ALTO USER'S HANDBOOK

SEPTEMBER 1979

XEROX
PALO ALTO RESEARCH CENTER

ALTO USER'S HANDBOOK

SEPTEMBER 1979

XEROX
PALO ALTO RESEARCH CENTER
3333 COYOTE HILL ROAD PALO ALTO CA 94304

Table of Contents

ALTO NON-PROGRAMMER'S GUIDE

by BUTLER W. LAMPSON

Preface

This handbook contains documentation for all the standard Alto services intended for use by non-programmers. It is divided into seven sections, separated by heavy black dividers:

The *Alto Non-programmer's Guide*, which has most of the general information a non-programmer needs.

The *Bravo* manual, which tells you how to prepare and edit text documents on the Alto.

The *Laurel* manual, which tells you how to send, receive, and manipulate messages using our inter-Alto electronic mail system.

The *Markup* and *Draw* manuals, which tell you how to add illustrations to documents. The Non-programmer's Guide contains some introductory material on illustrations.

Finally, two reference manuals, one for FTP, which transports files between machines, and one for Neptune, which provides facilities for managing files on your Alto disk. These manuals supplement the introductory information on these two programs in the Non-programmer's Guide.

If you are new to the Alto, start at the beginning of the Non-programmer's Guide. Read the first four sections there, and then the first two sections of the Bravo manual. Fairly early you should also learn about Laurel, since much day-to-day communication takes place using it. After that, you should be able to find what you need by looking at the tables of contents and browsing through the rest of the material. If you have trouble, don't hesitate to ask an expert for help.

This handbook was originally prepared by Butler Lampson, and the present edition is the responsibility of Ed Taft.

Alto Non-programmer's Guide

Table of Contents

1. Introduction

This document is intended to tell you what you need to know to create, edit, and print text and pictures on the Alto. It doesn't assume that you know anything about Altos, Maxc, IFS, or any of the other facilities available to you.

You will find that things are a lot clearer if you try to *learn by doing*. This is especially true when you are learning to use any of the services which use the display. Try out the things described here as you read.

Material in small type, like this paragraph, deals with fine points which can be skipped on first reading (and perhaps on subsequent readings as well).

Much of the documentation in this Guide is intentionally incomplete. More comprehensive information about almost all of the programs and services described here may be found in various on-line documents. Section 9 contains a summary of these and instructions for obtaining your own copies of any that you need.

2. Getting started

To do anything with an Alto, you must have a disk pack. This is a circular, yellow or white object about 15 inches in diameter and 2 inches high. Your secretary can tell you how to obtain a new one from the stock kept by your organization.

INITIALIZE YOUR DISK

The next step is to get the disk *initialized* with copies of all the programs you will need to use. Here is how to do this:

Obtain the disk pack labeled BASIC NON-PROGRAMMER'S DISK. Find an Alto that has two disk drives, each with four square lights, a white switch and a slanted plastic window. Load the BASIC NON-PROGRAMMER'S DISK into the drive labeled 0. You do this as follows:

> The drive should have the white switch in the LOAD position, and the white LOAD light should be lit. Open the door by pulling down on the handle. Put in the disk by holding it flat, with the label facing you, and pushing it gently into the drive until it stops. Then gently close the door and push the white switch to RUN. The white LOAD light will go out, and after about a minute the yellow RUN light will go on. The disk is now loaded and ready to go. If anything else happens, you need help.

On many double-disk Altos, the two disk drives are not labelled. The drive mounted inside the same cabinet as the Alto is drive 0. and the one sitting on top of the cabinet is drive 1. Also, some Altos have a switch on the back of the keyboard housing labelled NORMAL/ALTERNATE. Be sure that this switch is in the NORMAL position.

