, and significant new releases will change the integer part. If you send a message to LaurelSupport.PA, or otherwise refer to Laurel, please specify both the name and number that appear in the upper left corner of the screen.

2.4.2. Date and time

The date and time displayed in the upper right corner of the Laurel screen are set from the time maintained in your local network. If this time is wrong, please notify your local network services maintainer.

2.4.3. In-box status

The portion of the Laurel screen directly below the Laurel version is used for short messages regarding the status of your in-box(es). Message system users in Grapevine may own multiple in-boxes, allowing mail delivery even when their primary in-box server is down. Status messages for such users may vary slightly depending on the up-down status of individual in-box servers.

The meanings of status messages are as follows:

You have new mail. Laurel has determined through in-box polling that your in-box contains new mail for you (on at least one of your in-box servers). If you invoke the **New mail** command, this new mail will be retrieved to your current mail file. If your in-box server has become unavailable after the presence of new mail was detected, the *You have new mail* message will remain on the Laurel screen until you invoke **New mail**, at which time you will be given the bad news.

Mailbox empty at <time>. The most recent check of your in-box(es) has not determined the presence of new mail. Checking for new mail is done approximately every five minutes. The finest of points: a period terminates this message if every one of your in-box servers has asserted that no new mail exists. If some in-box cannot be contacted due to network failures or a down server, then the period is not used.

Login please. You have no password on your local disk. You must login (use the **User** command) before checking of your in-box(es) begins.

Bad user name or password. The name and password you have given do not correspond to those of any valid mail system user. Login with the **User** command. Remember that when you change your password on your disks and on file servers, you must change it in the message system also.

Can't contact any inbox servers. Due to network problems or down servers, no in-box servers can be contacted. Try again later.

Can't contact authentication server. Due to network problems or down servers, password checking cannot be performed. Try again later.

2.4.4. Free disk pages

The count of free pages left on your Alto disk is displayed below the date and time on the Laurel screen. You should check this number frequently to make sure that you have an adequate amount of space left for fetching new mail, etc. Laurel does not always recover gracefully from a full disk; it is a situation that should be avoided.

2.5. Message display region

This region is used for the display of messages from the current mail file. Messages may be scrolled and thumbed, but not edited in this region. Text displayed in this region is available as source text for shifted selections.

The middle menu, located above the message display region, contains commands that operate on the messages selected in the table-of-contents region. Descriptions of these commands are given in section 3.

2.6. Message composition region

The third region on the Laurel screen is used for composing messages to be delivered by the mail system. The message editor is quite general and it may be used for a variety of purposes in addition to its primary role as a message editor.

The editor contains many commands, some available on the screen and others available from the keyboard. The screen commands, used for initializing and filing messages, are discussed in section 3. The editor, including all keyboard commands, is discussed in detail in section 4.

2.7 The feedback region

Laurel uses the feedback region for three classes of information: status reports, exceptions, and confirmation requests. Status reports are displayed by various commands, e.g., **Deliver**, to report circumstances of interest to you but which require no direct action on your part. Exceptions are notifications of errors committed by you (or Laurel) and are flashed to alert you that some corrective action is probably required. Confirmation requests (see section 2.1) flash both the feedback region and a "?" in the cursor, alerting you to the need for immediate action before Laurel can continue.

3. Screen commands

3.1. Upper menu commands

3.1.1. User   Logging in

In order to perform **New mail** or **Deliver**, you must be an *authenticated user*. This means your name and password must be accepted by an authenticator for your registry. When you start up Laurel, it obtains your name and password from the Alto Operating System (and your registry from the Laurel profile, see section 5) and submits them to the authenticator for verification. If the authentication fails, a suitable message will appear in the in-box status area in the upper menu. Laurel will not allow you to receive or send messages until you have been authenticated.

The authenticator for your registry varies depending on the implementation of the Message Transport System in your registry. If there are currently any Grapevine (GV) servers in your registry, then Grapevine must know your correct password. This may be set or changed using the Maintain.laurel program (see section 3.6, **Run** command, and Appendix B, Maintain). If your in-box is on an IFS file server or on Maxc, then that machine must know your password also. You can set or change your password on an IFS or on Maxc by using the Chat.laurel program (see section 3.6, **Run** command, and Appendix B, Chat). There are situations in which your mailbox is on an IFS or on Maxc, yet there are also GV servers in your registry. If this is the case, then both GV and your in-box server machine must know your correct password, and it must be the same in both places.

To log in with a new name and password, point the cursor at **User** in the upper menu and click RED. Laurel will invite you to supply a user name in the brackets following **User**. Respond as you would for other **{**brackets**}** prompts (section 2.1). After you have entered your user name, Laurel will prompt you for a password in a similar way. You must supply the password *explicitly* ESC, YELLOW, etc. cause a null password to be entered. Laurel will not display your password as you type it.

If the **User** command is invoked with BLUE, then the current user name is accepted, and the **User** command will just prompt for your password.

Whenever you log in with a name that is the same as the name installed on your Alto disk, the password you supply is written back to the Alto Operating System. Thus, if you invoke **User** with name Guest and password Guest, the password is not passed back to the Alto Operating System (unless you happen to have installed the Alto Operating System with Guest as your name).

When using file commands, e.g. **Get**, **Put**, or **Copy** (section 3.5), if you try to operate on a remote file that requires connect credentials for access, you may have to log in with the **User** command with those connect credentials. Laurel will try several combinations of guest credentials and the current user credentials in attempting to access that file. The more subtle of these combinations require that the remote server accept Guest Guest as valid login credentials. Most IFS files servers work this way. Unfortunately, Maxc does not.

3.1.2. New mail Retrieving your new mail

New messages that are waiting for you reside in your *in-box*. The *in-box* is also called the *mailbox*. You instruct Laurel to move the contents of your in-box to your current mail file by pointing the cursor at the **New mail** command in the upper menu and clicking RED. The command will then appear on a gray background and the cursor will assume the shape of an hourglass, confirming that Laurel is busy shuffling messages. You may also observe the free page counter changing. When all messages have been transferred, the gray background and hourglass will disappear, and the table-of-contents window will be updated to reflect the new messages placed in your mail file. Transferring the contents of your in-box to your mail file renders the in-box empty.

During execution of the **New mail** command, terse progress indications are displayed in the feedback region at the bottom of the Laurel screen. These indications include the name of each in-box server that is contacted for new mail, the number of new messages retrieved from each, and possibly whether some of those messages have been archived. If the "archived messages" indication is shown, the mail retrieval may take slightly longer than usual; the in-box server has temporarily stored the oldest of your new mail on a secondary server to make room in its own storage.

The progress indication "failed" means that the in-box server broke the connection to your machine during message retrieval. This may be caused by a file server on which your archived messages reside being down.

When the **New mail** command has completed, the table-of-contents region is scrolled so that as much as possible of the new mail is displayed, and a selection pointer is set to point at the first new message. You will also observe that each new entry has a "?" mark character to its left this indicates that the contents of the associated message have not yet been examined. See section 2.3 for more information on mark characters.

3.1.3. Mail file Establishing your current mail file

Laurel maintains one or more *mail files* for you. You should think of mail files as folders in which you organize the messages you receive. When Laurel is first started, it normally checks to see if you have a mail file named Active.mail, and if not, it creates an empty mail file and names it Active.mail for you. This file is called your *default mail file* and is generally used to contain new messages that you have not yet processed. You will see how additional mail files are created in section 3.2.4.

To direct Laurel's attention to a mail file other than the current one, proceed as follows. Point the cursor at **Mail file** in the upper menu and click RED. A blinking caret will appear in the brackets following the command, inviting you to type the name of the file you wish to examine (see section 2.1.3.) Repeated clicks with RED on **Mail file** will fill in the brackets with the prefixes of all files on your disk that have a .mail suffix. After you have supplied the file name, Laurel will fill the table-of-contents region with the entries from the designated file. You can then use other Laurel commands to manipulate this file.

Mail file always reads from the file whose name is displayed in its brackets. However, it observes two conventions that simplify type-in. First, if the displayed name does not contain a period, Laurel implicitly appends .mail to the displayed name before accessing the file. Second, if **Mail file** is invoked with BLUE rather than RED, Laurel omits the prompt for a file name and uses the name currently displayed. Third, repeated clicks with RED on **Mail file** will successively fill in the brackets with the prefixes of files on your disk that have a .mail suffix.

A fine point: Laurel acts on deletions (see section 3.2.2) whenever either **Mail file** or **Quit** is selected. Thus, you cannot **Undelete** messages in a mail file once you have switched Laurel's attention to a different file.

If your mail file is malformed, then Laurel will inform you with a warning in the feedback region that says:

Format error in mail file. Get failed.

Laurel will not operate on that file any further. The most frequent causes of a malformed mail file are problems with your Alto disks or Alto hardware or booting your Alto during a **Mail file** or **Quit** command. In many cases, a malformed mail file can be restored to a relatively healthy state by running the MailFileScavenger.laurel program. See section 3.6, **Run** command, and Appendix B.6 for details.

Never turn off your Alto, turn off the disks, or interrupt Laurel's execution during **Mail file** *or* **Quit***, or your mail file may be permanently damaged!*

3.1.4. Quit Leaving Laurel

To exit from Laurel and return to the Alto Executive, point the cursor at **Quit** (in the upper menu) and click RED. Laurel will prompt you for confirmation. After you confirm, Laurel will act on the deletions indicated in the table-of-contents region, eliminating all messages from the mail file that have lines drawn through their table-of-contents entries (see section 3.2.2). When you re-enter Laurel at a later point, these messages will no longer appear in the table-of-contents. They are gone forever.

If you invoke **Quit** with BLUE, Laurel will omit the confimation prompt. It may still ask for confirmation if an undelivered and unfiled message exists in the message composition region.

Leaving Laurel by any means other than **Quit** is not recommended and will slow down subsequent re-entry to Laurel.

Never turn off your Alto, turn off the disks, or interrupt Laurel's execution during **Quit***, or your mail file may be permanently damaged!*

3.2. Received message manipulation

3.2.1. Display   Viewing a message

Normally, to examine the contents of a message, you must first select its entry in the table-of-contents. However, after obtaining new mail from your in-box, Laurel automatically selects the first new message for you. To display a selected message, simply point the cursor at the **Display** command in the selection commands menu and click RED. The selected message will then appear in the display region just below the menu. You may scroll the message or adjust the boundaries of the region for more convenient reading.

To examine the next message listed in the table-of-contents, click **Display** with RED again. The selection pointer will be moved to the next entry, and the text of the message will be displayed. Thus, although Laurel permits you to move the selection pointer explicitly, you need not do so. Simply click **Display** with RED repeatedly and Laurel will advance the selection pointer each time.

Fine points: Laurel will skip over deleted messages (see below) when advancing the selection pointer. To display a deleted message, you must select it explicitly, then click **Display**. If more than one message is selected, then **Display** will initially display the first selected entry, and subsequent clicks will display subsequent selected entries without disturbing the entire set of selected messages. After the last such selected entry is displayed, the next click of **Display** will display the first selected entry again. The only way to break this cycle is by explicitly selecting a single message.

3.2.2. Delete   Indicating messages to be destroyed

After examining some of the messages in your mail file, you may wish to delete them. The **Delete** command, when invoked by pointing the cursor at it and clicking RED, will cause a line to be drawn through all selected entries in the table-of-contents. You may find it convenient to use **Display** and **Delete** alternately when processing newly-arrived junk mail. (See also the DisplayAfterDelete option in section 5.)

Messages marked for deletion in this manner are not actually removed from your mail file at the time that the **Delete** command is given. Such messages are removed permanently from the current mail file whenever a **Quit** or **Mail file** command is executed.

3.2.3. Undelete   Indicating messages to be retained

If you discover that you have inadvertently marked some messages for deletion that you want to keep, simply reselect them if necessary, point the cursor at **Undelete**, and click RED. The lines drawn through the table-of-contents entries of the selected messages will be removed. Remember that you must give the **Undelete** command before you give a **Quit** or **Mail file** command; otherwise the deletions will take place.

3.2.4. Move to   Filing messages

Although Active.mail is your default mail file, and most of your message reading and storage may be done with this file, you may wish to classify messages of particular types in separate mail files. Laurel provides the **Move to {**file**}** command for this purpose. The **Move to** command will copy messages from your current mail file to the end of the file specified in the **Move to** brackets.

To invoke **Move to** proceed as follows. First, select the messages you wish to transfer. Next, point the cursor at **Move to** and proceed as you would for **Mail file** (section 3.1.3). After you have supplied the file name, Laurel will append the selected messages to the indicated file. The file name you supply is entirely of your own choosing; you may wish to use descriptive subject names, dates, persons' names, etc.

In addition to moving the selected messages, Laurel will draw a line through their table-of-contents entries as if you had invoked the **Delete** command. If you wish to retain any of those messages in the current mail file *as well*, you may use **Undelete** as described above. Laurel will also *mark* (in the mark character field) moved messages with an "m". This mark character may serve as a reminder that the marked message also exists in another file.

Move to does not move any messages that are indicated as deleted. If a set of messages containing both deleted and non-deleted messages is selected for **Move to**, only the non-deleted messages are moved.

Move to observes the same type-in conventions that **Mail file** does. It also implements one other convention that guards against misspelled file names: if the displayed file name does not correspond to any existing file, Laurel prompts you for additional confirmation before creating and writing a new file with that name.

3.3. Hardcopy   Printing messages

To print a copy of one or more messages in your mail file, select the message or messages in the table-of-contents, point at **Hardcopy**, and click RED. Messages that have been deleted (i.e., whose table-of-contents entries have lines through them) will not be printed, even if they are selected. Laurel will replace the middle menu with the *hardcopy submenu* that shows several properties to be used for the resulting hardcopies. The hardcopy submenu also contains two commands, **Hardcopy** and **Cancel**.

While the hardcopy submenu is on the Laurel screen, actual hardcopy processing is not being performed. In addition, no commands other than those in the Hardcopy submenu may be invoked while the hardcopy submenu is visible. Changing selections in the table-of-contents is permitted. After modifying the values of properties displayed in the hardcopy submenu, either point the cursor at **Hardcopy** and click RED, in which case hardcopy processing will begin, using the properties set this time, or point the cursor at **Cancel** and click RED, in which case the entire hardcopy command will be cancelled.

If you invoke **Hardcopy** initially with BLUE, then hardcopying will proceed immediately, using the current properties, without displaying the hardcopy submenu.

While hardcopy processing is proceeding, several status messages will appear in the feedback region of the Laurel screen to inform you of the progress being made. At any time before transmission to your printer has been completed you may cancel the hardcopy process by striking DEL or CANCEL.

When you invoke **Hardcopy** in the hardcopy submenu, any properties (except **Printer**) that you have set that are different from their default values will be used for the current hardcopy processing only. You will have to apply any non-default properties each time you invoke **Hardcopy**. Resetting these properties back to their default values also occurs when you invoke **Cancel** from the hardcopy submenu. You may change your default hardcopy property values via entries in the Laurel profile (section 5).

The standard values for the hardcopy properties will result in a single one-sided copy of each message printed on a standard blank form (no Xerox logo), with each message starting a new page. Detailed descriptions of each hardcopy property follow.

3.3.1. Printer

The name of the printer to which hardcopies will be transmitted appears in the **{**brackets**}** following the word **Printer**. The initial value for this property in each Laurel session is taken from your Laurel profile. Changes made to this property are retained for the rest of your Laurel session until you explicitly change this property. To change the **Printer** property, point the cursor at **Printer** in the Hardcopy submenu and click RED. A blinking caret will appear in the **{**brackets**}** and you may edit the value as with any other **{**brackets**}**.

If you set **Printer {**Local**}**, then printing will be done on a HyType printed connected to your Alto. If so, then be sure you are printing with a form that is appropriate for the HyType printer.

3.3.2. Form

Laurel 6 permits several different hardcopy layouts. These layouts are controlled by *hardcopy forms*, which are named and may be specified either in the messages being hardcopied or in the **Form {**brackets**}**. Laurel provides you with several built-in hardcopy forms: Blank, InternalMemo, Headers, and Archive. The purposes for which these built-in forms may be used are:

> Blank: Prints each message starting on a new page. Used for messages of a general nature. The message header is slightly formatted on output.
>
> InternalMemo: Prints each message starting on a new page. Used for messages relating to Xerox business only. The message header is substantially formatted on output. See section 6 for a fuller discussion regarding proper use of hardcopy forms.
>
> Archive: Prints messages continuously, with no page breaks in between. No formatting is performed except for a separator line between messages. Recommended over the previous two forms for archival purposes.

Headers: Prints an excerpt of the message header for each selected message without page breaks. Similar to the table-of-contents display.

In addition, you may specify your own custom hardcopy forms in the Laurel profile. Details of construction of custom hardcopy forms and details of the built-in hardcopy forms are given in Appendix C.

When the **Form {**brackets**}** are empty (this is the standard situation), the choice of hardcopy form used for each printed message is as follows: If a message contains a PrintForm: field in its header, then the form named in that field is used for printing that message. Otherwise, each message is printed on the default hardcopy form. By default, the name of the default hardcopy form is Blank. You may change the name of your default hardcopy form via an entry in the Laurel profile.

If the **Form {**brackets**}** contains the name of a hardcopy form, then this form will be used to print all selected messages during this invocation of the **Hardcopy** command. Any PrintForm: fields in the messages being hardcopied will be ignored.

After you invoke either **Hardcopy** or **Cancel** in the hardcopy submenu, the contents of the **Form {**brackets**}** will be reset to empty.

Although it is possible to change the default hardcopy form (used for those messages that do not contain a PrintForm: field), this change should not be made unless you have a real need to do so. The Blank form is adequate for most hardcopies, and special printing is possible using the **Form {**brackets**}** on a case-by-case basis. See also the discussion of hardcopy form usage in section 6.

3.3.3. Copies

To change the number of copies to be printed in a particular batch of hardcopy, point the cursor at **Copies** in the hardcopy submenu and click RED. A blinking caret will appear in the **{**brackets**}** following **Copies**, and you may then enter the number of copies in the normal manner. Only values from 1 through 99 are acceptable.

After you invoke either **Hardcopy** or **Cancel** in the hardcopy submenu, the contents of the **Copies {**brackets**}** will be reset to the default. The standard default value for **Copies** is 1; you may change this default via an entry in your Laurel profile.

3.3.4. Two sided

Some of the printers available in the Internet are capable of duplex printing, i.e., on both sides of the paper. If you send hardcopy to such a printer, you may wish to have your output printed two-sided. To change the value of the **Two sided** property, point the cursor at **Two sided** in the hardcopy submenu and click RED. The value in the brackets following **Two sided** will change from "no" to "yes" or from "yes" to "no".

When printing with **Two sided {**yes**}**, messages that normally start on on new page will start on a new sheet of paper.

After you invoke either **Hardcopy** or **Cancel** in the hardcopy submenu, the contents of the **Two sided {**brackets**}** will be reset to the default. The standard default value for **Two sided** is "no"; you may change this default via an entry in your Laurel profile.

3.3.5. Private

Spruce print servers (the standard printers available in the Internet) allow you to send hardcopy that will be printed only when you explicitly enter a password at the server. The **Private** property allows you to specify this for the current batch of hardcopy. To change the value of the **Private** property, point the cursor at **Private** in the hardcopy submenu and click RED. The value in the brackets following **Private** will change from "no" to "yes" or from "yes" to "no".

CAUTION: **Private {**yes**}** should not be used unless you have an important reason for doing so. The procedure for printing password protected files on Spruce is (currently) cumbersome and very error-prone. It also creates a substantial disruption in normal printer service. If you do use **Private {**yes**}**, then the password used for that batch of hardcopy is the password most recently given to the **User** command.

After you invoke either **Hardcopy** or **Cancel** in the hardcopy submenu, the contents of the **Private {**brackets**}** will be reset to the default. The standard default value for **Private** is "no"; you may change this default via an entry in your Laurel profile.

3.4. Message initialization commands

Composition is the process of building the header and body of a particular message. *Delivery* is the process of transmitting a message to its specified recipients. Laurel separates these two actions and supplies distinct commands appropriate to each in the lower menu.

Laurel provides four ways to initialize the content of a message. You may compose a *new* message, or *answer* one you have received, or *forward* existing message(s) to a new recipient, or *get* a previously composed form or message from a file. In the lower menu, there are four commands corresponding to these actions: **New form**, **Answer**, **Forward**, and **Get**. Simply select the one you wish by positioning the cursor appropriately and clicking RED. If the composition region contains an undelivered message, these commands request confirmation before constructing a new form. As in other such contexts, selecting these commands with BLUE automatically supplies confirmation. Even after confirmation, Undo or CANCEL (section 4.4.12) can still be used to recover the previous contents of the region.

It is important to understand that **New form**, **Answer**, and **Forward** only provide a message *form*; you must edit the form before requesting that it be delivered. The message you compose is simply an unformatted text string. However, Laurel does supply automatic line breaks as Bravo does, so you need not type CRs except to produce white space.

Laurel imposes format restrictions only on the message *header*. By definition, the header ends at the first blank line (i.e., two successive CRs). You should therefore be careful not to delete the blank line provided by Laurel in the initial forms. The header itself consists of a sequence of fields, some of which are required to be present. When composing a message you should always fill in the To: field and the Subject: field; you may delete the cc: field if it is not needed. When the message is ultimately delivered (section 3.7), Laurel will supply your name and the date, so you need not include them explicitly.

3.4.1. New form   Initializing a new message

The **New form** command gives you a new message form in the composition region. The form contains To:, Subject:, and cc: fields, each containing *placeholder* words, items that are bounded by matching triangles. These placeholders must be replaced by appropriate text or removed entirely from the form before delivery. In addition to a placeholder for extra recipients, the cc: field in the new form contains the current user's name.

3.4.2. Answer   Replying to a message

The **Answer** command initializes a message form so as to reply to the first message selected in the table-of-contents. The **Answer** command fills in the To: field with the sender of the selected message and sets the subject to be "Re: *subject of selected message*". The **Answer** command also sets the cc: field to include all of the recipients of the message being answered (as long as that message does not contain a Reply-To: field, see below). If you don't like these substitutions, you may, of course, change them using the editor.

If the message being answered contains a Reply-To: field in its header, then only those recipients listed in that field will be included in the To: field constructed by the **Answer** command. Also, your name (the current name in the **User** brackets) will be included in the cc: field of the message constructed by **Answer** in this case. The Reply-To: field in an outgoing message thus has the effect of restricting the recipients of answers to messages, when the recipients of such a message do not modify the recipient fields constructed by the **Answer** command. Including such a Reply-To: field is strongly suggested when sending a message to a large group of people. Good judgment and common sense exercised by message system users go a long way toward making our message system enjoyable for all. See also the discussion on Reply-To: in section 6.

3.4.3. Forward   Resending received messages

The **Forward** command initializes a message form by copying the complete text of all selected messages into the new form's message body, while providing a header similar to that provided by **New form**. Since Laurel enforces a limit of approximately 64,000 characters in the composed message, it is possible that not all of the selected messages will be copied.

3.5. File commands

Several commands in the lower menu retrieve files or public distribution lists, and store files. The files may be on your local Alto disk or on a remote server that accepts FTP connections.

The file manipulation commands all require arguments which you supply by typing into specific **{**brackets**}**. When one of these commands is invoked, the lower menu will expand so that its second line contains the appropriate **{**brackets**}** for that command. This second line will remain in the lower menu until such time as you invoke the command on the extreme right edge of this line represented as an upward pointing arrow under a short horizontal line. When invoked, by pointing the cursor at it and clicking RED, the lower menu will shrink back to one line.

To specify a local file (on your Alto disk) just use the normal file name. To specify a file stored on a remote machine, the most general form is [*server*]<*directory*>*filename*. If the remote machine is an Alto running FTP, then the <*directory*> must be omitted. If the remote machine is an IFS file server, then the *filename* may contain all the subdirectory text commonly used on such servers. Also, you may omit the <*directory*> if it is the same as your current **User** name. To specify a public distribution list, use the form *Name^.registry*. The *.registry* may be omitted if it is the same as your own registry.

The details of name interpretation are as follows:

0) A leading @ character is removed before applying the following rules.

1) If the first character in **{**brackets**}** is [, then the name is understood to be that of a remote file. The characters inside the square brackets are used as the name of the remote file server, and the remainder of the name is presented to the file server.

2) If not case 1, then if the **{**brackets**}** contain an ^ character, then the name is interpreted as a public distribution list name. If no *.registry* suffix appears in the name, then Laurel will implicitly append the *.registry* from the **User** brackets. The complete name is presented to the message transport system, which will return the contents of that list as would be used for delivery.

3) If neither case 1 nor 2 applies, then the name is interpreted as a local file name. If this name does not contain a period (.) and no local file of that name exists, then Laurel will use a file with that name with .form added as a suffix.

3.5.1. Get Reading a file or distribution list

The **Get** command replaces the entire contents of the composition region with the text contained in the file or distribution list specified. To invoke **Get**, point the cursor at **Get** and click RED. A second menu line containing

File name **{**brackets**}**

will appear, with a blinking caret within the **{**brackets**}**. Enter the filename (or distribution list name) using the brackets type-in facilities described in section 2.1.3.

Repeated clicks with RED on the **Get** command will cycle the contents of the *File name* **{**brackets**}** through all of the filenames on your local disk whose suffixes are .form.

If you invoke **Get** with BLUE, then Laurel will execute the command immediately using whatever name is already inside the *File name* **{**brackets**}**. This may be a bit chancy if the *File name* **{**brackets**}** menu line is not currently visible.

Get and **Put** are useful when you want to save a partially composed message to be completed and delivered later. You may also want to do this if you encounter persistent network or mail server problems upon attempting to **Deliver**. **Get** is also handy for obtaining custom forms (or form letters) that you have prepared previously. If the file read with **Get** has the suffix .form, then the Laurel editor will automatically select the first placeholder in that file.

Get may not be used for files containing over 64,000 characters. **Get** may be used safely only for text files. Files that contain binary information will be modified by the **Get** command. To copy files containing binary information to your disk safely, see the description of the **Copy** command, section 3.5.3.

See also the description of the G (COM-G) editor command (section 4.4.6), which replaces only the selected text in the composition region, rather than the entire text in the composition region.

3.5.2. Put Writing a file

The **Put** command writes out the entire contents of the composition region onto the local or remote file whose name is inside the brackets. **Put** will not write onto a public distribution list. Use Maintain (Appendix B). (If the file is local and a file with that name already exists, Laurel requires an extra confirmation before overwriting it. **Put** onto a remote file follows the file server's conventions, which usually means that a new version will be created.) To invoke **Put**, point the cursor at **Put** and click RED. A second menu line containing *File name* **{**brackets**}** will appear, with a blinking caret within the **{**brackets**}**. You may enter the filename in the normal way.

The **Put** command shares the same *File name* **{**brackets**}** with the **Get** command. The contents of the *File name* **{**brackets**}** are remembered across commands, and even between separate sessions of Laurel. They are initialized to empty when Laurel installs itself. This allows one to **Put** a message and on a subsequent **Get**, the filename used during the **Put** is still in the *File name* **{**brackets**}**, ready to be confirmed.

The .form automatic extension allowed by **Get** are not provided with **Put**. The only way to write on a file with a .form extension is by explicitly typing the .form extension with the filename.

See also the description of the P (COM-P) editor command (section 4.4.8), which writes out only the selected text in the composition region, rather than the entire text in the composition region.

3.5.3. Copy   Copying or transferring a file

The **Copy** command provides a simple means of copying files within a file system or of transferring files from one file system to another. To invoke the **Copy** command, point the cursor at **Copy** and click RED. A second line will appear in the lower menu containing

from **{**brackets**}**   *to* **{**brackets**}**

with a caret blinking in the *from* **{**brackets**}**. At this point, fill in or confirm the file or distribution list name in these brackets; this will be the the source from which a copy will be made. When this name has been confirmed in the usual manner, the blinking caret will move to the *to* **{**brackets**}**. Fill in or confirm these brackets in the same manner; this will be the name of the file (distribution list are not allowed here) that will be the new copy. Striking DEL or CANCEL at any time before confirming the *to* **{**brackets**}** will cancel the entire **Copy** command.

The names specified for either of the *from* or *to* **{**brackets**}** may be local or remote in any combination. If both names specified are local, then a local copy is made of a local file (similar to the Alto executive's Copy command). If the *from* name is local and the *to* name is remote, then the **Copy** command behaves as an FTP Store command. If the *from* name is remote and the *to* name is local, then the **Copy** command behaves as an FTP Retrieve command. Finally, if both names specified are remote, then a remote copy of a remote source is made. There is no analogous FTP command for this last case. Also, no buffering on the local disk is necessary for a remote to remote **Copy**.

If the **Copy** command is invoked with BLUE, then the *from* name is automatically confirmed, and the blinking caret starts in the *to* **{**brackets**}**.

3.6. Run   Running a program; Laurel extensions

The **Run** command allows you to run certain Mesa programs without leaving Laurel to do so. The **Run** facility is a general-purpose mechanism that allows a wide range of features to be added to your Laurel without increasing the size of the standard Laurel program. The basic idea is that Laurel provides general message system functions that you may extend simply by adding optional runnable programs to your local disk. The Laurel group has provided several "standard" runnable programs; these have and will be augmented by numerous special purpose programs built by a wide variety of message system users.

Some runnable programs (such as Chat.laurel) extend Laurel by providing useful features that are only marginally connected with Laurel, but provide a powerful increase in the utility of Laurel by integrating new functions. Other programs (e.g., InsertMail) are deeply integrated into Laurel's organization, providing functions on Laurel's own objects. Still other programs (say a desk calculator program) may provide useful (or frivolous) functionality that is currently lacking in Laurel. The possibilities are legion.

Many of the programs available use the composition region as a teletype style typescript (e.g., Chat.laurel, Maintain.laurel). When such a program has finished its execution, its typescript remains in the composition region, which may be scrolled, edited, filed, etc., using the standard Laurel facilities. Other programs (e.g., Fog.laurel, InsertMail.laurel) may merely affect certain objects that Laurel manipulates (e.g., mail files, the displayed message region) without affecting the composition region at all. Appendix B gives descriptions of the "standard" Laurel runnable programs made available by the Laurel group.

To invoke the **Run** command, point the cursor at **Run** and click RED. The second line of the lower menu will change to one that includes

Program **{**brackets**}**

with a blinking caret inside the **{**brackets**}**. Fill in or confirm the name of a program in the **{**brackets**}** in the usual manner. The program will then be loaded into Laurel and its execution will begin. No confirmation is necessary even if the text in the composition region has not been filed or delivered. This text is restorable (with Undo or CANCEL, section 4.4.12) after the **Run** command has finished. The **Run** command will remain gray during the entire execution of the program. Each program has its own rules about user interactions and termination. The "standard" runnable programs provided by the Laurel group have somewhat similar interfaces and tend to be self documenting.

The default extension for programs to be run in Laurel via the **Run** command is .laurel. You need not type the .laurel extension into the *Program* **{**brackets**}** if the program you wish to run has a .laurel extension. Also, after invoking **Run** with RED, subsequent RED clicks on the **Run** command will cycle the contents of the *Program* **{**brackets**}** through the prefixes of all files on your local disk that end in .laurel. For Mesa programmers: the programs that run in Laurel are standard Alto Mesa 6 .bcd files; a .laurel file is merely a .bcd file renamed. The .laurel extension was chosen so as to distinguish programs intended to run inside Laurel from the large population of random .bcd files that most Mesa programmers have on their local disks. To run a suitable program whose extension is not .laurel, just type the entire file name into the *Program* **{**brackets**}**.

The **Run** command also allows the running of a program that is not on the local disk at the time the command is invoked. If a remote filename is entered into the **{**brackets**}**, then that program will be retrieved onto the local disk first, the **{**brackets**}** will be changed to reflect the local filename, and the program will then be loaded and run in the normal manner. Such programs retrieved "on the fly" will remain on the local disk until you explictly delete them from the Alto file system. See also the description of the RunPath Laurel profile option (section 5).

Before a program is loaded with the **Run** command, Laurel makes several consistency checks to make sure it may properly be run within Laurel. After the program has completed, more checks are made to insure that the program just run has not damaged Laurel’s environment. Violations of these checks may produce amusing results.

3.7. Deliver   Sending a message

Before you attempt to send a message to its recipients, you must compose the outgoing message properly. This section gives the rules for composing messages and naming recipients first, then gives a detailed description of the delivery process.

In order to make inadvertent message delivery difficult, the **Deliver** command may not be present on the screen at some times. These times are at Laurel startup and immediately after the **New form**, **Answer**, **Forward**, and **Deliver** commands. The **Deliver** command will appear after any edit is performed on the composed message.

3.7.1. Editing the message header

Recall that a message contains two major parts: the header and the body. Since Laurel allows free editing of the outgoing message, there are certain rules that you must observe in order to keep your message intelligible to the Deliver parser. The header and body are separated by a double CR; this causes the appearance of a blank line between the header and body when displayed on the Laurel screen. All of Laurel's initial message forms have this blank line following the skeleton header. Do not remove that blank line or insert a blank line inside the header; otherwise information contained in the header will be misunderstood.

Each separate field of the header begins on a new line with a word immediately followed by a colon. The word preceding the colon specifies the meaning of the word or words following the colon, e.g., To: specifies that the words following the colon are to be interpreted as recipient names; Subject: specifies that the words following the colon are merely text and not recipient names. There should be no reason to type a CR in the text following the colon except for the CR that terminates the entire field. If it is necessary to insert a CR for line folding purposes, the combination CR BLANK is interpreted by all message parsers as equivalent to BLANK and will not terminate the field.

Laurel identifies items that it expects you to replace by supplying a keyword bracketed by matching black triangles. Laurel will refuse to deliver any message whose header contains one (the left one) of these black triangles. This protects you from simple oversights, such as forgetting to supply a Subject: field.

3.7.2. Naming recipients

The recipients of a message are specified in the To: and cc: fields of the header. You need to know the registered name of each individual or group to whom you want to send a message. Many organizations publish local telephone lists that include mail system recipient names. A Xerox-wide, message system "white pages" directory is planned. Of course, you can often discover these names by looking in the headers of incoming messages.

A recipient name has two parts separated by a period. The second part is a *registry* name, and the first part is the identifier for someone in that registry. Most registry names currently correspond roughly to "campuses" of activity within Xerox, which should make them easier to remember. At present, the following registries exist:

DLOS -	Dallas, TX
EOS -	Electro-Optical Systems, Pasadena, CA
ES -	El Segundo, CA (now served by Grapevine)
HENR -	Henrietta, NY
LB -	Leesburg, VA
PA -	Palo Alto, CA (now served by Grapevine)
RX -	Rank Xerox, London, England
STHQ -	Xerox Corporate Headquarters, Stamford, CT
WBST -	Webster, NY
XRCC -	Xerox Research Center, Toronto, Canada

Note that the registry names tend to be geographic rather than organizational it's easier to remember approximately where individuals work than what organizations they are a part of today. Also, at the time of this writing, only the ES and PA registries are served by Grapevine, thus enabling many advanced message transport features. In time, most of the other registries will be converted to Grapevine.

Most messages involve activities within a single campus, and since registries correspond to campuses, most of the recipients of a message tend to be in the same registry. It would be cumbersome if Laurel always required that you use the complete name for all recipients, just as it would be cumbersome to have to dial an area code to telephone your neighbor across the street. Laurel allows you to omit the registry name for recipients who are in your registry. For example, someone in the registry for the Palo Alto area, say "Someone.PA", could send a message with the following acceptable message header:

```
Subject: demonstration of recipient naming
To: Person1, Person2
cc: Person3, FarAwayPerson1.ES
```

Laurel presumes that the names that lack registries are in the sender's registry, which, in this case, is "PA". Since "FarAwayPerson1.ES" explicitly includes a registry, no ".PA" suffix is presumed by Laurel. Thus, in this case then message will go to "Person1.PA", "Person2.PA", "Person3.PA", and "FarAwayPerson1.ES".

3.7.3. Public distribution lists

The mail system provides a way to address messages to groups of recipients. Each registry includes some recipient names whose first part ends with the character "^". Such names identify groups rather than individuals. Using such a name as the recipient of a message causes the message to go to all the individuals included in the group. For example, the To: line

```
To: CSL^.PA
```

will cause the message to be delivered to all of the approximately 75 recipients in the Computer Science Laboratory of the Palo Alto Research Center. If the sender is in the PA registry, then the .PA may be omitted.

The public distribution lists for each registry are stored on mail server machines, and are maintained by the administrators of that registry, by the owners of the lists, and in some cases by the members of the lists. To create a public distribution list you must contact an administrator for your registry, who will make sure that your proposed name does not conflict with any others, and who will create the list for you. You can get your name added to appropriate lists by contacting an owner of that list, or, if you are in a Grapevine registry, by using the program Maintain.laurel (see section 3.6, **Run** command and Appendix B). In Grapevine registries, by convention, the owners of a list named *list^.registry* are listed in the companion list Owners-*list^.registry*. If you cannot make a desired change to a public list yourself, then sending a message to Owners-*list^.registry* is a good way to request that the changes be made.

While you are permited to address messages to any public distribution list (in any registry), you should think very carefully about your choice of message and list so as not to bother recipients with messages they don't care to read. Check with experienced users to find out which lists should be used for which kinds of messages (see also section 6).

You need not worry about a recipient appearing in more than one distribution list. The message transport system will detect duplicate names among the recipients and ensure that each receives the message only once. Thus, you may use multiple distribution lists freely within the To: and cc: fields of the message header.

3.7.4. Private distribution lists

Suppose you have a collection of people to whom you frequently send messages. To avoid having to type the entire list of names every time you send a message to the group, you may proceed as follows. Create an Alto file containing the list of names (separated by commas). You may find it convenient to "label" such distribution lists; you may do so by prefixing the list with identifying text followed by a colon, and then terminating the entire list with a "matching" semicolon. For example, such a file might contain

Laurel Alpha users: Brown, Doe, Jones, Smith;

Give this file a name, say "AlphaUsers.dl".

Now, when composing a message in Laurel, include the file name in the appropriate field of the message header (e.g., following To: or cc:), preceded by an "@" character:

To: @AlphaUsers.dl

While delivering the message, Laurel will read the file to determine the recipients. Multiple private distribution list specifiers may be used within a single field and may be intermixed with names of individual recipients and public distribution lists separate them all with commas. By convention, private distribution list file names end with .dl (for distribution list). @ expansion requires that the complete filename be given; no abbreviation mechanism exists.

Private distribution lists need not be stored on your local disk. Laurel will accept a complete, remote file name following an "@", for example:

@[Ivy]<LaurelSupport>LaurelUsers.dl

As with public distribution lists, you need not worry about a recipient appearing more than once. Laurel will detect duplicate names among the recipients and ensure that each receives the message only once.

3.7.5. Delivery

Once you have composed the message you wish to transmit, you may initiate its delivery to the recipients by pointing the cursor at **Deliver** and clicking RED. Laurel will fill in your name and the date (though they won't appear in the composition region) and proceed to send the message. A gray background will appear behind the **Deliver** command and the cursor will change to an hourglass. If Laurel discovers an error in the list of recipients, it will give you an opportunity to cancel the delivery and correct the mistake. When the list is acceptable (i.e., all specified recipients are known to have in-boxes), Laurel will deliver the message and remove the hourglass and gray background. After successful delivery, the word **Deliver** will disappear and will be replaced by "*delivered*". If an error occurs during delivery, an explanation of the error condition will be displayed in the feedback region. You may cancel delivery while the message "*Type DEL if you wish to cancel delivery*" appears in the feedback region.

Deliver tells you the number of recipients to whom it will deliver the message. If this number exceeds 30 or if the recipient list includes public distribution lists, and your message does not include a Reply-To: field, you must confirm the delivery. This is intended to minimize unintentional deliveries to large distribution lists.

When you are asked to confirm, you have three choices: to deliver the message including a standard Reply-To: field so that answers will go to yourself only, to deliver the message without a Reply-To: field so that answers will normally go to all recipients specified in your message, or to cancel the delivery at this point. A message will appear in the feedback region indicating the choices available. Striking ESC will deliver with the Reply-To: <self> included (this is recommended in most cases), striking "A" (for "All") will deliver without a Reply-To: field, striking DEL (or anything else) will cancel the delivery at this point. If you cancel the delivery at this point, a Reply-To: <self> field will be added to the message anyway; you may edit this field to include other names if you wish and re-invoke **Deliver** when ready.

Extra confirmations may be necessary if invalid recipient names are detected. One confirmation per invalid name will be asked for. If **Deliver** is invoked with BLUE then Laurel will not require these invalid recipient confirmations. Large list Reply-To: confirmation is still required with BLUE. This confirmed delivery mode is useful when sending a message to a large private distribution list that may contain a few invalid recipient names. Laurel will deliver to all the valid recipients and report the number of invalid recipients when delivery is completed. When finished, **Deliver** also tells you the size of the message (in characters).

Before filling in your name in a composed message, Laurel will check to see if the message header already contains a "From" field. If so, Laurel inserts a "Sender" field with your name and leaves the "From" field untouched. It is the "From" field that is normally displayed in the table-of-contents. The name Laurel furnishes is your logged-in user name see section 3.1.1, **User** command.

3.7.6. Corresponding with Arpanet recipients

The Xerox mail system is connected to the ARPA network, and Laurel users can correspond with individuals at other ARPA sites. Arpanet recipients at sites other than Parc-Maxc may be specified in the message header by either "*name@ArpaHost*" or "*name* at *ArpaHost*".

Laurel recognizes messages that include Arpanet recipients except when the only such recipients are contained in a public distribution list and performs a number of additional actions. When constructing a reply with **Answer**, Laurel will retain *ArpaHost* names for foreign recipients. During delivery, Laurel will append appropriate qualification to Xerox recipient names, ensuring that responses from the remote ARPA site will reach them. Thus, you never need to append "@Parc-Maxc" explicitly.

Laurel requires you to supply the *.registry* extension on all Xerox recipient names if you are sending the message to at least one Arpanet recipient and your registry is not .PA. Otherwise, Laurel will complain "local names need registry appended".

Laurel supports a subset of the ARPA standard for message headers (RFC 733/NIC 41952). Happily, users need not be aware of most of the requirements of this extensive standard except for the warnings about CR's in message headers stated at the beginning of this section.

If you include private distribution list names in the header, the ARPA standard requires that they be enclosed in quotes; for example:

"@[Ivy]<LaurelSupport>LaurelUsers.dl"

Laurel will still recognize and expand your private distribution list, and will remind you when quotes are necessary.

4. The Laurel Editor

The Laurel editor provides a convenient point-and-type style screen editor for editing messages in the composition region of the Laurel screen. Users of Bravo should be able to use the Laurel editor with almost no training, since the Laurel editor mimics most of Bravo's interaction style. Formatting facilities are restricted to line-turning only; there are no "looks" in the Bravo sense. Only one font (a variation of TimesRoman 10) is supported, and only one face (plain) is available for text. Given these restrictions, however, the Laurel editor does provide many facilities that in some areas go well beyond Bravo.