Now start the Alto. This is done by pushing the small button on the back side of the keyboard, near the thick black cable. Pushing this button is called *booting* the Alto. It resets the machine completely, and starts it up working on the disk you have just loaded. After you boot the machine, it will tell you at the top of the screen what it thinks the state of its world is, and then it will print a ">" about halfway down the screen. When the screen looks like that, anything you type will be read by the Executive, whose basic job is to start up the service you want to run. There is a section on the Executive later in this document. For now, you will find everything you need to know right here.

You are going to use a program called CopyDisk, which copies everything on the main disk (which you just loaded) onto another disk which you will load into the disk drive labeled 1. This copying erases anything which is already on the disk in disk drive 1, so you should be very careful not to copy onto a disk which has anything you want to keep. Load your new disk into the disk drive labeled 1, doing just what you did to load the BASIC NON-PROGRAMMER'S DISK into drive 0.

The CopyDisk program is not present on the BASIC NON-PROGRAMMER'S DISK, but it is available through a service called the NetExec, which can load a small number of commonly-used programs from the Ethernet. To start the NetExec, type

>NetExec^{CR}

The ^{CR} stands for the carriage RETURN key on the keyboard. In this and later examples, what you type is underlined in the example, and what the Alto types is not. On the screen, of course, there won't be any underlining. It doesn't matter whether you capitalize letters or not; the capitalization in this manual is chosen to make reading easier.

Within a few seconds, the NetExec will start up. The screen will look much the same as it did while the Executive was running, but with "Net Executive" in the upper left corner. Now type

>CopyDisk^{CR}

After a few more seconds, CopyDisk will start up, identify itself, and display a prompt of "*". You

should now go through the following dialogue:

*Copy from: DP0CR *the digit zero, not the letter O*
Copy to: DP1CR
Copying onto DP1 will destroy its old contents.
Are you sure this is what you want to do? [Confirm] Yes
Are you still sure? [Confirm] Yes

Now CopyDisk will copy the contents of the BASIC NON-PROGRAMMER'S DISK (DP0, "the Disk Pack in drive 0") onto your new disk pack (DP1). This takes about two minutes. While it is running, it records its progress by moving the cursor from the top of the screen to the bottom; this happens twice: once while the disk is being copied and again while the copy is being checked for correctness. When it is done, if all went well it will display the message "Done. DP0 and DP1 are now identical." followed by the "*" prompt. Now type

*QuitCR

to exit CopyDisk and return control to the Executive. If something goes wrong, the message "Copy complete, but do *not* trust DP1" will appear. This means that there is something wrong either with the Alto or with one of the disk packs. Consult your local support staff.

Now you can take both disks out of the machine. Before you do, you should tell the Executive that you are finished, by typing

>QuitCR

You will see that after a couple of seconds the screen goes blank and starts to display a white square that jumps around. This is an indication that the *memory test* program is running properly; an Alto should always be left in this state when it is not being used.

Now take out both disks, by pushing the white switch on each drive to LOAD. The yellow READY light should go out, and about 25 seconds later the white LOAD light should go on. Now you can open the door (against a *slight* resistance) and remove the disk. Put the BASIC NON-PROGRAMMER'S DISK back where you found it.

If you cannot find an Alto with two disk drives, you can do a CopyDisk from one standard (single-drive) Alto to another: the procedure for doing this is described in section 8.6. Since it is a little more complicated than the method just given, a novice should use it only as a last resort.

LABEL YOUR DISK

Before doing anything else, put a label on the new disk with your name, and any other identifying information you like. This is best done by preparing a paper label that can be slipped underneath the plastic insert on the front edge of the disk pack. Now you can take the new disk to any Alto, load it in, boot the machine by pushing the button on the back of the keyboard, and start working.