The single Laurel program allows a user to select between two distinct editing styles, called *modal* and *modeless*. A modal editor style means that depending on the state (or *mode*) of the editor only certain actions are allowable. The meaning of a particular keystroke may be different at different times, depending on the current mode of the system. A modeless editor style means any command may be given at any time, regardless of the actions performed up to that time. According to this definition, the Laurel modeless editor is not, strictly speaking, modeless. It does come much closer to this ideal, however than does the modal editor.

The Laurel modal editor is nearly identical in command structure to the familiar Bravo 7 and 8 editors and to the previous Laurel 5.1 editor. The actions necessary to insert text in this editor are roughly: Select an item; type a text insertion command letter (A, I, or R); type or select text for insertion; terminate insertion (type ESC).

The Laurel modeless editor includes all of the commands of the modal editor, but the actions necessary for common operations are more streamlined. The modeless editor is similar in command structure to the BravoX and LaurelX editors. In the modeless editor, the actions required to insert text are: Select insertion point; type or insert text. No explicit start or stop of insertion is required.

Since the commands available in either editor are similar, this section describes each command in only one place, giving differences between modal and modeless in fine points. When a keyboard command is discussed, it will be identified by its command letter in two ways, for modal and for modeless editor invocation. In the modal editor, a command is distinguished by its being given at the appropriate time, i.e., when a caret is not blinking. Such commands are given merely by typing the command letter. In the modeless editor, where the caret is always blinking (you are essentially in type-in mode all the time), some other way must be used to distinguish a command letter from a typed-in letter. The Laurel modeless editor does this by requiring that an extra key be held down while you type the command letter (similar to holding down SHIFT for capital letters). The key used for this purpose is called COM (for COMMAND; see section 4.4 for its location on various keyboards.)

An editor keyboard command, for example, the Time insertion command is described as T (COM-T), indicating that in the modal editor the command is given by typing the letter T when in command mode (no blinking caret), or by COM-T in the modeless editor (at any time). Some commands allow synonyms, as with the text deletion command, D (DEL, COM-D). Here, in the modal editor, the command is given by typing the letter D when in command mode, while in the modeless editor the command is given either by typing the DEL key or by typing COM-D.

4.1. Getting started

There are numerous parameters that modify Laurel editor operations that you may set in your Laurel profile if you choose (see section 5). Fortunately, all of these parameters have default settings, so that you may use the Laurel editor without setting any of these parameters yourself. Laurel chooses the modal editor by default for compatibility with previous versions.

Should you wish to use the modeless editor, you must place the entry

```
Editor: Modeless
```

in your Laurel profile. A strongly recommended entry for the Laurel profile should you choose the modeless editor is

```
BluePendingDelete: TRUE
```

which will enable a convenient feature in that editor.

After having used either of the Laurel editors for some time, you may wish to set some of the other parameters in the Laurel profile to suit your own taste.

4.2. Selecting Text

The mechanics of selecting text in the Laurel editor are quite similar to those of Bravo, although Laurel extends the available functionality considerably. With one mouse button click you may select a character, word, line or paragraph. With an extra click you can extend the current selection. By clicking multiple times it is possible to expand a selection from character to word to line to paragraph to the complete document, or you may shrink one edge of the selection back down through the selection hierarchy.

4.2.1. Target and source selections

Two general types of selection will be referred to in this manual. The first is *target* selection, known in Bravo terms as *primary* selection. This selection is called target because it indicates where the editing will occur. A target selection is indicated on the screen with a solid underline or with a solid bottomed video reversal in the modeless editor when the selection is a replace selection. The other type of selection is *source* selection, known in Bravo terms as *secondary* selection. This selection is called source because it indicates the source of text to be copied. Source selections, both secondary and shifted, are indicated on the screen by a dotted underline or by a "serated bottom" video reversal in the modeless editor for move selection.

The mechanics of selection described below apply to both target and source selections except when otherwise indicated.

4.2.2. The selection hierarchy

The *selection hierarchy* refers to the character, word, line, paragraph, and document levels at which a text selection may be made. Going up the hierarchy means changing levels to the next higher one, e.g., from character to word. Note that line is considered higher than word, although a very long word that occupies multiple lines may actually be longer than a line. Such a case is rare. Going down the hierarchy means the opposite, i.e., changing levels from a higher level down to the next lower one.

4.2.3. Character selection

A character is any single character. To select a character, point the cursor at that character and click RED. A solid underline will appear under the character selected. If you hold RED down while you move the cursor, you will see the solid underline move from character to character as the cursor moves. When you lift up on RED, the solid underline will remain under the last character thus underlined.

In character selection, as with all text selection actions in Laurel, the previous selection is remembered while the new selection is being made. If the cursor is moved out of the text region in which the selection is being made while the mouse button used for selection is still down, the previous text selection is restored. To make a selection "stick", the mouse button (in this case RED) must be lifted up while the cursor is still pointing at the selected object. This last feature allows you to push down a mouse button at any place on the Laurel screen and hold it down while moving the mouse to any desired location (e.g., a screen command) without danger of destroying any selections along the cursor's trajectory.

4.2.4. Word selection

A word is any of the following: a contiguous sequence of letters and digits, a contiguous run of blanks and tabs including at most one CR at the right edge, or a single punctuation character. To select a word, point the cursor at any place within that word and click YELLOW. A solid underline will appear under the word selected. If you hold YELLOW down while you move the cursor, you will see the solid underline move from word to word as the cursor moves. When you lift up on YELLOW, the solid underline will remain under the last word thus underlined.

A word may also be selected with two rapid clicks of RED, as described in *Selection expansion* below.

4.2.5. Line selection

A line is one line of text as displayed on the Laurel screen. Lines are somewhat arbitrary units of selection when several contiguous lines contain no CR, since the exact break point between lines depends on the widths of characters and on the width of the Laurel screen. To select a line, move the cursor into the *line bar*, the area of the Laurel screen to the left of the text and to the right of the scroll bar. When the cursor is in the line bar it appears as a right-pointing arrow. Point the cursor at the desired line and click RED. A solid underline will appear under the line selected. If you hold RED down while you move the cursor in the line bar, you will see the solid underline move from line to line as the cursor moves. When you lift up on RED, the solid underline will remain under the last line thus underlined.

A word may also be selected with three rapid clicks of RED in the text area or two rapid clicks of YELLOW in the text area, as described in *Selection expansion* below.

4.2.6. Paragraph selection

Although Laurel does not implement Bravo style paragraphs, it does provide a selection mode that gives similar results in simple cases. In Laurel, a paragraph is a unit of text bounded by double CR's or by the ends of the document. A double CR appears on the screen as a single blank line--the first CR terminates the previous line of text while the second CR is the blank line. A selected paragraph contains the terminating double CR (if any) but not the initial CR. In a run of several CR's, a selected paragraph will contain only a single CR, thus avoiding overlapping paragraphs.

To select a paragraph, move the cursor into the line bar (see line selection above), point the cursor at any line in the desired paragraph and click YELLOW. A solid underline will appear under the paragraph selected. If you hold YELLOW down while you move the cursor in the line bar, you will see the solid underline move from paragraph to paragraph as the cursor moves. When you lift up on YELLOW, the solid underline will remain under the last paragraph thus underlined. Paragraph selection in a large document that contains few double CR's may take several seconds on an Alto.

A paragraph may also be selected with four rapid clicks of RED in the text area, three rapid clicks of YELLOW in the text area, or two rapid clicks of RED in the line bar, as described in *Selection expansion* below.

4.2.7. Document selection

In Laurel, there is no single button click to select an entire document, although the E (COM-E) command (section 4.4.4) may be used for this purpose in the message composition region only.

A complete document may be selected with five rapid clicks of RED in the text area, four rapid clicks of YELLOW in the text area, three rapid clicks of RED in the line bar, or two rapid clicks of YELLOW in the line bar, as described in *Selection expansion* below.

4.2.8. Selection extension

After having made an initial selection (with RED or YELLOW), you may wish to extend the selection to include more text. This is done by pointing the cursor at the new endpoint for your selection and clicking BLUE. At any time, a selection is said to be at a particular level in the selection hierarchy. In general, the extension to the selection will be made at the same level as the selection. For example, if your existing selection is at the character or word level, and the extension is made by pointing within the text, then the extension will be made at the character or word level respectively.

If you extend a character, word or line selection by pointing in the line bar, then the extension is made at the line level and the selection level is changed to the line level. When a selection is already at the line level, then extensions made by pointing within the text will also be made at the line level.

If a selection is already at the paragraph level, then extension by pointing within the text or within the line bar is made at the paragraph level. If a selection is already at the full document level, then extension merely reselects the full document.

Extending a selection to some point, when that point is not inside the existing selection, will extend the selection so that all of the existing selection plus the extra text up to (or down to) that point is included in the resulting selection. Extending a selection to some point that is within the existing selection will actually cause the selection to shrink. In this case, the endpoint of the existing selection that is closer to the beginning of the message will be retained in the resulting selection.

4.2.9. Selection expansion

Selection expansion differs from selection extension in that it expands the current selection to the next higher level in the selection hierarchy. Selection expansion is accomplished by rapid multiple RED or YELLOW mouse clicks, where each click after the first is given within the selection existing at that time. For example, pointing at a character and clicking RED once will select that character as usual. Pointing at a character and clicking RED twice will select the word containing that character; clicking three times will select the line containing that character; four times will select the paragraph containing that character; five times will select the entire message. Multiple clicks with YELLOW are similar, except that the first click selects a word and subsequent clicks go up the hierarchy from there.

Multiple clicks with RED in the line bar start at the line level, then up to paragraph level and then to document level. Multiple clicks with YELLOW in the line bar start at the paragraph level, then go to document level.

Clicking rapidly is important. There is a timeout associated with multiple clicking for selection expansion. If the next click is given within the timeout after the last click is completed, the selection is expanded. If the next click is given after the timeout has elapsed, then the selection starts at the original level in the selection hierarchy (for that button and position) again. The timeout should be selected so that stationary multiple clicks are naturally produced within the timeout, while mouse motion between clicks uses enough time so that the multi-click hieraracy starts over.

The default timeout for selection expansion via multiple clicks is 0.39 seconds. It is possible that your own mouse habits may not be in agreement with this setting, causing you to get selection expansion when you don’t want it or vice versa. If this is the case, or if you wish to disable the multiple click--selection expansion feature altogether, then you may want to use the Click: entry in the Laurel profile (see section 5).

4.2.10. Selection reduction

The level of a selection may be reduced by multi-clicking BLUE. The first BLUE click causes extension as usual. If another BLUE click is given within the timeout, then the level of the selection is reduced by one (but never below character level). This feature is useful if you start a selection at the word (or higher) level, but after extending the selection you realize that the new endpoint should be at a character boundary. Just re-click BLUE several times to knock the selection level down, and select the endpoint as you wish. Selection reduction uses the same timeout as selection expansion (see above).

4.2.11. Type-in point selection (modeless editor only)

To set the type-in point in the modal editor, a keyboard command such as A, I, or R must be given (see section 4.4). In the modeless editor, the type-in point is selected along with the text whenever you make a target (primary) selection. The type-in point is shown by a blinking caret (also called the insertion caret); the tip of the caret shows where typed-in characters will be inserted. Thus in the modeless editor, you get both an underlined piece of text and a blinking caret together with each selection action.

The type-in point is placed between characters at one end of the selected text. This end is the one closest to the tip of the cursor when the selection is made.

Text geography lesson: The point after the last character on a line is logically the same as the point before the first character on the next line. Laurel will place the type-in point *before* the last character on a line when the above rule states that it should be placed after that character. This is done because 1) a caret meant to blink at the point after the last character on a line is always placed before the first character of the next line, 2) such a placement would be spatially distant from the text selection and thus confusing, and 3) it is the "right" thing. Thus you may sometimes see the display of the blinking caret slightly "inside" of the solid underline. If you wish to place the caret after the last character of a line, then select the beginning of the next line. The only way in the modeless editor to position the insertion point after the last CR of a message is with the COM-A command (section 4.4).

In general, one is interested in either selecting a piece of text or in selecting the type-in point, but not both. If you want to select a piece of text in the modeless editor, just select that text ignoring the placement of the insertion caret. If you want to place the type-in point, then point the tip of the cursor at the point between characters where you want the type-in point to be. The selection underline will be drawn under either the previous or subsequent item. The underline placement in this case is irrelevant, the important fact to remember is that the insertion caret will be placed properly if you point at the spot where you want it to be.

4.2.12. Replace selection (modeless editor only)

When you make a replace selection, subsequent type-in will delete the selected text automatically in addition to inserting the typed-in characters. Although in the modeless editor it is possible to replace text by selecting that text, deleting it, and then typing in the new text, it is easier to select the original text with replace selection and then just type the new text.

Whenever a selection is made a replace selection, it is indicated on the screen by video reversal rather than by the usual solid underline. For example, the initial text selection in the skeleton form provided by the **New form** command is made this way.

There are several ways in which you can select text with replace selection. One is that several commands (**New form**, **Answer**, **Forward**, **Find**, and NEXT--section 4.4) select text this way. A second way is enabled if you have set the BluePendingDelete feature in your Laurel profile (highly recommended if you use the modeless editor). If this feature is enabled, then all selections made with BLUE automatically become replace selections. The reasoning behind this is that if you use BLUE for selection, you are more interested in the selected text than in the insertion point, and your interest in this text is for purposes of deleting or replacing it. On the other hand, if you only want to insert text, then there is no need to use BLUE in making the selection. The parameter name "BluePendingDelete" is retained for historical reasons.

4.3. Text input

Text may be inserted into the message in the composition region in several ways. The editor displays a blinking caret in the position in which text will be inserted whenever the editor is ready for text insertion. In the modal editor, the blinking caret is only displayed when in *insert mode*. In the modal editor, you must enter insert mode by giving one of the three start-of-type-in keyboard commands, A, I or R. In the modeless editor, the caret is always blinking; it is essentially in insert mode all of the time.

When type-in is acceptable, character insertion may be performed simply by typing characters from the keyboard. The characters will be entered into the message in the composition region at the blinking caret, and Laurel will continuously format and display the entire message to accomodate the new type-in.

4.3.1. Special characters

Several characters have special meanings during type-in:

BS, CTRL-A (either editor): Back up over (erase) the previous character. Characters of newly typed-in text that are erased by BS or CTRL-A are not recoverable via U (CANCEL, COM-U) (section 4.4). In the modal editor, BS or CTRL-A will not back up beyond the original insertion point for this sequence of type-in. In the modeless editor, BS or CTRL-A will back up from any position regardless of the original insertion point.

SHIFT-BS, SHIFT-CTRL-A (modeless editor only): Erase the following character.

CTRL-W (BW on Alto II keyboard, modal editor only), (COM-BS, modeless editor only): Back up over (erase) the previous word. This "word" is of the form A^+N^*, where A^+ is one or more alphanumeric characters (letters and digits) and N^* is zero or more non-alphanumeric characters (white space and punctuation.) This is different from, and consists of roughly two "words" as defined in section 4.2, *Word selection*. In the modal editor, CTRL-W will not back up beyond the original insertion point for this sequence of type-in. In the modeless editor, CTRL-W will back up from any position regardless of original insertion point.

SHIFT-CTRL-W, SHIFT-COM-BS (modeless editor only): Erase the following word. Here a word is defined as N^*A^+, with N^* and A^+ defined as above.

ESC, DEL (modal editor only): Terminates insertion. Insert mode is terminated; command mode is entered.

TAB: Laurel 6 supports tabs. There are twelve equally spaced tab stops (in Bravo parlance, 40 point plain tabs) across one screen line. A TAB will insert white space up to the next tab stop from the point at which the caret is blinking. Fine point: A TAB must be at least 5 points wide, the width of a space, to make selection of tabs reasonable. A TAB inserted less than five points before a tab stop will tab over to the next tab stop. Although TAB's may be inserted across line breaks in Laurel, it is recommended that they be used only on lines preceded by a CR. Otherwise, subsequent insertions can rearrange these tabs in unexpected ways, and in addition, hardcopy may not correspond to the display.

LF: The LF key in the modal editor inserts a Line Feed character into your message. There is generally no need to use this character, as Laurel displays new lines based on CR or line breaks alone. In the modeless editor, this key is actually a separate command key (PASTE), therefore it should be avoided during type-in.

4.3.2. Shifted selection

A special kind of selection, called *shifted selection*, is available as an alternative to type-in whenever type-in is allowed (either editor). To make a shifted selection, hold down either SHIFT key, make a selection in either the message display region or in the message composition region according to the standard conventions *while holding the SHIFT key down*, and then lift up on the SHIFT key. The selection will be highlighted with a dotted underline as you make it. When you lift up on the SHIFT key, the dotted underline will disappear, and a copy of that text will be inserted at the blinking caret.

In either editor, no ESC is necessary to confirm the shifted selection. Lifting up the SHIFT key is all that is required. Also, insert mode is not terminated by using shifted selection. You may proceed to type or perform another shifted selection immediately. You may find intermixing type-in and shifted selections during a single insertion sequence to be very convenient. All selection rules apply while making a shifted selection, in particular, multi-clicking for selection expansion is allowed.

In the modeless editor, a replace selection will be replaced by shifted selection just as it is replaced by type-in.

If you start to make a shifted selection by mistake, you may cancel that shifted selection any time before lifting up the SHIFT key. Just press the DEL or CANCEL (modeless) key while the SHIFT key is still down. The dotted underline will disappear, and no text copying will take place.

4.3.3. Secondary selection (modal editor only)

A holdover from the previous version of Laurel (and from Bravo) is *secondary selection*. This kind of text selection is available only in the modal editor. It is similar to shifted selection except that

a) It may be performed only immediately after issuing an A, I, or R command,

b) The SHIFT key is not held down,

c) You must type an ESC to confirm the secondary selection, and

d) You leave insert mode and return to command mode after the ESC.

Shifted selection is in almost all respects superior to secondary selection. Secondary selection has been retained to preserve compatibility with the previous Laurel editor.

4.3.4. Delete selection (modeless editor only)

Replace selections wait for type-in before the selected text is deleted; they may be cancelled by reselecting somewhere else. A delete selection can be made by holding down the CTRL key while making a selection. In this case, the selection is indicated by video reversal regardless of the mouse button used for making the selection, and when all buttons go up, the text is deleted immediately. This gives a very rapid means for deleting several pieces of text in a message.

If you start to make a delete selection by mistake, you may cancel that selection any time before lifting up the CTRL key. Just press the CANCEL key while the CTRL key is still down. The video reversal will be replaced by a solid underline, and no deletion will take place.

4.3.5. Move selection (modeless editor only)

A logically elegant kind of selection called *Move selection* is available in the modeless editor only. CTRL-select performs a delete selection, i.e., it is deleted immediately after lifting up the keys. When coupled with SHIFT-select, which means that the selected text is copied to a insertion point, we get CTRL-SHIFT-select, or move selection, which deletes the text from its current position and inserts it at the insertion point.

Before making a move selection, verify that the insertion caret is in the correct place for inserting the text to be moved. Push down both the CTRL and SHIFT keys, make the selection of text to be moved, and lift up the CTRL and SHIFT keys. When the move selection is made, the selected text will be highlighted by video reversal (as with CTRL-select) with a "serated bottom" (as with SHIFT-select). When the CTRL and SHIFT keys are raised, the selected text will be deleted and inserted at the insertion caret. The insertion caret will blink at the point just after this newly inserted text, and you may continue type-in.

If you start to make a move selection by mistake, you may cancel that selection any time before lifting up the CTRL and SHIFT keys. Just press the CANCEL key while the CTRL or SHIFT key is still down. The serated bottom video reversal will disappear, and no move will take place.

4.4. Keyboard Commands

There are many other functions available in the Laurel editors. These functions are present in both editors, but the ways in which they are invoked differ substantially. This section first describes how the commands are invoked in either editor and then lists each function in alphabetical order of command character. Those commands invoked by special keys (other than the alphanumeric keys) are listed last. Each command description will give the command letter with which it is invoked in the modal editor first, with the letter(s) with which it is invoked in the modeless editor given in following parentheses. For example, the Time command is described as the T (COM-T) command.

Modal editor command invocation

The modal editor accepts keyboard input in two different modes, command mode and insert mode. At the beginning of a Laurel session, the editor is in command mode, i.e., any character typed will be interpreted as an editor command rather than as type-in. Three commands--A, I, and R--put the editor into insert mode. When in insert mode, a blinking caret appears, and all type-in is treated as characters to be inserted into your message until an ESC or DEL character is struck. At that time, insert mode is terminated and command mode is re-entered.

It is possible in Laurel 6 to invoke screen commands, i.e., those invoked via screen menus and the mouse, even when in insert mode. When the screen command has completed, you will find the caret still blinking, and type-in will continue to be accepted.

Modeless editor command invocation

The modeless editor uses more function keys than does the modal editor. These names of these keys and their positions on the Alto keyboards are given in the following table:

Key name	Microswitch (Alto I) keyboard position	ADL (Alto II) keyboard position(s)
DO	ESC key	ESC key
CANCEL	Top right blank	BW key
NEXT	Middle right blank	Fourth right blank key from the top
COM	Lower right blank	Lower left, upper right, and lower right blanks
PASTE	LF key	LF key
DEL	DEL key	DEL key

You may find it convenient to write the key names in pencil next to the corresponding keys on your keyboard or to tape these key names onto these keys. If you have a BravoX keyboard, these keys are already labelled properly except for the PASTE key, which is labelled MOVE.

Since there is no special command mode in the modeless editor, a special way of typing command letters is necessary to distinguish commands from ordinary type-in. There is no problem when using one of the special keys, e.g., DO, CANCEL, NEXT, PASTE, or DEL; these are always interpreted as commands. For ordinary characters, this is accomplished with the COM key. The COM key (any one of the COM keys) is used like a SHIFT key--hold it down while you type the specified command letter. When a character is "COM shifted", the modeless editor understands that this character is to be treated as a command rather than as type-in. COM key type-ahead is not supported.

In the modeless editor, there is always an insertion point represented by the blinking caret. After typing in text but before making a new selection, there is no underlined (selected) text. In this case, the selection is considered to be the zero-length selection at the insertion point. Several commands, e.g., the bracket commands (section 4.4.16), that work on the selected text will operate on this null selection when no text is underlined.

4.4.1. A (COM-A) Append text

The A (COM-A) command places the insertion point (blinking caret) *after* the selected text in the message composition region.

In the modal editor, this command puts the editor into insert mode. Until ESC or DEL is struck, characters will be inserted at the insertion point as described in section 4.2, Text Input. Also, before typing any characters you may make a secondary selection.

In the modal editor, if the first character typed after the A command is ESC, then the previous text insertion is copied to the insertion point; if the previous command was a deletion, then that deleted text is copied to the insertion point.

In the modeless editor, the insertion point is merely moved to be *after* the selected text. This command is almost never needed in the modeless editor unless you want to insert after a final CR in a message.

4.4.2. B (COM-B) Placeholder brackets

The matching black triangles that surround words in the skeleton forms provided by **New form**, etc., are called *placeholder brackets*, and the complete word including these triangles is called a *placeholder*. The B (COM-B) command replaces the selected text with a copy of itself surrounded by these placeholder brackets.

In the modeless editor, the insertion point is placed to the *left* of the right bracket thus inserted. If there is no underlined text, and the COM-B command is given, then the null text at the insertion point is replaced by these placeholder brackets, the insertion point is between them, and subsequent type-in fills in the placeholder.

The B (COM-B) command is implemented as a replacement of text. A subsequent ESC (DO) will *not* bracket new text but will instead replace the selected text by the text inserted by the previous B (COM-B) command. This is almost never what you want; use another B (COM-B) to repeat the command instead.

4.4.3. D (DEL, COM-D) Delete text

The D (DEL, COM-D) command deletes the selected text from the message in the composition region. This text is not yet gone forever; it may be recovered with a U (CANCEL, COM-U) command, an I ESC or A ESC, or a (PASTE) command.

In the modeless editor, this command is generally called the DEL command. COM-D is allowed to preserve the command spellings from the modal editor. See also the description of replace selection, section 4.2.12.

In the modeless editor, a DEL command leaves the caret blinking at the spot from which the selected text was removed. Thus, to replace text, you may select the text to be replaced, type DEL, and then type in the new text. The editor understands that if no target selection is made between the DEL command and subsequent type-in (including shifted selection), then the entire sequence is treated as one replacement for purposes of CANCEL or DO (see sections 4.4.12 and 4.4.13).

In the modeless editor, yet another method of deleting text is available, the CTRL-SHIFT tap. You perform a CTRL-SHIFT tap by quickly depressing and releasing both the CTRL and SHIFT keys together. The selected text is briefly video reversed, and then is deleted. A timeout (the Tap: parameter in the Laurel profile, section 5) is checked; if the CTRL-SHIFT copmbination is held down longer than the timeout, then the CTRL-SHIFT has no effect. The CTRL-SHIFT tap is thus a "left hand delete" key for modeless editor users.

4.4.4. E (COM-E) Select everything

The E (COM-E) command merely selects the entire message in the composition region. The selection is made at the character level.

In the modeless editor, the selection is a replace selection.

4.4.5. F (COM-F) Find

The F (COM-F) command initiates the **Find** screen command. This screen command, when visible, is located on the second line of the lower menu. F (COM-F) will always reset the lower menu so that the **Find** command occupies that position. When the **Find** command is visible, it may also be invoked with the mouse, in addition to this method of invoking it through the keyboard.

When the **Find** command is invoked with RED, or via the F (COM-F) command, it blinks a caret in the following **{**brackets**}**. A prompt will appear in the feedback region to remind you of the special characters that may be used in the pattern to be typed into these **{**brackets**}**. Once the pattern is typed into these **{**brackets**}** and is confirmed, the **Find** command will search for that pattern in the text following the current selection (or insertion point in the modeless editor if no text is underlined). If a match is found, it is selected and the message is normalized so that this match is visible. In the modeless editor the selection is a replace selection.

The ESC (DO) command (section 4.4.13) does *not* repeat a **Find** command. ESC (DO) always repeats the last text *modification*. To repeat the same **Find** command, point the cursor at **Find** and click BLUE. An equivalent of the Bravo Y(es) command is thus ESC (DO) followed by BLUE click of **Find**. The ESC (DO) does the text replacement, and the BLUE click of **Find** does the subsequent search.

The patterns accepted by the **Find** command may be simple or somewhat complex. When the **Find** command is initiated, a prompt appears in the feedback region giving an abbreviated list of the special characters available and their meanings. You may never need the more complex patterns, but a complete list follows.

Simple character (other than the special characters listed below): Match that character in either upper or lower case.

’ (Single quote): Match the next character in the pattern exactly. Use the single quote to override the upper or lower case feature or to match a special character (including single quote!) All of the special characters are shifted keyboard top row characters (except {, }, and ’). If you wish to match any such character it is a good idea to precede it with a single quote.

#:	Match any single character (as in the Alto Executive).
*:	Match any sequence of characters (including the trivial sequence of no characters, as in the Alto Executive). The phrase "any sequence" here and in the following means "the shortest such sequence of characters that allows the rest of the pattern to match".
@:	Match any single alphanumeric character (letter or digit).
&:	Match any sequence of alphanumeric characters (including the trivial sequence of no characters).
!:	Match any single non-alphanumeric character (punctuation, blank, special characters, etc.).
~:	Match any sequence of non-alphanumeric characters (including the trivial sequence of no characters).
{:	Make the resulting selection start at this position, rather than at the start of the matched pattern. This special character is not used to match any characters in the text.
}:	Make the resulting selection end at this position, rather than at the end of the matched pattern. This special character is not used to match any characters in the text.

These last two special characters ({ and }) allow you to specify *context* in a **Find** command search. If you have ever been bothered by issuing a search command in some editor, for say "and", only to have that editor grin back at you with the last three letters of the word "command" selected, then search context is for you. An isolated word such as "and" is always surrounded by non-alphanumeric characters in text. Consider the pattern:

    !{and}!

Each ! must match one non-alphanumeric character in the text. Thus the pattern !and! would match only an isolated "and" together with its surrounding characters. The curly braces are added to the pattern to limit the resulting selection to only the word "and". Note that if you wish to find only lower case "and" words, then you must precede each of the a, n, and d characters by a single quote as in:

    !{’a’n’d}!

At most one pair of curly braces may be used in any pattern, and if both are used, then the left curly brace must precede the right curly brace.

By combining curly braces and various special characters in a pattern, quite complicated and precise searches may be specified. However, simple patterns involving no special characters should suffice for most applications of the **Find** command.

4.4.6. G (COM-G) Get file

The G (COM-G) command is used to replace the selected text with the contents of a specified file. This editor keyboard command differs from the **Get** screen command in that only the selected text is replaced, not the entire message.

When you issue the G (COM-G) command, the **Get** screen command acquires a gray background, the *File name* **{**brackets**}** appear, and a blinking caret appears in those **{**brackets**}**. Unlike Laurel 5.1, the brackets are not initialized with the selected text. At this point, you may fill in the name of the file whose text will replace the selected text in the message in the composition region. When the **{**brackets**}** are filled in and confirmed, the replacement will take place. In the modeless editor, if no text is selected, then the file contents are merely inserted at the blinking caret.

The same restrictions on length of file and text files only from the **Get** screen command apply to the G (COM-G) command as well. See section 3.5.1.

4.4.7. I (COM-I) Insert text

The I (COM-I) command places the insertion point (blinking caret) *before* the selected text in the message composition region.

In the modal editor, this command puts the editor into insert mode. Until ESC or DEL is struck, characters will be inserted at the insertion point as described in section 4.2, Text Input. Also, before typing any characters, secondary selection is available.

In the modal editor, if the first character struck after the I command is ESC, then the previous text insertion is copied to the insertion point, or if the previous command was a deletion, then that deleted text is copied to the insertion point.

In the modeless editor, the insertion point is merely moved to be *before* the selected text.

4.4.8. P (COM-P) Put file

The P (COM-P) command is used to write the selected text onto a specified file. This editor keyboard command differs from the **Put** screen command in that only the selected text is written, not the entire message.

When you issue the P (COM-P) command, the **Put** screen command acquires a gray background, the *File name* **{**brackets**}** appear, and a blinking caret appears in those **{**brackets**}**. At this point, you may fill in the name of the file to be written with the selected text in the message in the composition region. When the **{**brackets**}** are filled in and confirmed, the writing will take place. A confirmation is necessary if the file already exists.

As with the **Put** screen command, only text is written; for binary files see the **Copy** command, section 3.5.3.

4.4.9. R (COM-R) Replace text

The R command in the modal editor is used to replace the selected text by type-in (including shifted selection) or by secondary selection. The R command puts the modal editor into insert mode. See section 4.2, Text Input.

In the modal editor, if the first character struck after the R command is ESC, then the last text inserted or the last text deleted if the last command was D is inserted to replace the selected text.

The (COM-R) command in the modeless editor is provided primarily for compatibility with old modal habits. It is equivalent to the (DEL) command in that it deletes the selected text and blinks the caret at the vacated point. With replace selection available in the modeless editor, the (COM-R) command is never needed.

4.4.10. S (COM-S) Substitute

The S (COM-S) command initiates the **Substitute** screen command. This screen command, when visible, is located on the second line of the lower menu. S (COM-S) will always reset the lower menu so that the **Substitute** command occupies that position. When the **Substitute** command is visible, it may also be invoked with the mouse, in addition to this method of invoking it through the keyboard. The second line of the lower menu will look like this:

Substitute **{**new text**}** *for* **{**pattern to be replaced**}**

Prior to invoking the **Substitute** command, you should select the range of text in which substitution is to take place. When the **Substitute** command is invoked with RED, or via the S (COM-S) command, it blinks a caret in the following **{**brackets**}**. This should be filled in with the exact text to be written into the message. When this argument is filled in and confirmed, the caret will blink in the *for* **{**brackets**}** and a prompt will appear in the feedback region to remind you of the special characters that may be used in the pattern to be typed into these **{**brackets**}**. Once the pattern is typed into these **{**brackets**}** and is confirmed, the **Substitute** command will search for that pattern within the selected text of the message in the composition region. Each sequence of text within the selection that matches the pattern given in the *for* **{**brackets**}** will be replaced by the text in the initial **{**brackets**}**. After a pattern match is found and the selected portion replaced by the text in the initial **{**brackets**}**, the search for the next match within the text proceeds with the first character following the text just matched. Thus, substitution within newly replaced text cannot occur in a single invocation of the **Substitute** command.

The special characters and pattern specifications allowed in the *for* **{**brackets**}** of the **Substitute** command are the same as those allowed by the **Find** command (section 4.4.5).

The results of the **Substitute** command are treated as if one large replacement of text over the initial selection had occurred. Thus U (CANCEL) (section 4.4.12) works properly. The ESC (DO) command (section 4.4.13) will repeat this replacement on subsequently selected text. This is generally not what you want. To repeat the same **Substitute** command, select the text in which you want the Substitute command to operate, point the cursor at **Substitute** and click BLUE.

Unlike Bravo, the **Substitute** command cannot be terminated by typing DEL. However, since this command can be completely undone with U (CANCEL) after it has finished, the lack of a premature termination mechanism should not be a hardship.

4.4.11. T (COM-T) Time insertion

The T (COM-T) command inserts a text representation of the current time into the message in the composition region. This representation is

<day number> <month> <year> <time of day> <time zone> (<day of week>)

as in 5 May 1981 3:23 pm PDT (Tuesday).

This is the same format as the time inserted into the Date: field by Laurel during message delivery. (This representation is slightly different from that used in the time area at the upper right of the Laurel screen.)

In the modal editor, the time is inserted at the start of the current selection. In the modeless editor, the time is inserted at the insertion point (blinking caret), and the insertion point is moved to just after the newly inserted time. If the selection is a replace selection, then the selection is replaced by the time.

Repeating this command with ESC (DO) will result in the same text, which may not be the most current time.

4.4.12. U (CANCEL, COM-U) Undo or cancel last text modification

The U (CANCEL) command is used to restore the text in the message composition region to the state it had before the most recent text modification. It should be used when you detect an editing mistake that cannot be corrected simply by backspacing.

U (CANCEL) will only go back one level, and its invocation becomes the next text modification. A sequence of invocations of U (CANCEL) will thus cycle between the state of just before the first U (CANCEL) in the sequence and the state as of before the text modification preceding the first U (CANCEL) in the sequence.

U (CANCEL) in Laurel differs from the U(ndo) command in Bravo in that Laurel ignores any new selection position in performing this operation. The previous text modification is undone exactly in the place where it was originally done. Thus, U (CANCEL) cannot be used to move text. There are several other ways to move text (modal D, select, I ESC, etc.; modeless CTRL-SHIFT-select; DEL, PASTE, etc.).

In the modal editor, the last text modification is easily defined. It is the last deletion, the last insertion, or the last replacement.

In the modeless editor, the last text modification is less easily defined, since there is no command mode giving a convenient demarcation of the boundaries between text modifications. In general, any time a solid underline appears, a text modification boundary is created. It is not necessary to puzzle out exactly what text will be restored when you issue a (CANCEL) command; it nearly always does the right thing. When the (CANCEL) command errs, it generally does so on the side of restoring more text than you might have expected. This is considered to be much less serious than restoring less than you expected.

There are pathological situations when CANCEL may remove as much as it restores. The sequence CANCEL, PASTE will place all the restorable text that Laurel has remembered onto the screen, from which point you may edit the text as you wish.

Note that newly typed-in text erased by backspacing (with BS, CTRL-A or CTRL-W (COM-BS)) is not restorable via U (CANCEL). However, in the modeless editor, text deleted by backspacing before the original insertion point of a text modification is restorable by CANCEL.

4.4.13. ESC (DO) Repeat last text modification

The last text modification is either an insert, an append, a replace, or a delete. The ESC (DO) command repeats this text modification in the position given by the current selection. The DO key is the same as the ESC key.

In the modal editor, an insert is repeated by inserting the same text before the current selection; an append is repeated by appending the same text after the current selection; a replace is repeated by replacing the current selection by the same text inserted previously; and a delete is repeated by deleting the current selection.

In the modeless editor, a deletion followed immediately by an insertion with no target selection in between those two actions is treated as a replacement. This includes deletion due to replace selection. A subsequent DO command will replace the selected text. Text input that follows a non-replace target selection is always treated as an insertion. A subsequent DO command will insert the same text at the blinking caret. A DO command following a simple deletion (with DEL or CTRL-select) will delete the selected text.

In the modeless editor, the DO command is ignored if no solid underline exists. This prevents an annoying behavior that might occur when the DO key is pressed inadvertently at the end of a type-in sequence. Since modal editor habits die hard, when you switch to the modeless editor, you may find yourself typing these spurious DO's quite a bit. Laurel will merely ignore them. However, when the mouse is used to place a new selection on the screen, even if the insertion point does not move, then a subsequent DO will take effect.

4.4.14. (NEXT) Select next placeholder

The NEXT command is available in the modeless editor only. It searches forward in the text from the current selection (or insertion point if there is no selection) for the next placeholder (text surrounded by matching triangles). This next placeholder, if found, is selected with replace selection.

The NEXT command gives a very convenient means for filling out new forms, etc. When you have finished replacing the first placeholder in a form (selected for you by the form initializing command), press NEXT and continue typing the replacement for the next placeholder.

SHIFT-NEXT behaves just like the NEXT command, except that it searches backwards in the message for the previous placeholder. Type-ahead of SHIFT-NEXT is not supported.

4.4.15. (PASTE) Insert previous deletion

The PASTE command is only available in the modeless editor. (The PASTE key is the same as the LF key.) When invoked, PASTE inserts at the current insertion point the text deleted by the previous text modification. This allows one to move text by selecting it, typing DEL, selecting the new position, and typing PASTE. Repeated PASTE commands are possible; just re-select and type PASTE.

If this method of moving text raises too much anxiety due to the text disappearing during this sequence of commands, then you may wish to use move selection, section 4.2.

For abstruse, yet logical, reasons, a CANCEL given after a PASTE usually results in no change in your text. This is not a disaster; if you wish to remove the text thus pasted, select it and type DEL.

4.4.16. ', ", <, (, [, {, - (COM-', COM-", etc.) Bracket commands

These commands surround the selection with the particular matching pair of symbols suggested by the character typed. For example, [(COM-[) surrounds the selected text with matching square brackets. The hyphen - (COM--) command is special it surrounds the selected text with short lines of several hyphens, similar to the way that the Forward command sets off different sections of text.

In the modeless editor, the insertion point is placed to the *left* of the right bracket thus inserted. If there is no underlined text, and one of these bracket commands is given, then the null text at the insertion point is replaced by the appropriate brackets, the insertion point is between them, and subsequent type-in goes between the bracketing symbols.

These commands are implemented as replacement of text. A subsequent ESC (DO) will *not* bracket new text but will instead replace the selected text by the text inserted by the previous bracket command. This is almost never what is wanted; use another bracket command instead.

4.4.17. 0, 1, ... 9 (COM-0, COM-1, ... COM-9) Shortcut screen rearrangement

These commands 0 (COM-0) through 9 (COM-9) were included in Laurel due to a request by a well respected but particularly indolent Laurel user who found the boundary movement boxes on the Laurel screen too bothersome. These commands rearrange the regions on the Laurel screen to various pre-set positions by typing one keyboard command.

By default, 0 (COM-0) resets the screen to roughly the same positions as Laurel uses at start-up, 1 (COM-1) sets the screen so that the table-of-contents region is as large as possible, 2 (COM-2) sets the screen so that the message display region is as large as possible, and 3 (COM-3) sets the screen so that the message composition region is as large as possible. Also by default, 4 (COM-4) through 9 (COM-9) are the same as 0 (COM-0).

The screen arrangements selected by each of these commands may be tailored to your needs by entries in the Laurel profile. See section 5 for more details.

5. The Laurel profile

5.1. The Laurel profile file

Laurel obtains certain configuration information and options from your *Laurel profile*, contained in the file Laurel.Profile (or in User.cm) on your Alto disk. A standard Laurel profile is installed when you obtain Laurel using the LaurelNewUser.cm command file (section 1.3). The only thing in the Laurel profile you are likely to want to change immediately is the name of your printing server. For completeness, however, nearly all possible options are documented here. The only mandatory field in the profile is "Registry" (see below).

If there is no Laurel.profile file on your Alto disk, then Laurel will look in your User.cm file for a subsection beginning with [Laurel]. All options listed below may be listed in this subsection of User.cm if you prefer. In addition, if the normal profile options are read from User.cm, then Laurel will use the printer name listed in the [Hardcopy] subsection of User.cm in the Press: *PrinterName* entry for your default print server name and the name listed in the PrintedBy: *Name* entry of that subsection as the name to appear on your hardcopy output. As the Laurel profile may be contained in either Laurel.profile or in User.cm, this manual refers merely to the *Laurel profile*, letting you decide which actual file it is in.

Laurel configures itself automatically according to the Laurel profile when Laurel starts up, if it perceives that there have been changes in your Laurel profile or in several other files, e.g., Fonts.widths. No explicit installation is ever required. If you have edited your Laurel profile using Laurel, you must **Quit** and restart Laurel before the changes will take effect. During startup, if Laurel does reconfigures itself, it will display a "coffee cup" cursor, indicating that startup may take slightly longer than usual.

The Laurel profile is a text file containing a sequence of *parameter lines*, each of which has the following form:

name: *value* CR

All lines in the profile must adhere to this format.

The *name* must start with the first character on a line, and is terminated by the first colon. Do not precede the colon by blanks; otherwise the *name* will not be recognized. The *value* may be preceded by any number of blanks and begins at the first non-blank character. Subsequent characters may be blank and are included in the *value*. The *value* is terminated by the CR at the end of the parameter line.

A typical Laurel profile for use at PARC might be as follows:

```
Registry: PA
Printer: Clover
```

This example leaves out many possible parameter lines. Parameters that are missing from the Laurel profile are given default values by Laurel. Parameter names may be spelled in any combination of upper and lower case letters.

5.2. Profile entries

Registry: *RegistryName*

> The name of the registry in which you are a registered mail system user. This field is *required* in your profile; it cannot be defaulted. This is the only required profile field.

The following Laurel profile parameters are listed in alphabetical order.

Background: *color*

> Controls the video presentation. *color* may be either the word black (white letters on black background) or the word white (black letters on white background).
>
> (Default: white)

BluePendingDelete: *TrueOrFalse*

> (Meaningful only if Editor: Modeless is also present in the Laurel profile.) If the *value* is TRUE, then when making text selection extensions with the BLUE mouse button, the resulting selection becomes a replace selection (section 4.2.12). If enabled, a simple way to select an item with replace selection is to select it with RED or YELLOW, then immediately re-select it with BLUE.
>
> (Default: FALSE)

Boundary: *BoundaryCommandNumber, TocLines, DMLines, CMLines*

> The Boundary entries specify how the Laurel screen is to be configured when the 0, 1, 2, ... , 9 (COM-0, COM-1, COM-2, ... COM-9) commands are given (see section 4.4.17). The four parts of the *value* (all numbers) are separated by commas, and all parts must be present.
>
> The *BoundaryCommandNumber* must be in the range from 0 through 9, and it identifies for which boundary adjustment command the following numbers apply. The *TocLines, DMLines, CMLines* numbers specify the *relative* sizes for the table-of-contents region, the message display region and the message composition region respectively. Examples:
>
> ```
> Boundary: 5, 1, 1, 1
> Boundary: 6, 1, 2, 3
> ```
>
> (Defaults:
> Boundary: 0, 12, 19, 16
> Boundary: 1, 1, 0, 0
> Boundary: 2, 0, 1, 0
> Boundary: 3, 0, 0, 1
> all others (Boundary 4 through 9) same as Boundary 0)

BracketDelay: *Milliseconds*

This entry, along with BracketTimeOut:, controls the speed at which **{**brackets**}** arguments cycle through filenames when the RED mouse button is held down over the associated screen command (see section 2.1.3). The *value* is the number of milliseconds that Laurel will pause between finishing displaying one filename in **{**brackets**}** to beginning to search for the next filename.