NAME YOUR DISK

When you do this, if you look at the information displayed at the top of the screen just after you do the boot, you will see that it says

---- OS Version x/x ----- Alto #xxx ----- NoName ---- Basic Non-programmer's Disk ----

This is because your new disk is an exact copy of the BASIC NON-PROGRAMMER'S DISK, which has no owner, and owner and disk name information got copied along with everything else. To give the disk your own name as owner, you should type

>InstallCR

to the Executive. It will ask you whether you want the "long installation dialogue"; answer No. When it asks you for your name, type in your Maxc or IFS account name (usually just your own

last name), followed by a CR. When it asks you for a disk name, choose a suitable one and type that in, again followed by a CR. Next it will ask you whether you want to give your disk a password. If you do this, the Alto will ask you for the password every time you boot it, and won't let you do anything until you provide it correctly. This provides a modest level of security for the information on your disk. If you do give your disk a password, it is best to use your Maxc or IFS password, since the Alto will then know it and use it automatically whenever you communicate with Maxc or IFS. *Don't forget the password, since there is no simple way to find out what it is, and you will need an expert to get access to anything on your disk.*

There will be a pause for a few seconds, and then the Executive will come back. If you assigned a password to your disk, you will be asked for it first. Now your name is installed on the disk, and the system will display it near the top of the screen whenever the Executive is in control, and will put it on the cover page of anything you print.

After you initialize a disk, you have to edit your *user profile*, discussed in section 2.4 of the Bravo manual, and your *Laurel profile*, described in section 3.6 of the Laurel manual. If you are reading this manual for the first time, you will be told how to do these things at the appropriate points. This is mentioned here so that you will remember it next time you initialize a disk.

3. The Executive and the NetExec

The Executive is the program to which you are typing right after a boot, and whenever any other program finishes its job. It has a large display area in the middle where your typing and the Executive's responses appear. Above this the Executive displays a digital clock and some other useful status information: the versions of the Executive and the operating system, the owner name and disk name installed on the disk, the Ethernet address of the Alto you are using, and the number of free pages on the disk. Whenever you call another program, the Executive's display is erased and replaced by that of the program that you called.

3.1 Correcting typing errors

When you are typing at the Executive and you make a mistake, there are a few special keys you can type to correct the mistake. The BS (backspace) key erases the last character you typed. Holding down the CTRL key and typing W erases the last word you typed. The act of holding down CTRL and typing W is called control-W and is denoted by W^C, and similarly for other control characters. The DEL key cancels the command you were typing completely; it prints "XXX", and then starts a new line with a fresh ">" character. Nearly all programs accept A^C (control-A) as a substitute for BS.

3.2 Starting a program

As we said before, the Executive is for starting up other programs which do the work you want done. To start a program called Alpha, you just type

>Alpha^{CR}

It doesn't matter whether you type in capitals, lower case, or a mixture of the two. If the program needs some other information about what to do, you type that after the name of the program. For example, there is an Executive command to type out a document on the screen. Suppose you want to type out the document called "Notes". You just say

>Type Notes^{CR}

The Executive won't ever do anything until you type the final CR; if you change your mind, just type DEL to cancel the command any time before you type the CR.

Certain operations, such as Type, are performed entirely by the Executive itself, whereas most others are performed by separate programs (also called *subsystems*) kept on your disk or obtained from the Ethernet. Ignore this distinction for now.

3.3 Aborting

You can usually stop what is going on and get back to the Executive by holding down the left-hand SHIFT key and striking the SWAT key, which is a blank key in the lower right corner of the keyboard on Alto-Is, in the upper right corner on Alto-IIs. If this doesn't work, you can push the boot button.

If you push the SWAT key while holding down both CTRL and SHIFT, you will find yourself talking to a service called Swat which is of no interest to non-programmers. Usually no harm is done if this happens; you can get back to what you were doing before by typing P^C (control-P; hold down the CTRL key and type P).

3.4 The NetExec

The NetExec is a program much like the Executive in that its main purpose is to start up other programs for you. Unlike the Executive, it loads programs from a *boot server* available via the Ethernet rather than from your own disk. The NetExec makes available certain programs (such as hardware diagnostics) that are used infrequently and that most users won't wish to keep on their own disks. Also available are several programs useful for recovering from various sorts of disasters that may make it impossible for you to invoke the normal Executive.