(Default: 750)

BracketTimeOut: *Milliseconds*

This entry, along with BracketDelay:, controls the speed at which **{**brackets**}** arguments cycle through filenames when the RED mouse button is held down over the associated screen command (see section 2.1.3). The *value* is the number of milliseconds that Laurel will wait from the time that RED is held down until beginning to search for the first filename. If the *value* is 0, then the bracket filename search feature is disabled.

(Default: 750)

Click: *Milliseconds*

This entry is the timeout that Laurel uses to determine whether a mouse click has been performed soon enough after the previous mouse click to be considered a multiple click for the purposes of selection expansion (see section 4.2.9). The *value* is a number expressed in milliseconds. If the *value* is 0, then the multiple click selection expansion feature is disabled.

(Default: 390)

Comment: *text* or
C: *text*

The *text* is ignored by Laurel. Comment lines are merely explanatory text for the user.

Copies: *Number*

This entry specifies the default number of copies to be printed by the **Hardcopy** command (see section 3.3.3). This default may be overridden by explicit use of the **Copies** command in the hardcopy submenu. The *value* must be a number in the range 1 through 99.

(Default: 1)

CopiesField: *c*

For those people who prefer to send messages with a c: header field rather than a cc: header field, this entry may be used. If the *value* is the letter c, then a c: field is provided in all default message forms. Any other *value* results in a cc: field provided in the default forms. Caution: Arpanet message systems in general do not support the use of a c: field.

(Default: cc)

DefaultHardcopyForm: *HardcopyFormName*

This entry specifies the default hardcopy form on which messages are printed by the **Hardcopy** command (see section 3.3.2). This default may be overridden by explicit use of the **Form** command in the hardcopy submenu. Caution: do not use this entry with a *value* of InternalMemo unless you have a valid reason for doing so (see section 6, *Hardcopy forms*).

(Default: Blank)

DisplayAfterDelete: *TrueOrFalse*

If the value is TRUE or YES, then whenever the **Delete** or **Move to** commands are invoked, the next message in the mail file will be displayed in the message display region as long as:
a) Only one message is deleted or moved,
b) That message is the one currently displayed,
c) A subsequent message exists in the mail file, and
d) That subsequent message is marked with a "?".

This option can make processing newly arrived mail a bit easier.

(Default: FALSE)

Editor: *Mode*

If the value is the word Modeless, then the modeless editor is used rather than the modal editor.

(Default: Modal)

ErrorKeys: *LaurelX*

If the value is the word LaurelX, then some feedback region prompts are phrased slightly differently (in the LaurelX style). These prompts also name the modeless editor key names, i.e., DO rather than ESC, etc.

(Default: Normal)

Font: *Number FamilyName PointSize Face*

Font entries define the specific printer fonts to be used when hardcopying messages. The four parts of the *value* of a Font: entry are separated by spaces. The *Number* must be present and must be in the range from 0 through 9. This *Number* identifies the font as referenced in hardcopy forms (see Appendix C). The *FamilyName* and *PointSize* must be present; the *Face* is optional. If the *Face* is missing, then regular is assumed. Examples:

```
Font: 5 TimesRoman 14 BI
Font: 9 HyType 11
Font: 7 Helvetica 36 B
```

The standard InternalMemo form built into Laurel uses Font 1 for header field names, Font 2 for the herald line, Font 3 for the company logo, and Font 0 for everything else (including the text).

(Defaults:
Font: 0 TimesRoman 10
Font: 1 TimesRoman 8
Font: 2 TimesRoman 12 B
Font: 3 Logo 24
all others (Fonts 4 through 9) same as Font 0)

FromField: *ScreenPoints*

The right edge of the From: field display in the table-of-contents region may be set by using this entry. The value is a number in units of Alto resolution screen dots from the left margin of the Laurel screen. See also the SubjectField: and SubjectFieldExtension: profile entries. Note: the fixed left edge of the From: field is 110.

(Default: 250).

Hardcopy: *PrinterName*

A synonym for the Printer: *PrinterName* entry (see below).

HardcopyForm: *HardcopyForm*

There may be many of these entries in the Laurel profile. Each entry specifies a format in which messages are printed by the **Hardcopy** command. If a HardcopyForm *value* specifies a form with the same name as one of the built-in hardcopy forms, then the form mentioned in the Laurel profile replaces the standard form. Building hardcopy forms is complicated--see Appendix C for details.

HomoTollens: *TrueOrFalse*

The CTRL-select feature in the modeless editor behaves quite differently than does CTRL-select in the Tools Environment. If you use that system, and you find the differences in CTRL-select bothersome, then use this entry to declare yourself a Tools user by setting the *value* to TRUE or Yes. If you do so, then the CTRL-select feature is disabled in the Laurel editor.

(Default: FALSE)

LaurelSupport: *name*

Do not use this entry unless advised to do so by your local network administrator.

The *value* is the name to which Laurel's error reports are sent by the automatic error reporting mechanism (see section 7).

(Default: LaurelSupport.PA)

NewFormAfterDelivery: *TrueOrFalse*

> If the value is TRUE or YES, then whenever the **Deliver** command completes successfully, the text in the message composition region is replaced by a new skeleton form just as if the **New form** command had been invoked. U (CANCEL) will undo this replacement if you wish to retain the delivered text. This option has been found useful by users who sometimes forget whether the text in the composition region has been sent.
>
> (Default: FALSE)

PrintedBy: *name*

> A line of text that will be printed on hardcopy break pages to identify the person making the hardcopy. Any $ characters will be expanded to your logged-in name at the time you generate the hardcopy.
>
> (Default: $)

Printer: *PrinterName*

> The default name or network address of the hardcopy server machine to which all hardcopies will be sent. This *PrinterName* can be overriden using the **Printer** comand in the hardcopy submenu.
>
> (Default: an illegal server name)

Private: *YesOrNo*

> This entry specifies the default of the **Private** option used by the **Hardcopy** command (see section 3.3.5). This default may be overridden by explicit use of the **Private** command in the hardcopy submenu. The *value* may be either Yes or TRUE to set this parameter to Yes. All other *value*s are equivalent to No. Caution: printing with **Private** set to Yes involves complicated interaction at the printing server. Don't use this option unless you have a valid reason for doing so and you understand how to operate the print server in password protected mode.
>
> (Default: No)

Retrieve: *HostName*

> This entry is ignored if your registry is implemented by Grapevine servers. Unless your local network maintainer informs you otherwise, do not use this entry.
>
> If you are in an MTP registry and your mailbox is located on a server other than the default server for that registry, then use this entry to specify the name or network address of that server. (The default server for such registries has the registry name as a synonym for its more widely used name.)
>
> (Default: determined by your registry)

RunPath: *FilePrefix*

The string given as the *value* will be used as a prefix for a remote file when the **Run** command as asked to run a program that is not on your local disk (see section 3.6). This entry should be filled in with the server and directory prefix that you normally use to retrieve Laurel software, e.g., for PARC this would be

RunPath: [Ivy]<Laurel>

(Default: a null prefix)

ScrollDelay: *Milliseconds*

This entry, along with ScrollTimeOut:, controls the speed at which lines are scrolled up or down when the RED or BLUE mouse button is held down in the scroll bar (see section 2.2.1). The *value* is the number of milliseconds that Laurel will pause between successive single line scroll actions. Note: there is a threshold below which lower *value*s for this parameter have no effect, as you reach the maximum speed your hardware and software can handle.

(Default: 200)

ScrollTimeOut: *Milliseconds* (threshold)

This entry sets the amount of time that the RED or BLUE mouse button must be held down in the scroll bar before continuous scrolling occurs (see section 2.2.1). The *value* is the number of milliseconds that Laurel will wait from the time that RED or BLUE is held down until beginning to scroll continuously. If the *value* is 0, then the continuous scrolling feature is disabled. Note: this parameter should be chosen with care--high enough to avoid entering continuous scroll mode too quickly, but low enough to keep from having an excessive delay when you want continuous scrolling. Our testing has shown that users have a very wide range of preferences for this setting. The default has been deliberately chosen to be on the slow side of this range.

(Default: 1000)

Send: *Hostname*

This entry is ignored if you are using Grapevine mail sending protocols. Unless your local network maintainer informs you otherwise, do not use this entry.

The *Hostname* is the name or network address of the MTP maildrop server machine to which you will send all outgoing mail. Any mail system server can be used for sending mail by any user. Network address constants of the form *number#number#* must be used at sites where there is no gateway providing name lookup services.

(Default: determined by your registry)

SendMode: *SendMode*

Do not use this entry unless specifically instructed to do so by your local network administrator. In almost all cases, Laurel will figure out the correct value for this parameter automatically.

The only legal values are: Auto, MTP, or GV. This entry, if MTP or GV, will force Laurel to use that protocol when delivering messages. If the value is Auto, then Laurel will use its own algorithm to figure out the correct protocol to use.

(Default: Auto)

SubjectField: *ScreenPoints*

The left edge of the first line of the Subject: field display in the table-of-contents region may be set by using this entry. The value is a number in units of Alto resolution screen dots from the left margin of the Laurel screen. See also the FromField: and SubjectFieldExtension: profile entries. Note: the fixed right edge of the Subject: field is 480.

(Default: 260).

SubjectFieldExtension: *ScreenPoints*

The left edge of the second and third lines of the Subject: field display in the table-of-contents region may be set by using this entry. The value is a number in units of Alto resolution screen dots from the left margin of the Laurel screen. See also the FromField: and SubjectField: profile entries. Note: the fixed right edge of the Subject: field is 480.

(Default: 275).

Tap: *Milliseconds*

This entry is the timeout that Laurel uses to determine whether a CTRL-SHIFT tap has been performed fast enough to be considered a DEL command (see section 4.4.4, D (DEL, COM-D)). The *value* is a number expressed in milliseconds. If the *value* is 0, then the CTRL-SHIFT tap feature is disabled.

(Default: 390)

TwoSided: *YesOrNo*

This entry specifies the default of the **Two sided** option used by the **Hardcopy** command (see section 3.3.4). This default may be overridden by explicit use of the **Two sided** command in the hardcopy submenu. The *value* may be either Yes or TRUE to set this parameter to Yes. All other *value*s are equivalent to No.

(Default: No)

5.3. Profile errors

As you can see, the Laurel profile can be simple or as complicated as you want. There are several possibilities for error, so Laurel has a mechanism for reporting these errors to you. Should Laurel detect an error in your Laurel profile, it will ignore any command line switches, and instead will read a mail file named InstallErrors.mail in which Laurel has placed descriptions of all profile errors that it detected. Most error messages are simple to understand, and the corrections necessary should be straightforward. Error messages referring to user defined hardcopy forms fall significantly short of this standard.

Even if you have installation errors, Laurel will still operate. You should read through the message displayed, use the **Get** command to read your Laurel profile, make the appropriate changes, and use the **Put** command to rewrite that file. At this point, you should use the **Quit** command to return to the Alto Executive, and then restart Laurel.

Although Laurel continues to operate, when profile errors are detected, several of the entries may not take effect. This is particularly noticeable when the Editor: modeless entry is not accepted by Laurel, and you find yourself having to edit the profile in a modal manner.

You may notice that the one message in the InstallErrors.mail file is marked for deletion. When you **Quit** from Laurel, that file will be deleted, as is any mail file in which all messages are marked for deletion.

6. Message System Mores

The great art of living easy and happy in society is to study proper behaviour, and even with our most intimate friends to observe politeness; otherwise we will insensibly treat each other with a degree of rudeness, and each will find himself despised in some measure by the other.

--BOSWELL, London Journal (Dec.1, 1762)

What is this?

This section is an essay on manners, that is, message system manners. Laurel in its various releases has been in use for over three years at the time of this writing. In this time many patterns of message system user behavior have been discovered, and doubtless many more such patterns will be discovered in the future. This section gathers together several observations of Laurel user behavior in an effort to spread understanding of this new electronic message medium and to instruct users in proper behavior.

The contents of this section may be divided into roughly two kinds, objective observations of message system social phenomena and definitely biased suggestions of standards. The opinions expressed herein are solely those of the author. These opinions are not based on scientific studies or samples, but rather on certain gut feelings that have evolved through a close association with Laurel since its inception. I expect that several of the opinions set down here will receive vigorous debate, but so much the better to spread the word.

A brief outline of this section follows.

Communication patterns	A brief discussion of structures within which communication takes place.
The wrong number	What to do when you receive a message intended for someone else.
Rudeness and vulgarity	Why it appears in electronic mail.
Message system costs	How the way we pay for communication affects what we say.
Unsolicited mail	What it is and when it is or isn't appropriate.
The chain reaction	A description of a phenomenon peculiar to electronic mail. The Reply-To feature and how it helps.
Miscellaneous distribution list peccadillos	Distribution list etiquette.
Off-the-record responses	When and when not to publish.
Hardcopy forms	How to permanently engrave messages properly.
Masquerading	Anonymous (or worse) messages.
Wizards vs. naive users	How to keep arcana to yourself.
The moral of this tale	Be considerate of others.

Communication patterns

Part of the evolution of a society is the structure within which its members communicate. Face-to-face communication, both spoken and through gestures, has been with us for a very long time. Written communication and telephone communication have been employed for a substantially lesser amount of time. Nevertheless, these modes of communication have been around long enough to have developed certain standards of conduct and a framework in which reasonable communication can take place.

The electronic message medium has been available for a much shorter period of time, perhaps twenty or so years. I am purposely ignoring telegraphic communication, which has very different characteristics due to its long delays and high cost. Electronic message systems available on personal computers have been available for even less time, certainly less than ten years. In this time, standards of electronic communication have not yet had time to mature, so we are still groping toward a workable electronic messaging society.

In any of the mature communication media, each society places limits on what is considered acceptable behavior. Vulgar language or gestures are generally frowned upon in face-to-face communication, except in smaller sub-societies in which this mode of behavior is necessary to be part of the group. Shouting at close range is similarly considered to be in bad taste. Methods of dealing with such behavior in face-to-face communication run from mild rejection of the speaker to complete avoidance of that speaker in the future. As the number of human societies is large, and each has had much experience with this means of communication, the means employed for dealing with such situations are quite varied. Within each group, however, the methods used can be quite effective in stifling unwanted behaviors.

I will try to list several kinds of situations that arise in the electronic message medium and means for dealing with them. Where possible, I will try to draw parallels to other more traditional modes of communication to illustrate acceptable manners. In addition, I will try to point out the ways in which communicating via electronic mail is different from the traditional communication media, and how this modifies the problems to be dealt with.

The wrong number

We all have dialed wrong numbers and received calls from people who have dialed wrong numbers. The protocol for handling such situations is simple, and arises naturally as a result of the way in which standard phone calls are initiated. A typical wrong number dialog may be as follows:

Callee:	Hello.
Caller:	Hello. May I speak to John?
Callee:	There is no one at this number by that name. I believe you have the wrong number.
Caller:	Oh. Isn’t this 555-1234?
Callee:	No it isn’t. (And sometimes ...) This is 555-4321!
Caller:	Thank you. I’m sorry to have bothered you.

In postal communication, receiving misaddressed mail or mail for a former resident who has moved is akin to the telephone's wrong number. The post office's suggested remedy is for the recipient to line out the address and remail the letter. The post office will then attempt to forward the letter to the correct address, deliver it to the proper address, or return the letter to the sender.

Note that in both of these situations, it was not necessary to begin the actual conversation or open the letter. Enough information is exchanged at the outset to determine if the parties in the communication are the correct ones. This is usually not true when comunicating via electronic mail.

In electronic message systems, it is seldom the case that a message sent to a particular name is actually delivered to a recipient with a different name. A different situation is (unfortunately) common when a recipient has a popular name. The problem is that several people may have the same last name, and Laurel (plus Grapevine) has not had convenient facilities for mapping a person's actual name into that person's message system name. Thus, a person named Doe may receive mail for ADoe, BDoe, etc. Here, the original error is committed by the sender, who did not consider that ADoe's message system name was actually ADoe, but just assumed that it was Doe.

The parallel to this situation in the telephone medium is actually a bit more elaborate than the dialog given above. It is more like:

Callee:	Hello.
Caller:	Hello. Is Johnny there?
Callee:	Hold on, I'll get him.
John:	Hello?
Caller:	Hey Johnny, let's boogie on down to the hoedown.
John:	Who is this?
Caller:	Come on buddih, this is good old Bodine!
John:	I don't know any Bodine.
Caller:	Oh. Ain't this 555-1234?

and so on. Notice that in this case a partial name match has occurred, and it is only later in the conversation that one of the parties discovers that something is awry. In the electronic mail case, it is nearly always the case that the message must be at least partially read to determine that it has reached an incorrect recipient.

This situation can be (and has been) handled in several inappropriate ways. First, (and worst), the incorrect recipient can just ignore the message. No one gains through such inaction. Second, the incorrect recipient may send a response to the sender of the form "Stop sending me this trash!" This is a bit more helpful, but not quite the best that can be done. Third, the incorrect recipient may send the correct recipient a message of the form "Tell your senders what your name is!" This is not even as good as the previous response, as a message system user cannot know all possible senders.

Proper consideration by all involved can alleviate the "wrong number" syndrome considerably. Senders of messages should know their recipients. When sending a message, if you are not sure of a person's message system name, look it up. In Palo Alto, the phone list has everyone's message system name correctly listed. Other organizations should do the same, and eventually a message system wide "white pages" will be published. All these help, but not if the senders don't use these lists.

When you realize that a message is not for you, use the **Forward** command to send it back to the sender along with your polite comment that the message has reached a "wrong number". Forwarding the message back is important, as the sender may not have a copy of that message any more. Once you have determined that a you have received a "wrong number" message, STOP READING IT. The messages sent through the message system may have personal material, and it is none of your business to peruse the entire message. It is for this reason that I do not suggest forwarding the message to the proper recipient. Determining who is the proper recipient is the job of the sender. It is presumptuous to believe that you know who the proper recipient is; you may actually forward the message to yet another incorrect recipient. Besides, determining the correct recipient may require reading more of the message than you ought to read. (If you think you know the message system name of the correct recipient by the time you realize that you are not the correct recipient, then you might include that name in your short covering note back to the sender. However, the mistaken sender should not expect correct identification of the intended recipient, just as he or she would not expect it in the telephone or postal mail systems.)

Some further points to consider are these. The "wrong number" mishaps generally happen to people who have common names and whose system names are exactly their last names. The honor of having one's system name be exactly one's last name is generally historical ("I was the first Doe hired here, therefore I'm entitled to be Doe.pa forever!") A reasonable solution would be that no one have the plain name, but instead when ADoe arrives, then Doe has his or her message system name changed to BDoe (or whatever). In this way, the existing message system facilities will catch messages sent to Doe and return them as being sent to a non-existent name, at which point the sender can look up the correct message system name. (Note of course that the author has a relatively uncommon name, and makes these observations knowing full well that they don't apply to him.)

One final point: one often heard response to this and other problems is "Why doesn't Laurel do it?" The answer is that some of these societal questions have been addressed by Laurel, but many of them are so subtle that it would take a large amount of research into these problems before workable institutions could be built into such a system. Piecemeal solutions will be forthcoming (in the form of the "white pages" and some .laurel runnable programs). In the meantime, consideration for others can go a long way.

Rudeness and vulgarity

The electronic mail medium joins several disparate properties of other communication media in an interesting way. The display of mail on a personal computer is a rather personal experience. Certain feelings of privacy and ownership pervade a personal computer user's relationship with his or her machine. Thus, the process of reading one's own electronic mail includes many of the personal aspects of face-to-face communication.

On the other hand, sending electronic mail is much more impersonal. The recipient is not present, and nearly none of the social strictures that govern one's face-to-face comunication are present. The sender is also able to speak his or her piece completely, without any intervening exchanges with the recipients that might moderate the entire business. This situation is enhanced when the recipients are not named directly, but are addressed indirectly through an impersonal distribution list. This imbalance in feelings between sender and recipient has wide ranging consequences.

An obvious consequence of this imbalance is that opinions expressed and the language used to express them in messages can be wildly inappropiate to the customs and expectations of the recipients of such a message. A reader may justifiably feel slapped in the face by a message he or she considers to be in extremely bad taste.

When rebuked for such behavior, errant senders have been known to say "I didn’t intend it that way!" This is not good enough. The damage has already been done. The only remedy is for senders to think about what they are saying and to whom they are saying it. The message system to date has been fairly unrestricted. Only as long as the society of message system users practices self-restraint will such a free-wheeling communication medium be tolerated. There are several means of applying institutional censorship to the message system traffic, means that we hope will never need to be implemented.

Message system costs

Many of the problems associated with improper use of the message system are exacerbated (caused?) by the lack of charging for message system usage. In nearly all other modes of communication, "sending a message" implies a certain cost (or risk) which rises with the number of recipients that are being reached. Free speech is, in this sense, not free at all. Certainly in a free society, one can say what one pleases, but not without paying for the means to say it. Let me illustrate this with some examples.

In nearly every communication medium, costs for the use of that medium are borne by the sender of messages. Postal mail requires the sender to pay for a stamp for each copy of a message that is sent. Telephone service is charged to the originator of calls, and each call (in general) goes to only one recipient. Broadcasting messages via radio or television requires a large investment on the part of the sender. The costs of printing handbills or posters are likewise borne by their authors. Public speeches, if they are to reach a large audience, require use of sound systems, etc., that are paid for by the speaker.

It may be argued that recipients do pay some of the costs for using some of these systems. However, these costs (the price of a radio receiver, basic telephone service, etc.) are generally constant; they do not increase as received message usage increases. A receiver’s cost for electronic mail is similar in this respect in that the cost of a workstation on which Laurel runs is borne by the receiver.

Some other modes of communication do require explicit payment by the receiver. Commercial films, books, magazines and records fall into this category. However, publication of these materials does involve a substantial financial risk. Material that is not likely to be well-received is seldom published, and when it is, large costs are often incurred by the publisher.

Electronic mail as implemented in Laurel and Grapevine has a very different cost structure. The cost for a sender is minimal. It essentially consists of the time it takes to compose and send a message. If time is considered the major cost factor, then it is the recipients who pay dearly for the messages they receive. When the amount of time each recipient spends on a message sent to a large distribution list (even if a quick scan of part of the message followed by a **Delete**), is summed over all recipients, this is easily much more than the time consumed by the sender of that message.

While we would like to keep the free structure of a message system, where any user can send any message to any other users, this freedom must be used with some care. When electronic message systems become widespread, they will undoubtedly change their cost structures to match those of the more traditional communication systems.

Unsolicited mail

The existence of large public distribution lists in our message system makes it easy for a sender to reach a very wide audience. Each distribution list has a distinct purpose, e.g., lists of people interested in particular topics, lists of employees in certain organizations, lists of members of particular projects, etc. Some lists are used primarily to keep track of all users of the message system. These include such lists as AllPA^.PA, AllES^.ES, etc., which contain the names of all individuals in those particular registries. There are also some lists maintained on a purely geographical basis, e.g., PaloAlto^.PA, which lists all message system users in Palo Alto, California. This is not necessarily the same as AllPA^.pa, which includes people in the PA *registry*, but who may not actually work in Palo Alto.

The audiences addressed by these lists should not be considered a captive audience for all users of the message system. The purpose of any distribution list may be discovered by any user (in a registry served by Grapevine) by running the Maintain.laurel program and using the Type Entry command for that list (Appendix B). The purpose of the list will be printed in the Remark: entry for that list. Although all lists are (currently) available for use by any message system user, many lists, e.g., All*x*^.*x* where *x* is a registry name should not be used by anyone who doesn't have a very good reason for doing so.

Many distribution lists exist for the enjoyment of their members who wish to receive items of interest to them. One should feel free to send an anouncement of an upcoming musical event in Northern California, for instance, to Music^.PA. Such a message is quite inappropriate to send to AllPA^.PA, PaloAlto^.PA, etc. There are lists of message system users who have agreed to suffer through any and all messages. These lists (Junk^.PA, various CrankMail.dl files, etc.) are the only lists to which ridiculous messages may be sent without incurring the justifiable wrath of message system users.

A Laurel user should understand when a message is appropriate to send to all people in his or her work group. Social values are different in different locations, and the members of each group should understand what they are. It has been observed that messages that are sent to audiences wider than the sender's immediate group are the ones that cause the most trouble.

Unfortunately, unsolicited messages have continued to be sent to inappropriate lists. Examples of inappropriate messages for standard organizational or geographic lists are:

> "Does anyone know how to get my Alto fixed?"
>
> "This is to let everyone in the message system world know that my phone number has changed."
>
> "I want everyone to know that I really like my roofing contractor."

I'm sure that each user of the message system can recall some other similar gem. The following sections explore some of the consequences of unsolicited mail.

The chain reaction

To add insult to injury, after some piece of particularly ridiculous mail has been broadcast to an inappropriate audience, it invariably follows that some recipients cannot control their urge to make even bigger spectacles of themselves by sending their two cents to everyone who received the original nonsense. While the original event is thought by many message system users to be annoying, the latter is considered to be downright stupid. Remember that once you push the **Deliver** button and watch the last chance to cancel fade away from your screen, there is no way to erase your comments from the collective memory of your peers.

Further on, I will give details of the facilities available in Laurel to counteract this menace. For now, I would like to list some of the typical responses that have been sent not just to the original perpetrator, but to the entire list of victims.

> "Your message is inappropriate to send to all these good people."
>
> "If you don't like junk, then get off Junk^".
>
> "How do I get off Junk^?"

and, my favorite,

> "Do you realize that if all of us replied to all of us (as I am doing right now) that the number of messages that would be sent would exceed the number of atoms in the known universe . . . "

It is my opinion that bombarding only the original sender of a ridiculous message with equally nonsensical replies is poetic justice. There have been many requests for a **Fed up** command to be added to Laurel for just this purpose. Although I am sympathetic to such requests, for now we've just added them to the collected Laurel annals.

An answer to the question "How do I get off Junk^?" (in a registry served by Grapevine) is that the program Maintain.laurel (Appendix B) can be used to examine and modify public distribution lists. If you cannot modify the list yourself due to its access controls, then send a message to one of the people listed as an owner of that list.

The measures taken within Laurel to counteract the chain reaction phenomenon involve use of a special header field in messages called the Reply-To: field. Please note the spelling of "Reply-To": it contains a hyphen.

When an answer to a message containing a Reply-To: field is initiated with the **Answer** command, only the name(s) listed in that field (plus your own name in the copies field) are put into the answer form as recipients. In conjunction with the automatic addition of Reply-To: fields upon delivery, this gives a simple mechanism to break the chain of replies.

When a message is sent with the **Deliver** command, if that message contains a large number of recipients or any public distribution lists, and it has no Reply-To: field, then the delivery is interrupted pending user interaction to specify what kind of Reply-To: field is desired. A prompt will appear in the feedback region specifying the number of recipients and the number of distribution lists to which the message is being sent. It also asks you to choose a "Reply-To" option, with the reminder

"ESC = answers to self only, A = answers to all, DEL = cancel delivery."

At this point you must choose one of these options; delivery is postponed until you do so.

The recommended option when sending to a large list is for you to strike the ESC key. This will automatically insert a Reply-To: <self> field, where <self> is your name. Anyone who receives such a message and who initiates a response by using the **Answer** command will begin editing a form that includes only you and himself or herself as recipients.

There are situations in which replying to the entire list of original recipients is appropriate. These situations include sending technical messages to members of a project, scheduling queries for backgammon nights, etc. In these cases, strike an A (upper or lower case) for your "Reply-To" choice. This will send the message without a Reply-To: field, so that recipients who use **Answer** will get forms with all recipient names and lists included as recipients.

If you are hopelessly confused by this, or you realize that you would like to edit the Reply-To: field slightly, then strike the DEL key (actually any key other than ESC or A) in answer to the "Reply-To" prompt. The Reply-To: <self> line will be added to your message anyway, but the message will not be sent. At this point you may edit your message, perhaps adding a few extra names to the Reply-To: field, and then invoke the **Deliver** command again.

Note that you are not bothered by this prompt if you have already included a Reply-To: field in your message. The reasoning behind this is that if a Reply-To: field is already in the outgoing message, then you must have already noticed the wide distribution of the message and taken appropriate steps. Good for you!

One final note on this topic. Although Laurel provides these mechanisms to help break chain reactions, the ultimate responsibility for messages sent lies with their senders. Always check the list of recipients in any message you are about to send. Excuses of the form "Laurel let it get away." are feeble indeed.

Miscellaneous distribution list peccadillos

Here are several other tips to bear in mind when using distribution lists.

A private distribution list is reasonable only when you wish to control all messages to that list. If there is any reason to allow others to send to the list, then set up a public list. Instructions for doing so are found in section 3.7.

If you do set up a private distribution list, then do not include any public distribution list names in it. Others may modify that list, thus indirectly modifying your private list.

Names included in any distribution list should always be fully qualified, i.e., contain the .registry suffix. Only this way will the list be useable by others outside your own registry.

Off-the-record responses

There are many situations in which a user submits a question to a wide audience, say to a distribution list of people interested in such questions, and indicates that he or she will collect responses and later make them public. This is a most reasonable thing to do, and it helps to reduce the chain reaction effect. In Laurel 6, be sure to include a Reply-To: <self> field when performing such services for your audience.

A note of caution is in order here. Messages should be considered PRIVATE, unless otherwise indicated. If your intention is to publish the responses, then by all means make that intention clear in the same message that poses the original question. If your message did not make that intention clear, and you decide that you would like to publish the responses, then follow up on each response asking whether you may do so.

If the intention to publish responses is clearly indicated in the original message, then publication of any response is fine, as long as that response does not explicitly mention that it should be considered private.

Hardcopy forms

The message system in Xerox is used for communication about Xerox related business and for personal messages. It is appropriate to put onto Xerox internal memo forms (electronically generated or not) only those messages whose purpose is related to Xerox business. (The corporation has specific guidelines relating to the use of the Xerox logo, internal memo forms, etc. Common sense is all you need to derive these guidelines for yourself.) Many of the features and defaults of Laurel have been designed to allow users of the message system to behave properly with respect to these guidelines.

If a message you send to others is intended to be a Xerox internal memo, include a PrintForm: InternalMemo line in its header. When your recipients print this message in the normal way, the message will appear as an internal memo. On the other hand, if a message is truly frivolous, a PrintForm: Blank line in the header of your message is likely to prevent inappropriate printing. Refrain from mentioning any custom hardcopy form in a PrintFrom: field unless you know that all of your recipients have included that custom form in their Laurel profiles.

Although it is possible to change the default hardcopy form (used for those messages that do not contain a PrintForm: field), this change should not be made unless you have a real need to do so. The Blank form is adequate for most hardcopies, and special printing is possible using the **Form {**brackets**}** in the hardcopy submenu on a case-by-case basis.

Masquerading

On occasion, people have received messages from fictitious senders, or even worse, from someone masquerading as another real message system user. This is a most serious breach of message system etiquette, and should be considered so by all message system users.

A fictitious From: field is legitimate when a valid Sender: field is included. For instance, messages that are properly signed with an organization’s name, say "The Laurel Group", may be sent by explicitly typing a "From: The Laurel Group" line in the message header. Laurel will notice that a From: field is already there, and it will include a Sender: <User name> line in the delivered message instead of its usual From: <User name> line. Any time you receive a message that has a strange From: field, you may check the Sender: field for the actual sender.

By a "masquerader" I mean someone who subverts the normal mechanisms embedded in the standard message system programs to send messages of dubious value, without having his or her name appear in such messages. This action is possible not only in electronic message systems, but in other more traditional communication media as well. Masquerading as another may be a criminal act when committed using traditional communication media, with penalties specified in laws that prohibit libel, slander and fraud. Other situations, such as telephone "breathers", are similarly outlawed.

At this time, I do not know of any court cases involving libel, slander, etc. in an electronic mail context. Such cases are sure to arise when electronic mail does become more widespread. Masquerading in the message system is not cute or clever. Don’t do it.

Wizards vs. naive users

This section is addressed mainly to the wizards who should know better. The population of message system users covers a broad range from those who have knowledge of the most arcane details of a system to those who just barely understand the basics of using that system. When you send a message to a wide audience, be considerate of the naive users, who may get confused by technical jargon.

This admonition extends to those who are using a new, restricted program. It does not help a recipient to hear "Oh you’re using that old program. Well, I guess you’re stuck." Just don’t mention such things to users who cannot take advantage of them.

The moral of this tale

The moral of all this is simple: Be considerate. As we strive toward this goal, everyone’s use of the message system will become even more of a joy than it already is.

7. Look before you leap . . .

As you become familiar with Laurel, you will doubtless discover features that please you and "features" that annoy you. This section reports some of the annoying "features" other users have encountered and, in some cases, what to do to get around them.

Formatting messages for non-Laurel users

Laurel breaks lines in a transmitted message by a rather simple-minded algorithm. This may lead to line overflow when the message is read on certain terminals using MSG (or other mail systems). If you know that some of your recipients have this problem, you may wish to exercise care in the formatting of your outgoing messages. Use explicit CRs and keep your lines down to about 80% of the Laurel screen width. (A relatively painless way to do this is to compose the entire message *without* using CRs, then just before delivery make a final pass over the message, substituting CRs for spaces at appropriate points.)

Shifted (and secondary) selection from the message display region

Text in the message display region generally has real CRs in it. Laurel respects this formatting information, even when displayed text is copied into a composed message. When you insert text from a displayed message into the message composition region, you may discover that the real CRs break lines in the wrong places. If you don’t like this, you will have to replace those explicit CRs by spaces after copying the text. The **Substitute** command (S (COM-S), section 4.4.10) provides a convenient method for doing this if a large number of CR’s are involved.

The Answer form

The form provided by the **Answer** command is not always exactly what you might want. Laurel takes the viewpoint that it is easier to delete than to add text, and therefore includes all the information that seems to be relevant. For example, Laurel often includes your name in the cc: field of the answer form, even though you may not want a copy of the message you are composing. You should *not* expect that the answer form will be exactly right; always examine the header to be certain it contains the desired information.

In particular, if you reply to a message that was directed to a distribution list, the **Answer** command will copy the distribution list name into the cc: field of the answer form. You should consider carefully whether or not it is appropriate for your reply to be sent to that distribution list, and if not delete the distribution list name. (See also the discussion of Reply-To: fields in section 6).

Space requirements

Many Laurel users find it convenient to designate a disk on which they will use Laurel exclusively. You will need about 700 disk pages to hold Laurel and a moderate-sized mail file.

Some users, when converting from MSG to Laurel, have had a rude shock when they transferred their archival mail files from Maxc to their Alto disk. Recall that one Maxc disk page becomes four Alto disk pages when a file is moved. This is another reason for dedicating an Alto disk to mail processing and filing, particularly if you maintain archival mail files.

Hardcopy usage

Laurel's **Hardcopy** command is intended for use in printing copies of isolated messages that you may wish, for whatever reason, to obtain in hardcopy form. It is only suitable for obtaining hardcopy archives of all your messages, say, to put in your filing cabinet, if you print using the "Archive" hardcopy form (section 3.3.2). The other standard hardcopy forms, that print each message on a separate page, are definitely unsuitable for hardcopy archives.

If you do wish to maintain a hardcopy archive, and the "Archive" hardcopy form is not to your taste or itself produces too much paper, we recommend that you use Empress.run (or other printing programs) to print your mail file directly as continuous, unformatted text.

Type-ahead and button-ahead

Laurel directly senses the COM key (modeless editor only) and the SHIFT key for the SHIFT-BS and SHIFT-NEXT combinations (also modeless editor only) at the time that the command is performed. Thus type-ahead for these combinations is not supported. For normal text type-in, type-ahead is supported.

Mouse button-ahead is not supported at all.

Type-ahead is flushed on entry and exit from the **Run** command. Have patience at these times.

8. If things go wrong . . .

In Laurel, as in most interactive systems, lots of things can go wrong. Also as in most systems, some of them are your fault and some of them are not. Laurel tries to prevent you from wreaking destruction upon your environment, and reports any (perceived) violations in the feedback region.

The error reports in the feedback region are intended to be self-explanatory. If you cannot figure out what one means or what to do next, please send a message to LaurelSupport.PA (using Laurel if possible).

If Laurel detects certain internal error conditions, it interrupts whatever command was in progress and posts a message in the feedback region. In some cases, Laurel may decide that the error should be reported to LaurelSupport.PA, and, in such cases, it will prepare an error report form (after confirmation from you). This form contains internal status information of interest to the Laurel implementers and should be used whenever possible. If you confirm the use of this form, Laurel will restart itself and display the form in the composition region. You should then follow the instructions contained within the form, after which Laurel will again be available for normal use.

The vast majority of Laurel internal errors reported in the last year have turned out to be due to problems with the disk or with the Alto (Dolphin, Dorado) processor. If Laurel "freezes" or otherwise behaves peculiarly for no apparent reason, a simple check of your disk will probably fix the problem. Run the Scavenger.run program from the Alto Executive. If it reports a "Read error", then have a hardware maintainer check out your disks thoroughly. If Laurel continues to behave strangely, e.g., it doesn't run at all, suspect the processor. If you have sent in an error report through the built-in error reporting mechanism, and you later discover that the problem really was due to faulty hardware, then please send a message to LaurelSupport.PA to explain what happened.

"Freezing" is defined as Laurel becoming completely unresponsive to mouse movement and keyboard input. If the cursor moves when the mouse is moved, then Laurel has not "frozen"; be patient and wait a bit longer for the current operation to complete. Only if an operation takes a very unreasonable amount of time should you suspect that Laurel is in a bad state. Use your good judgment; a long file transfer or a **Quit** or **Mail file** command given when the current mail file has over a hundred messages may take a while. If your cursor does not move when you move the mouse, but keyboard input is accepted by Laurel, then check to see whether your mouse is properly connected. (We get them all.)

Regardless of the above discussion, when Laurel enters automatic bug reporting mode, a real Laurel bug may be the culprit. Please do fill out the description part, and deliver the error report message. Only because of the diligent attention paid to these error reports by users in the past has Laurel has become as error free as it is.

9. Things a casual user doesn’t really need to know

For each mail file *x*, Laurel creates a separate file *x*-dmsTOC, which holds various internal information about the table-of-contents. When you invoke **Mail file** {*x*}, Laurel will recreate this file if necessary. Naturally, this slows down the **Mail file** command.

Laurel creates scratch files while it is working and leaves them on your Alto disk. You may delete these files (named DMS-*n*.TMP for various values of *n*) if necessary, though Laurel ensures that they will not grow very large. If you terminate a Laurel session abnormally (or Laurel crashes) then these files might actually be quite large. Deleting them lengthens the time required to start up Laurel.

If Laurel generates an automatic error report directed to LaurelSupport.PA (see section 4), it will leave behind a file Laurel.BugReport$ containing the text of that report. You may delete this file if you wish.

SHIFT-SWAT is disabled in Laurel soon after start-up, since SWAT is one of the COM keys. You may achieve a SHIFT-SWAT by holding down the combination SHIFT-NEXT-COM. Performing such an action is of dubious value.

Appendix A: Command line options

Laurel has several command line options that can be specified when you invoke it from the Alto Executive.

Mail file selection

Laurel's default action, triggered when you type

>Laurel CR

is to perform an implicit **Mail file** on Active.mail. If you type

>Laurel *filename* CR

Laurel will read *filename*.mail instead. If you supply an explicit extension, Laurel will respect it, otherwise it will add the .mail suffix.

In-box interrogation

You can also request that Laurel access your in-box automatically when it begins execution. Type

>Laurel/n CR   or
>Laurel/n *filename* CR

Laurel will first perform an implicit **Mail file** on the indicated or defaulted file (see above). It then will check to see if messages are present in your in-box. If so, Laurel performs an implicit **New mail** command. Upon the completion of the **New mail** command (if performed), Laurel is available for normal use.

You may also invoke Laurel by

>Laurel/c CR   or
>Laurel/c *filename* CR

If messages are present in your in-box, Laurel will behave as though "/n" were specified. If no new mail exists, Laurel will omit the implicit **Mail file** and return directly to the Alto Executive.
A fine point: if Laurel is unable to contact your in-box(es), it returns to the Alto Executive. Laurel will not return to the Alto Executive, however, if there is no password on your disk. In this case it will allow you to log in with the **User** command and continue to run Laurel.

Send message mode

If you merely want to send a message, you may request that Laurel omit its implicit **Mail file** when it begins execution. Type

>Laurel/s CR

Instead of reading a mail file, Laurel will perform an implicit **New form** as it starts up. You may then compose and deliver your message in the usual way. If, after doing so, you wish to process a mail file, simply invoke **Mail file** and supply a file name as described in section 3.1.3.

Appendix B. Programs available for the Run command

The programs listed in this appendix are available for use with the **Run** command (section 3.6). These programs are supported by the Laurel group and are available on the <Laurel> directory on your neighborhood file server.

You may wish to keep these programs on your local disk, or you may retrieve them "on the fly" using the **Run** command. When programs have been retrieved in the latter manner, they are copied to your disk and are left there even after you are finished with them. You may wish to include a RunPath: entry in your Laurel profile to make this process a little simpler (section 5).

Many programs that run via the **Run** command use the composition region of Laurel as a typescript. This typescript is limited to 60,000 characters, i.e., only the last 60,000 characters are retained for any session. When you are finished with such a program, this typescript remains in the composition region, where you may edit it, file it, deliver it, etc. Normally, this typescript replaces the previous contents of the composition region. It may be restored after the **Run** command is finished by invoking U (CANCEL). If you begin another **Run** command immediately after finishing a **Run** command (no editing in between the two **Run**'s), then the typescript (if any) for the subsequent program is appended to the previous typescript (if any).

Many of the programs listed here allow you to invoke other Laurel screen commands. The commands that are typically enabled are **Copy** in the lower menu and all commands in the upper and middle menus except for **Quit** and **Hardcopy**.

Shifted selection is always available as an alternative to type-in. This makes acting on lengthy requests contained in a received message considerably easier.

In the modeless editor, the PASTE key reverts back to a LF key when the **Run** command is executing. It is restored to a PASTE key when the **Run** command finishes.

B.1. Chat

Chat.laurel provides a teletype interface to servers that support the Telnet protocol. Chat.laurel is similar to Chat.run, etc. in its effect, but it does not support the variety of commands available in the other Chat programs. Chat.laurel interacts with you via a typescript in the composition region of the Laurel screen.

Starting and stopping Chat

When you start Chat (by invoking the **Run** command with program name Chat), it greets you with the lines:

> Laurel Chat of *date*
> (CTRL DEL closes connection and returns to Chat command level.)
> C(onnect to), L(ogin to), or Q(uit)?

A blinking caret appears after the question mark, inviting you to begin your Chat session. At this point you are at *Chat command level*. Only the three characters C, L, or Q are valid inputs at this level (upper or lower case does not matter).

A "Q" typed at command level will terminate both the Chat program and the **Run** command.

A "C" typed at command level will produce the prompt:

> Connect to host:

with a blinking caret following the colon, inviting you to type the name of a server machine (terminated with a CR) to which you wish to connect. If you type a DEL character before the CR, then you will return to Chat command level. Once you have connected to the server machine, all interactions are as defined by that server.

An "L" typed at command level will produce the prompt:

> Login to host:

with a blinking caret following the colon, inviting you to type the name of a server machine (terminated with a CR) to which you wish to connect. This command behaves identically to the "C" command, except that "L" also issues an automatic Login command to the server after the connection is established. The name and password used for this Login command are the most recent ones entered via the **User** command (section 3.1.1), or the ones on your local disk if you haven’t used the **User** command yet in this Laurel session.

Note: characters are only placed on the screen when they are echoed by the server to which you are connected. Most servers will not echo your password. Also, if the server is heavily loaded, there may be a slight delay between the time that you type characters and the time that they are displayed.

When you are connected to a remote server, if you type CTRL-DEL, then your connection is closed, and you return to Chat command level. If your connection is closed explicitly, e.g., by issuing a "Logout" command to the server, then you will also return to Chat command level.
Maxc often takes a significant amount of time to close your connection after you have logged out. Strike CTRL-DEL to notify Chat that you don't wish to wait.

Whenever you return to Chat command level, the one line prompt:

C(onnect to), L(ogin to), or Q(uit)?

is displayed. From this point you may make another connection or quit as described above.

Chat usage tips

The Chat program significantly extends the range of operations you may perform while still inside of Laurel. Some of these are listed here.

Most servers support remote printing in a natural way. IFS servers, for example, support a "Press" command that prints files stored on that IFS on a printer you designate without your having to retrieve those files to your local disk. If you receive a message that mentions the name of a press file stored remotely, using Chat plus shifted selection to type the file name to the "Press" command, is the most convenient way to have that file printed.

Command files of input directed to a remote server are possible using shifted selection from the message display region. Once you SHIFT-select text, all of it will be input to Chat as if you typed it yourself at the appropriate times. Be careful to ensure that you include all CR's, confirmations, abbreviations. etc. just as the server would expect them as if you had typed the text yourself. To learn how to get arbitrary text into your mail file and thus into your message display region, see section B.3, InsertMail.laurel.

B.2. Maintain

Maintain.laurel is useful for interacting with Grapevine servers for various purposes including changing your password, updating public distribution lists, etc. The Maintain program is only useful to you if you are in a registry served by Grapevine (currently only PA and ES, most others will be soon).

This section is an abbreviated guide to the use of Maintain. It contains descriptions of the commands needed by the casual user, i.e., a person who is not an owner of a list or a registry. A more complete explanation of Maintain will appear in the near future. A thorough technical description (not intended for casual users) of the facilities provided by Grapevine is contained in the file [Ivy]<DMS>Interface.press.

Using Maintain

When you start Maintain (by invoking the **Run** command with program name Maintain), it greets you with the lines:

> Grapevine Registration Server Maintenance Program
> Version of *date*
> Login *Name.registry* ...
>
> GV:

A blinking caret appears after the GV:, which is the command prompt, inviting you to type a Maintain command. If you type a question mark, a complete list of commands will be typed out, and the GV: prompt will be issued again. Most of these commands are quite technical; we will deal with only a small subset of them here.

To issue a command to Maintain, you must type only the initial letters that uniquely distinguish that command from the others. As soon as enough of a command word (usually only one letter) has been typed, Maintain will complete that word for you. If you type any subsequent letters in that word beyond the unique prefix, those letters will be used for the rest of the command, which almost always results in an error. In the list of commands that follow, the complete command name will be given first, with the unique letters you should type to issue that command following in parentheses.

Most commands require names as arguments. Maintain will generally suggest a name for each argument based on the names you have entered to previous commands. If the name is correct, just confirm it by typing a space or CR. Otherwise, type in the name (shifted selection works) and terminate with a space or CR (may be included in the shifted selection). All names given to Maintain must be fully qualified, i.e., they must include the registry extension.

If you type a DEL character at any time during command input, that command will be cancelled and a new GV: prompt will be given.

Maintain commands

The following commands may be given in any order, with the exception that you must be properly logged in to give any command other than ?, Login, or Quit. When Maintain is started, it tries to log you in automatically with the current **User** name and password. If that fails, then you may log in using the Login command in Maintain.

The commands listed here are in alphabetical order.

Add Member (AME)

First type the name of a person, including the *.registry* suffix, to be added to a public distribution list. Maintain will respond with *to group*. Now type the name of the list (remember to include the registry). If you are allowed to add that person to that list (there are several controls on this), then the requested change will be made. If not, Maintain will flash the screen and type out the reason for the rejection. In general, you are allowed to add yourself to general interest lists, but not allowed to add anyone else.

Login (L)

Maintain will prompt with *Your name please:* . Type your name, including the *.registry* suffix. Maintain will then prompt with *Your password:* . Type your password. Maintain will echo a * for each letter in your password. Grapevine maintains its own copy of your password, and the password you give here must be the one the Grapevine knows. If you give your password successfully, Maintain will respond with *ok*. Once you are logged in, you may proceed to issue other Maintain commands. If you cannot log in, because you have forgotten your password or for some other reason, then you will have to ask an administrator for your registry to issue a Set Password command for you (see below).

Quit (Q)

This command must be confirmed with a CR or the letter Y. When confirmed, this will terminate Maintain and the **Run** command.

Remove Member (RME)

First type the name of a person to be removed from a public distribution list. Maintain will respond with *from group*. Now type the name of the list (remember to include the registry). If you are allowed to remove that person from that list (there are several controls on this), then the requested change will be made. If not, Maintain will flash the screen and type out the reason for the rejection. In general, you are allowed to remove yourself from general interest lists, but not allowed to remove anyone else.

Set Password (SP)

Maintain will prompt with *to be*. Type your new password. Maintain will echo a * for each letter in this password. After terminating the password with a space or CR, Maintain will prompt with *for individual*. Type your name, including the *.registry* suffix. When you terminate this name with a CR or space, Maintain will store this new password in the Grapevine name data base. From now on, this is the password you must use to access your new mail or to send mail. Be very careful when typing your password. If you make a mistake you may type DEL to cancel this command. Once you type the terminating CR or space for the entire Set Password command you are committed to whatever password you have typed. If you can't reproduce this password later, you will have to obtain the help of an administrator for your registry to correct it.

After setting your password with the Set Password command, you will need to re-invoke the Login command with your new password if you wish to continue using Maintain. Also, when you are back in Laurel, you should re-invoke the **User** command to allow access to your new mail.

Type Entry (TE)

Maintain will prompt with *for R-Name*. Type the name of the individual or public distribution list for which you wish to see information. For individuals, you can discover the current in-box server(s) and forwarding established for that individual. For lists, you will see the purpose of the list (in the Remark: field), the owners of the list (in the Owners: field), and the people who are allowed to add or remove themselves from that list (in the Friends: field).

Type Members (TM)

Maintain will prompt with *of group:*. Type the name of the public distribution list whose members you wish to see. Maintain will list the members of this group, some of which may be lists themselves. To see the members of those lists, extra Type Members commands may be used. Type Members of the special group "Groups.*registry*" produces a listing of all the distribution lists available in *registry*. The Type Members command can only access public distribution lists held in the Grapevine name data base. The **Get** command (section 3.5.1) can access any distribution list to which you can deliver a message.

B.3. InsertMail.laurel

InsertMail is a program that inserts the contents of the message composition region as a message into your current mail file. The message is inserted as is; unlike the **Deliver** command, no From: or Date: fields are added in this process. If you wish to include such fields in the inserted mail, you must edit them in yourself prior to running InsertMail. The T (COM-T) command is useful in preparing a Date: field, should you wish to do so.

InsertMail inserts the message *after* the last selected entry in the table-of-contents. If no messages are selected in the table-of-contents, then the message is inserted at the beginning of the mail file, i.e., it becomes message number 1.

When the message is inserted, the table-of-contents is reformatted, and the newly inserted message is selected and displayed. If the message as inserted does not conform to legal message header rules, or if some header fields are missing, then the table-of-contents display for that entry may have some parts missing or filled in with question marks.

InsertMail does not use the composition region for a typescript. It does its job, terminates, and leaves the text in the composition region as it was.

Uses for InsertMail

Have you ever wanted to annotate or otherwise edit a message in your mail file? The simple way is to move that message into the composition region (**Display** the message, use E (COM-E), and replace the contents of the composition region with a shifted selection of the entire message display region--use multi-clicks to select it.) Then, edit the message and **Run** InsertMail. If you haven’t moved the table-of-contents entry, then the new version will be inserted immediately after the original version. If you wish, you may select the original version and invoke **Delete**.

To prepare a command file from which a shifted selection will be taken as input to some other runnable program, just prepare that command file in the composition region and InsertMail it into your mail file. You may wish to prefix the command file message with a portion of a message header, say the Date:, From:, and Subject: fields, to have that command file appear reasonably in the table-of-contents.

To print text that you are preparing in the composition region, InsertMail it into your mail file and invoke **Hardcopy**.

B.4. Files

Files.laurel is a program that provides many of the functions on files that are provided by the Alto Executive.

When you start Files (by invoking the **Run** command with program name Files), it greets you with the lines:

> Laurel File Utility of *date*
> Type ? for help.
> \>

A blinking caret appears after the greater-than sign, inviting you to issue a Files command. The ">" characters is the prompt character, and always indicates that Files is ready for another command.

When giving a command to Files, any unique prefix of that command may be typed (as in the Alto Executive). ESC command completion is not supported; typing an ESC will insert that (illegal) character in the command. Terminate command lines with a CR.

Typing CTRL-DEL will cancel commands in progress. This is particularly useful for interrupting long typeout from the List or Type commands.

Most commands allow a file pattern wherever a single filename would be acceptable. A file pattern is expressed using the # and * characters, meaning match exactly one and match zero or more characters respectively, as in the Alto Executive.

Files commands

Copy *NewFile _ FilePattern FilePattern . . .*

> The Copy command is similar to the Copy command in the Alto Executive. The contents of the old files to the right of the arrow will be concatenated and copied into the new file mentioned to the left of the arrow.

Delete *FilePattern FilePattern . . .*

> The Delete command is similar to the Delete command in the Alto Executive. The Delete command requests confirmation for each file before it actually deletes that file. A CTRL-DEL to cancel the entire Delete command will not be acted on until after you finish confirming or cancelling the current file.

FileStat *FilePattern FilePattern . . .*

> The FileStat command is similar to the FileStat command in the Alto Executive. The length and create, read, and write dates for each file specified are listed.

List *FilePattern FilePattern* . . .

The List command is used to list all files on the local disk that match the patterns specified. In the Alto Executive, this function is provided by the TAB character. TAB used in this way is not supported in Files.laurel; use the List command instead. The matching files are listed in the order in which they occur in your directory. Although the output may look alphabetized, that only reflects the periodic sorting done in the Alto directory by the Alto Executive. You may find files out of alphabetical order near the end of your List command output.

Rename *OldFile NewFile*

The Rename command is similar to the Rename command in the Alto Executive.

Quit

The Quit command terminates the Files program and the **Run** command.

Type *FilePattern FilePattern* . . .

The Type command types the contents of the specified files one after the other, without pausing. Each file is preceded by a short length description. You may terminate its output by typing CTRL-DEL.

B.5. SearchMail

The SearchMail.laurel program will search through your entire current mail file, looking for text that matches a pattern. The message number of any message in which a match is found will be displayed in the typescript, and that message is selected in the table-of-contents. When SearchMail is finished, it terminates the **Run** command, leaving in its typescript the message numbers of all messages that contain a match, and with all such messages selected.

The patterns that are allowed are the same as those allowed by the **Find** command (section 4.4.5). Only one pattern is allowed per run of SearchMail; each subsequent run of SearchMail begins a new table-of-contents selection. You must terminate the pattern with an ESC character. Once the pattern is terminated, the search begins.

After SearchMail is finished, you may operate on the set of matched messages in a variety of ways. You may invoke any of the commands in the middle menu to delete, move, or hardcopy the entire set. If the set of selected messages contains two or more messages, then the **Display** command (section 3.2.1) will display each selected message in turn, without destroying the table-of-contents selection.

SearchMail thus provides a rudimentary keyword search capability to Laurel. It is limited to searching for only one pattern (no combinations of patterns are allowed), and it will search within the current mail file only.

B.6. MailFileScavenger

In the unlikely event that your mail file becomes damaged, there is a remedy in the form of the MailFileScavenger.laurel program that usually restores the internal structure of your mail file to a reasonable state. It cannot deal with disk errors; use Scavenger.run first if you suspect disk errors. The MailFileScavenger copies the damaged mail file into a new scratch file as it operates, therefore you must have slightly more free disk pages available for this scratch file than the number of disk pages that your damaged mail file occupies. The MailFileScavenger will warn you if there is not enough room.

To run the MailFileScavenger, just invoke Run with the program name MailFileScavenger. The MailFileScavenger will ask you to type the name of the mail file to be scavenged. Terminate this name with a CR. If you type a name without a period, .mail will be added to the name automatically. MailFileScavenger will proceed to copy your mail into its scratch file (named MailFileScavenger.scratch$). After each fifth message, MailFileScavenger will print out that message number, just to let you know it is still working.

When anomalies are detected in your mail file, MailFileScavenger will print out a short message such as "Message 53: existing count was 231 bytes too small." These messages indicate that the formatting information present in the mail file that Laurel uses to distinguish individual messages was inconsistent with what the MailFileScavenger believes to be distinct messages. When the MailFileScavenger is finished, it is a good idea to check any messages it complained about; these messages may be missing several characters or be malformed in other interesting ways. You should also check neighboring messages--some of the characters in those messages might really be part of other messages.

After the MailFileScavenger has finished copying (and reformatting) your mail into its scratch file, it will pause and ask if it should copy that file back into the original mail file (Type CR or Y to confirm). If there have been few error reports, this may be done without any trepidation; the MailFileScavenger will do the copy, delete the scratch file, and terminate. You may at this point invoke **Mail file** on your mail file once again and gaze upon the damage. On the other hand, if there have been many error reports, you may wish to type a DEL character to deny the automatic copy, and examine the MailFileScavenger.scratch$ mail file before performing the copy yourself. You may invoke **Mail file** on the MailFileScavenger.scratch$ file merely by typing (or SHIFT-selecting) that name into the **Mail file** brackets.

The mail file that MailFileScavenger produces should always give you a readable mail file, i.e., one that Laurel will not complain about. This mail file may have "messages" that are fragments of messages in the original file and/or duplicate messages. As long as Laurel will allow you to read that file, you may repair the damage with the Laurel editor and the InsertMail program (Appendix B.3). If the file produced by the MailFileScavenger is not readable by Laurel, please send a message describing your difficulties to LaurelSupport.PA.

Appendix C. Hardcopy Forms

Laurel allows you to create your own custom hardcopy forms by including a description of each form in your Laurel profile. The language used to specify these forms is described in this appendix. The forms that are built into Laurel are listed at the end of this appendix, written in this forms language.

The syntax for a hardcopy form definition in your Laurel profile is

HardcopyForm: <*form*>

where <*form*> is described in more detail below. Each <*form*> contains the name of the defined hardcopy form. If a hardcopy form included in your profile has the same name as a built-in hardcopy form, then the form in your profile replaces that built-in form. There is no intrinsic limit on the number of forms you may put in your Laurel profile.

In this description, words in angle brackets refer to items to be described in further detail. All parentheses shown are included in the actual hardcopy form. Items separated by whitespace may be separated in the hardcopy form by one or more whitespace characters or by a CR followed by at least one whitespace character. Never put two CR's together in the hardcopy form. It will terminate the parse.

When creating your own hardcopy forms, it is usually best to modify an existing form, rather than building it from scratch. The profile error message for most errors in hardcopy forms is just *Profile syntax error near <text>*.

C.1. <Form>

The printed page is divided into two areas: the text area and the frame area. The text area is the rectangular space inside the four margins of the page in which message text is printed. The frame area is the space left over, from the margin lines to the edges of the page.

A <*form*> specifies the name of the form, the four margins, some leading (vertical spacing) information, and a list of <*options*> and <*rows*> of the form.

An <*option*> specifies an item that is generally printed in the frame area, e.g., a page number. Each <*option*> is printed on either the first page only or on all pages except the first page.

A <*row*> of the form extends across the text area horizontally and as far vertically as the text printed within requires. The <*rows*> are printed one after the other, in the same order as given in the <*form*>. Each <*row*> contains one or more <*column*>'s. <*Column*>'s are printed side-by-side within a <*row*>. The height of a <*row*> is the height of its longest <*column*>.

The types of values included in hardcopy forms are: text, mica, font number, and Boolean.

A text value is a string of characters bounded by double quote characters. If the string contains at least one character and it does not contain any whitespace characters, then the double quote characters may be omitted. To include a double quote character in a text value, use two double quote characters.

A mica value is a positive integer giving the number of micas in the X (horizontal) or Y (vertical) direction from the lower left corner of the printed page, as per in Press file page coordinates. One inch equals 2540 micas.

A font number is a positive integer between 0 and 9 inclusive that refers to a hardcopy font, either built-in or listed in the Laurel profile (section 5). The built-in forms use font numbers 0 through 3, and the standard HyType form uses font 9. These font numbers should be considered "reserved" when creating a new hardcopy form.

A Boolean is a logical value represented by the words TRUE or FALSE. The abbreviations T and F are also accepted.

The syntax for a <*form*> is:

> <*FormName*> <*TopY*> <*BottomY*> <*LeftX*> <*RightX*> <*LineLeading*> <*StartNewPage*>
> (Options <*Options*>) (Rows <*Rows*>)

The individual fields of the <*form*> are:

<*FormName*>: A text value that names the form.

<*TopY*>: The top margin in micas, measured from the bottom of the page. The first <*row*> will be printed at this location as long as it has no vertical tab.

<*BottomY*>: The bottom margin in micas, measured from the bottom of the page. Any text within a <*row*> that would be printed below this point will be printed on the next page instead.

<*LeftX*>: The left margin in micas, measured from the left edge of the page. This value is overridden by <*leftX*> values in <*row*>'s and <*option*>'s.

<*RightX*>: The right margin in micas, measured from the left edge of the page. This value is overridden by <*leftX*> values in <*row*>'s and <*option*>'s.

<*LineLeading*>: No longer used. Preserved for compatibility with extant forms.

<*StartNewPage*>: A Boolean value that controls whether each message will start on a new page. The built-in InternalMemo and Blank forms set this value to TRUE; the Archive and Headers forms set it to FALSE.

<*Options*>: The <*options*> consist of one or more <*option*>'s separated by spaces, i.e., <*option*>$^+$. The (Options <*options*>) part of the form, complete with surrounding parentheses and keyword, may be omitted if no options are desired.

<*Rows*>: The <*rows*> consist of one or more <*row*> separated by spaces, i.e., <*row*>$^+$. The (Rows <*rows*>) part of the form, complete with surrounding parentheses and keyword, may be omitted if no rows are desired.

See the last section of this appendix for examples of <*form*>'s.

C.2. <Option>

An <*option*> may be a caption, heading, or page number. Each is considered in turn.

C.2.1. Caption option

A caption option prints a fixed piece of text at a fixed location on the page. The syntax for a caption option is:

(Caption <*LeftX*> <*BaseLineY*> <*Font*> <*OnFirstPage*> <*Text*>)

The parameters are:

<*LeftX*>: The horizontal position, measured from the left edge of the page in micas, for the left edge of the caption.

<*BaseLineY*>: The vertical position, measured from the bottom of the page in micas, for the baseline of the letters in the caption.

<*Font*>: The number of the font in which the caption is printed.

<*OnFirstPage*>: If TRUE, then this caption is printed on the first page only. If FALSE, then this caption will be printed on all pages except the first page. To print the same caption on all pages, include two caption options, one with this value TRUE and the other with this value FALSE.

<*Text*>: The text of the caption. If the text includes white space, surround the text with double quote characters.

For example, the following caption option taken from the built-in InternalMemo form prints the words "Laurel Message" at the top of the first page of each message printed.

```
( Caption 4445 26670 2 T "Laurel Message" )
```

C.2.2. Heading option

A heading option prints the initial portion of a specified message header field's value at a fixed location on the page. The syntax for a heading option is:

(Heading *<LeftX>* *<BaseLineY>* *<Font>* *<OnFirstPage>* *<FieldName>* *<RightX>*)

The *<LeftX>*, *<BaseLineY>*, *<Font>*, and *<OnFirstPage>* parameters are as in the caption option. The other two parameters are:

<FieldName>: The name of the message header field whose value is printed as the heading.

<RightX>: The horizontal position, measured from the left edge of the page in micas, for the right edge limit of the heading. The heading text may be truncated if it is too long to be printed before this limit.

For example, the following heading option taken from the built-in InternalMemo form prints the value of the Subject: field at the top of the second and subsequent pages of each message, without running into the page number.

(Heading 3175 26670 0 F Subject 15875)

C.2.3. Page number option

A page number option prints the page number at a fixed location on the printed page. The syntax for a page number option is:

(PageNumber *<LeftX>* *<BaseLineY>* *<Font>* *<OnFirstPage>*)

All parameters are as in the caption option.

For example, the following page number option taken from the built-in InternalMemo form prints page numbers at the upper right of the second and subsequent pages of each message printed.

(PageNumber 18415 26670 0 F)

C.3. <Row>

The syntax for a *<row>* is:

(*<RowLeading>* *<VerticalTab>* *<LineLeading>* *<Columns>*)

The individual fields of the *<row>* are:

<RowLeading>: The spacing in micas between the bottom of the previous row and this row.

<VerticalTab>: The top line of the row is printed at or below this vertical position, measured in micas from the bottom of the page. If the previous row has finished below this position or if this value is zero, then it is ignored.

<LineLeading>: The vertical spacing in micas between printed lines in this row in addition to that provided by the hardcopy fonts themselves.

<Columns>: The *<columns>* consist of one or more *<column>*'s separated by spaces, i.e., *<column>*$^+$.

For example, the following row taken from the built-in InternalMemo form prints the message body with no additional row leading, no vertical tab, and 35 micas additional leading between lines of the message body.

```
( 0 0 35 ( Body 3175 19050 0 ) )
```

C.4. <Column>

A *<column>* may be a caption, field, other fields, body, or everything. Each is considered in turn.

C.4.1. Caption column

A caption column prints a fixed piece of text within a row. The syntax for a caption column is:

(Caption *<LeftX>* *<RightX>* *<Font>* *<Text>*)

The parameters are:

<LeftX>: The horizontal position, measured from the left edge of the page in micas, for the left edge of the caption.

<RightX>: The horizontal position, measured from the left edge of the page in micas, for the right edge limit of the caption. The text of the caption may be truncated if it is too long to be printed before this limit.

<Font>: The number of the font in which the caption is printed.

<Text>: The text of the caption. If the text includes white space, surround the text with double quote characters.

For example, the following caption column taken from the built-in InternalMemo form prints the word XEROX in font 3 near the left edge of the page after most of the header fields have been printed.

```
( Caption 635 19050 3 XEROX )
```

C.4.2. Field column

A field column controls the printing of a message header field name and its value. There are many parameters used to control this printing. The syntax for a field column is:

> (Field <*LeftX*> <*RightX*> <*ValueFont*> <*NameFont*> <*NameAboveValue*> <*Colon*> <*ValueLeftX*> <*FieldName*> <*FieldNameAlternate*> <*Required*> <*Suppress*> <*PrintFieldName*> <*BreakOnComma*>)

The individual parameters of the field column are:

<*LeftX*>: The horizontal position, measured from the left edge of the page in micas, for the left edge of the field name.

<*RightX*>: The horizontal position, measured from the left edge of the page in micas, for the right edge of the space reserved for this field. If there is more text in the field value than will fit, then the field value is continued on the next line(s).

<*ValueFont*>: The number of the font in which the field value is printed.

<*NameFont*>: The number of the font in which the field name is printed.

<*NameAboveValue*>: A Boolean value. If TRUE, then the field value is printed starting on the line below the field name. If FALSE, then the field value is printed on the same line as the field name.

<*Colon*>: A Boolean value. If TRUE, then a colon (in the font for the field name) is printed after the field name.

<*ValueLeftX*>: The horizontal position, measured from the left edge of the page in micas, for the left edge of the field value. If the printed field name encroaches on the space reserved for the field value, then the field value begins one space after the field name (if <*NameAboveValue*> is FALSE). Subsequent lines of the field value start at the <*ValueLeftX*> position.

<*FieldName*>: The name of the field to be printed.

<*FieldNameAlternate*>: An alternate name for this field. The field values from both the <*FieldName*> and the <*FieldNameAlternate*> fields are printed in the space reserved for the field value. This is useful for fields that go by two names, e.g., c and cc. The <*FieldName*> is the one that is printed in the field name area. Use "" to specify no alternate field name.

<*Required*>: A Boolean value. If TRUE, then the field name is printed even if this field is not in the message. If FALSE, then this field is omitted from the hardcopy if it (and its alternate) is not present in the message.

<*Suppress*>: A Boolean value. If TRUE, then the field (name and value) is not printed even if this field is present in the message. If FALSE, then this field is printed if the <*FieldName*> or the <*FieldNameAlternate*> is present in the message.

<*PrintFieldName*>: A Boolean value. If TRUE, then the field name is printed along with the field value. If FALSE, then the field name is suppressed, but the field value is printed. This printing is subject to the field's presence in the message and the other parameters for this field.

<*BreakOnComma*>: A Boolean value. If TRUE, then when printing the field value, commas are not printed but instead cause line breaks. If FALSE, then the field body is printed normally. This parameter is useful for cc: fields.

For example, the following field column taken from the built-in InternalMemo form prints the required field "From" with a colon and with its value following on the same line.

```
( Field 3175 10795 0 1 F T 4445 From "" T F T F )
```

C.4.3. OtherFields column

An OtherFields column controls the printing of all message header fields in the message that are not specifically mentioned in field columns elsewhere in the form. Generally, only one OtherFields column should be used in any form. The syntax for an OtherFields column is:

(OtherFields <*LeftX*> <*RightX*> <*ValueFont*> <*NameFont*> <*NameAboveValue*> <*Colon*>
<*ValueLeftX*> <*InterFieldLeading*>)

All parameters in an OtherFields column have the same meanings as in the field column, except that they apply to all fields of the message that are not dealt with by field columns. The one new parameter is:

<*InterFieldLeading*>: The vertical spacing in micas between successive fields not mentioned elsewhere in the form. This value is independent from the <*LineLeading*> value taken from the row in which this column is embedded.

For example, the following OtherFields column taken from the built-in InternalMemo form prints all fields found in the message header other than those specifically mentioned in the InternalMemo form with colons and with their values following on the same line. The spacing between these fields is 500 micas (about .2 inches).

```
( OtherFields 3175 19050 0 1 F T 4445 500 )
```

C.4.4. Body column

A body column prints the message body, i.e., the portion of the message following the first double CR in the message. Generally, only one body column should be used in any form. The syntax for a body column is:

(Body <*LeftX*> <*RightX*> <*Font*>)

The individual fields of the body column are:

<*LeftX*>: The horizontal position, measured from the left edge of the page in micas, for the left edge of the body.

<*RightX*>: The horizontal position, measured from the left edge of the page in micas, for the right edge of the space reserved for the body. If there is more text in the body than will fit, then the field value is be continued on successive lines.

<*Font*>: The number of the font in which the body is printed.

For example, the following body column taken from the built-in InternalMemo form prints the message body in the text area of the page in font 0.

(Body 3175 19050 0)

C.4.5. Everything column

An everything column prints the entire message: header, double CR, and body. Generally, only one everything column should be used in any form. The syntax for an everything column is:

(Everything <*LeftX*> <*RightX*> <*Font*>)

All parameters of the everything column have the same meanings as in the body column.

For example, the following everything column taken from the built-in Archive form prints the entire message in the text area of the page in font 0.

(Everything 3175 19050 0)

C.5. Laurel’s built-in forms

C.5.1. InternalMemo form

The InternalMemo form prints a message on the familar Bravo-style memo form. The herald "Laurel Message" appears at the top of the first page, and the fields are formatted. A Xerox logo appears after most of the header fields, with the In-Reply-To: field following if present in the message. The message body is printed, and the cc: field is printed last, with all names in that field printed on separate lines. Subsequent pages have a portion of the Subject: field printed as a heading.

```
HardcopyForm: InternalMemo 25400 2540 3175 19050 0 T
  ( Options
    ( Caption 4445 26670 2 T "Laurel Message" )
    ( Heading 3175 26670 0 F Subject 15875 )
    ( PageNumber 18415 26670 0 F )
  )
  ( Rows
    ( 0 0 35 ( Field 3175 10795 0 1 F T 4445 From "" T F T F )
      ( Field 12065 19050 0 1 F T 13018 Date "" T F T F ) )
    ( 0 0 0 (Field 3175 19050 0 1 F T 4445 PrintForm "" F T T F ) )
    ( 420 0 35 (OtherFields 3175 19050 0 1 F T 4445 500 ) )
    ( 1000 0 35 ( Caption 635 19050 3 XEROX ) )
    ( 1000 0 35 ( Field 3175 19050 0 1 F T 4445 In-Reply-To "" F F T F ) )
    ( 1000 0 35 ( Body 3175 19050 0 ) )
    ( 420 0 35 ( Field 3175 19050 0 1 F T 3810 cc c F F T T ) )
  )
```

C.5.2. Blank form

The Blank form prints a message on a slightly formatted, but much plainer form, than does the InternalMemo form.

```
HardcopyForm: Blank 25400 2540 3175 19050 0 T
  ( Options
    ( Heading 3175 26670 0 F Subject 15875 )
    ( PageNumber 18415 26670 0 F )
  )
  ( Rows
    ( 0 0 35 ( OtherFields 3175 19050 0 1 F T 4445 500 ) )
    ( 1000 0 35 ( Body 3175 19050 0 ) )
  )
```

C.5.3. Headers form

The Headers form prints a subset of the fields of the selected messages, one after another, without skipping to a new page before each new message. This form is useful for printing extracts of message headers similar to the Laurel table-of-contents. A caption on each page labels the fields printed underneath.

```
HardcopyForm: Headers 25400 2540 1800 20500 0 F
  ( Options
    ( Caption 8000 26670 0 T "Laurel Table of Contents" )
    ( Caption 1800 26300 1 T From: )
    ( Caption 6245 26300 1 T Subject: )
    ( Caption 15000 26300 1 T Date: )
    ( PageNumber 20000 26670 0 T )
    ( Caption 8000 26670 0 F "Laurel Table of Contents" )
    ( Caption 1800 26300 1 F From: )
    ( Caption 6245 26300 1 F Subject: )
    ( Caption 15000 26300 1 F Date: )
    ( PageNumber 20000 26670 0 F )
  )
  ( Rows
    ( 300 0 35 ( Field 1800 5610 0 1 F F 2000 From "" T F F F )
      ( Field 6245 14365 0 1 F F 6550 Subject "" T F F F )
      ( Field 15000 20500 0 1 F F 15300 Date "" T F F F ) )
  )
```

C.5.4. Archive form

The Archive form prints messages unformatted, one after the other without skipping to a new page. This form is useful for printing messages for long term storage, rather than as individual messages.

```
HardcopyForm: Archive 25400 2540 3175 19050 0 F
  ( Options
    ( PageNumber 18415 26670 0 T )
    ( PageNumber 18415 26670 0 F )
     )
     ( Rows
    ( 200 0 35 ( Caption 2000 19050 0
            "--------------------  Start of message  --------------------" ) )
    ( 200 0 35 ( Everything 3175 19050 0 ) )
     )
```

Appendix D. Laurel and MSG

This section addresses the relationship between MSG and Laurel. MSG users *must* thoroughly understand the contents of this section *before trying to use Laurel for the first time.* If you don't use Maxc or you have never used MSG, you may skip this section entirely.

Message.txt

If you are an MSG user on Maxc and become a Laurel user, then your Message.txt file on Maxc will become your Laurel in-box. Getting new mail (with Laurel) will move messages from Maxc to your Alto disk, *emptying Message.txt*. This makes it very inconvenient to continue to use both Laurel and MSG, for once you have mail files on your Alto disk, MSG can no longer access them. Furthermore, MSG and Laurel maintain mail files in formats that are incompatible. You can continue to use MSG to maintain mail files on Maxc, and you can use Laurel to maintain mail files on your Alto, *but you cannot move mail files back and forth.* If you want to convert completely to Laurel, you can move MSG-constructed mail files to your Alto for use by Laurel, as explained below. *However, this is a one-way street; once on your Alto the files cannot conveniently be moved back to Maxc and processed by MSG.*

Mail file philosophy

Laurel encourages you to use your default mail file (Active.mail) as a *temporary* storage area, not an archive. Accordingly, deleted messages are expunged whenever you leave Laurel or shift your attention to another mail file. There is no analogue of MSG's Quit command (which preserves deleted messages); Laurel's **Quit** is like MSG's Exit. If you wish to classify messages, use mark characters (section 2.3.2). Eventually, selective display on the basis of mark characters may be possible. Use separate mail files to obtain an archive facility. Laurel's performance will be better and your screen will be less cluttered.

Overwrite

Laurel doesn't have a command corresponding to MSG's Overwrite. However, the same effect can be obtained by selecting **Mail file** with BLUE.

Processing your mail away from an Alto

You can use MSG from any reasonable terminal dialed to Maxc or connected to the Arpanet, but you can't use Laurel without an Alto. Consequently, if you have no Alto available and would like to process your mail, you must use MSG. *If you observe a few conventions while using MSG,* you will not need to reprocess your old messages with Laurel. Thus, you can use MSG from a home terminal or when you are out-of-town, and revert to Laurel when your Alto is again at hand.

As long as you manipulate *only* your Message.txt file with MSG, Laurel will note what you have done when you perform a **New mail** command. (Remember, Laurel uses Message.txt as your in-box.) Specifically, deleted messages in Message.txt will not be retrieved, and the examined/unexamined status of each message retrieved will be reflected in the mark field in Laurel's table-of-contents.

Moving mail files from Maxc

Because MSG and Laurel have incompatible mail file formats, you cannot simply move your archival mail files to your Alto disk and expect Laurel to read them. The following procedure should be followed:

1. Start Laurel, and use the **Run** command to run Chat.laurel.

2. Using Chat, login to Maxc.

3. Using Chat, append the desired mail file, say *Name*.txt, to Message.txt:
   @Append *Name*.txt Message.txt CR
   If Message.txt is empty, you must undelete it first, by giving the command:
   @Undelete Message.txt CR

4. Point the cursor at **Mail file**, click RED, and supply the file name *Name*.

5. Point the cursor at **New mail** and click RED. After the messages have arrived, ensure that no unexpected ones have been included at the beginning and/or end of the file. (New mail might have arrived on Maxc before or after step 3.) Move any extraneous messages to "Active.mail".

6. Repeat steps 3 through 5 for each mail file on Maxc.

7. Logout from Maxc and Quit from Chat.laurel. You may now proceed in Laurel.

Fortunately, once you have moved all of your files in this way, you won't have to do it again, since Laurel will maintain them in its own format on your Alto disk.

Appendix E. Laurel 6 vs. Laurel 5.1 and LaurelX

The facilities available in Laurel 5.1 and LaurelX have been merged in Laurel 6. Where features of those two systems were in conflict, there is usually a profile option in Laurel 6 that chooses between them. Laurel 6 also has a significant number of changes and improvements over Laurel 5.1 and LaurelX. This appendix lists the most important of these changes.

Profile options

Several Laurel profile options are no longer supported in their former ways. The profile options Authenticate:, Herald:, HeraldFont:, Logo:, LogoFont:, Poll:, and DisplayErrorPups: are no longer recognized by Laurel. The Authenticate: option has been subsumed by the Registry: entry in your profile. The other entries that deal with hardcopy options are subsumed by the more general hardcopy forms (Appendix C).

The Laurel profile error reporting mechanism catches these entries and gives you the explanation:

> The profile field *x* is no longer supported.

where *x* is one of the above mentioned profile fields.

Message transport system

Grapevine is replacing the older MTP message transport implementation. If your registry is served by Grapevine, your password must be understood by that system. If you are in such a registry, and your name and password are not acceptable to Laurel, then you should contact an administrator for your registry to correct the password held for you by Grapevine.

The administration of public distribution lists in Grapevine registries is much more flexible than in registries servd by MTP (IFS's). Consult Appendix B, Maintain, for further details. Due to the change in the way distribution lists are stored, the results of a **Get** on a public distribution list may be slightly different than in the past.

The Laurel user interface

The arrangement of screen commands has been altered slightly from previous versions of Laurel. The **Delete** command has been moved next to the **Display** command, with the **Hardcopy** command now placed at the right margin. The **{**brackets**}** for several commands, e.g., for **Get** and **Put**, are now located on a second menu line that only becomes visible when you need it. Just invoke these commands in the normal way; their **{**brackets**}** will appear automatically.

The **Hardcopy** command now displays a property sheet with several settable parameters available. Try it, you'll like it.

You can make discontiguous selections in the table-of-contents region. See section 2.3.1 for details.

Facilities for filling in **{**brackets**}** have been expanded considerably. See section 2.1.3. for details.

Continuous scrolling now occurs when you hold a mouse button down continuously in the scroll bar. Various parameters that change the timing behavior of continuous scrolling are settable in your Laurel profile.

A new kind of selection, shifted selection, is an alternative to type-in whether you use the old-style modal editor or the newer modeless editor. See section 4.3.2 for details.

New screen commands, **Copy** and **Run**, are available. **Copy** provides FTP-like services; **Run** allows you to run various programs while still in Laurel. These commands are described along with all the Laurel screen commands in section 3.

The Laurel editor

The Laurel editor now comes in two flavors, modal and modeless. The modal editor is similar to the editor in Laurel 5.1; the modeless editor is similar to the one in LaurelX. A large number of new editor commands are available in either editor. See section 4 for details.

If you used Laurel 5.1 previously

The default settings for profile options are set to emulate the behavior of Laurel 5.1. Thus, Laurel 6 is upward compatible with Laurel 5.1. However, a substantial amount of functionality has been added to Laurel 6 that was prevously available in LaurelX only. You may wish to change some of your option settings or take advantage of additional facilities, so read on.

If you used LaurelX previously

The editor in LaurelX has become the Laurel 6 modeless editor. You will want to set the Editor: modeless option in your Laurel profile to enable the modeless editor. The Laurel 6 modeless editor does expand the functionality of the LaurelX editor considerably. See section 4 for details.

Laurel 6 will print on a HyType printer. You will need to put appropriate hardcopy forms in your Laurel profile to use the HyType properly.

To get the LaurelX style prompts in the feedback region, use the ErrorKeys: LaurelX option in the Laurel profile.

The **Editor** command of LaurelX, with its fast swapping to BravoX is gone. The only way to return to BravoX (or any other stand-alone program for that matter) is to quit from Laurel 6 and enter the other program normally through the Alto Executive.

Support for LaurelX is terminated. Laurel 6 is its supported replacement.

KEY NAMES:

	Microswitch (Alto I)	ADL (Alto II)
CANCEL	upper-right blank	BW
DO	ESC	ESC
NEXT	middle blank	next-to-bottom blank
PASTE	LF	LF
COM	lower-right blank	lower-left, upper-right, or lower-right blank

SCROLLING:

Scroll bar is on extreme left. Thumb bar is at top of editor window.

Scroll UP with Left button; places adjacent text at window top.

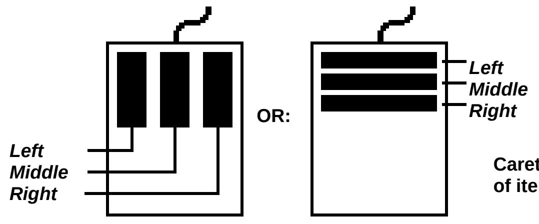
Scroll DOWN with right button; moves top of window adjacent to cursor.

Thumb with middle button by pointing at relative position on thumb bar.

Thumb marker indicates relative position of target in text.

Continuous scroll by holding down Left or Right button in scroll bar.

THE MOUSE:



Caret blinks at end of item nearest cursor

	Single Click Actions: IN TEXT AREA/IN LINE BAR	Multiple Click Actions:
Left	Select CHARACTER/LINE as target	Extend selection up thru hierarchy
Middle	Select WORD/PARAGRAPH as target	Extend selection up thru hierarchy
Right	Extend selection	Reduce selection down thru hierarchy

+SHIFT	make source selection
+CTRL	make delete selection
+both	make move selection

Selection hierarchy is:
character, word, line,
paragraph, everything

MOUSE BUTTONS PLUS SHIFT AND/OR CONTROL KEYS:

	IN TEXT AREA/IN LINE BAR
SHIFT+Left:	select CHARACTER/LINE as source
CTRL+Left:	select CHARACTER/LINE for deletion
CTRL+SHIFT+Left:	select CHARACTER/LINE for move
SHIFT+Middle:	select WORD/PARAGRAPH as source
CTRL+Middle:	select WORD/PARAGRAPH for deletion
CTRL+SHIFT+Middle:	select WORD/PARAGRAPH for move
SHIFT+Right:	extend source selection
CTRL+Right:	extends selection for deletion
CTRL+SHIFT+Right:	extend move selection

Pressing CANCEL before releasing all buttons will nullify actions

COMMAND KEYS (invoke by holding down COM and striking the command key)

A:	Append. Place caret after selection.
B:	Bracket target selection with placeholder brackets. ▶◀
D:	Delete target selection.
E:	Everything. Select everything with replace selection.
F:	Find. Type or select string for search. Accepts pattern for search with special characters as follow: # = any char; * = any string; @ = any alpha; & = alpha string; ! = any non-alpha; ~ = non-alpha string; ’x = use "x"; {} bounds selection.
G:	Get. Type or select file name. Contents of file is inserted at caret.
I:	Insert. Place caret before selection.
P:	Put. Type or source select file name. Puts selected text into file.
R:	Replace. Delete target selection (for modal editor compatibility).
S:	Substitute. Type or source select new text, then old text. Accepts pattern in "for" argument as in F command.
T:	Time. Insert date/time at caret.
U:	Undo. Undoes previous commands. See CANCEL.
’ " ([{ < - :	Bracket target selection with specified bracket type.
0, 1, ... , 9:	Set window boundaries to pre-set positions.

OTHER COMMANDS

BS or CTRL+A:	Delete previous character.	*COM+BS :*	Delete previous word.
SHIFT+BS:	Delete next character.	*COM+SHIFT+BS:*	Delete next word.
CTRL+W:	Delete previous word.		
SHIFT+CTRL+W:	Delete next word.		
DEL:	Delete target selection. Same as COM+D.		
DO (ESC):	Repeats previous text modification (only if target selection exists).		
CANCEL:	Undo previous text modification.		
PASTE (LF):	Insert the previously deleted text at caret.		
NEXT:	Replace select the next placeholder.		
SHIFT+NEXT:	Replace select the previous placeholder.		

Laurel 6 Modeless editor command summary

Figure 2.