The NetExec may be started in either of two ways. If the Executive is already running, you may simply type

>NetExec^CR

A fuzzy cursor will appear in the center of the screen for a few seconds, and then the NetExec will start up. If the Executive is not running, you can invoke the NetExec directly by holding down the BS key and the ' (quote) key and then pushing and releasing the boot button. Keep the keys pressed down until you see a fuzzy cursor in the center of the screen; this can take up to 5 seconds.

The NetExec's display looks much like the Executive's, but the herald contains the words "Net Executive". The type-in conventions are identical to the Executive's. To start up a program from the NetExec, simply type the name of that program followed by CR.

Any time this manual instructs you to "use the NetExec to invoke *p*", where *p* is the name of some program, you should follow the above procedure. An example of this was given in the instructions for using CopyDisk to initialize your disk (section 2).

The Executive also has a few commands for invoking programs directly from the Ethernet, without your first having to start up the NetExec. At the present time, these programs are Chat, FTP, and Scavenger. More precisely, these commands will obtain the correspondingly-named programs from your disk if they are present and from the Ethernet otherwise.

4. Files

The Alto stores on your disk all of the material you are working on (text and pictures), as well as most of the programs which provide the various services described here. The named unit of storage on the disk is called a *file*. Each different document you handle will be stored on its own file. The facilities for identifying files are not ideal, but you will get used to them after a while. Better facilities are the subject of current research.

A file is identified by its *name*, which is a string of letters (upper and lower case can be used interchangeably), digits, and any of the punctuation characters + − . ! $. A file name can have two parts, which are called the *main name* and the *extension*; they are separated by a period. For example, "Alto.Manual" is a file name, with main name "Alto" and extension "Manual". File names cannot have blanks in them, or any punctuation characters except the ones just mentioned. A file name must not have more than 39 characters; most people don't notice this restriction.

4.1 Naming conventions

It is important to name your files in some systematic way, using the extension to tell what kind of file it is, and the main name to identify it. For instance, useful extensions might be Memo, Letter, Note, Figure, Calendar. If you are a secretary keeping material for several people on one disk, you can stick the person's initials in front of the extension, e.g. BWLmemo, JGMmemo etc. If you don't have anything specific in mind, it is customary to make the extension the same as the name of the program that creates the file, e.g., Report.bravo for a document that doesn't have any special properties, and is written using Bravo.

Here is a modest list of extensions commonly encountered on Non-programmers' disks:

.al	Alto display-format font file
.boot	A file that can be booted from
.bravo	Bravo-format text file
.cm	Command file for the Executive or other programs
.image	Runnable Mesa program (subsystem)
.press	Press-format file (suitable for printing)
.run	Runnable Bcpl program (subsystem)
.~	An Executive command that is executed directly by the Executive (there is no actual file corresponding to this name).

The Alto doesn't care whether you capitalize letters in file names or not (i.e., ALPHA and alpha and aLpHa refer to the same file), but it is a good idea to use capitalization to make names more readable. This is especially useful when a name consists of more than one word, since blanks are not allowed in file names: e.g., TripReport or MasterList.

4.2 File name patterns

The Executive provides some simple facilities for handling files. First of all, it allows you to name a group of files by using file name *patterns* containing the magic characters "*" and "#". The "*" character stands for any string of characters. For example, the pattern "*.memo" stands for all the files which have the extension "memo", and the pattern "*.BWL*" stands for all the files which have BWL as the first three characters of the extension. The "#" stands for any single character; for instance, "# # #.memo" stands for all the files which have a three character main name and the extension "memo". If you are curious to see what a pattern expands into, you can type X^C immediately after typing it to get it expanded.

If you type a file name or a pattern to the Executive, and then type a TAB, it will give you a list of all the files whose names start with that name. So, for example, typing

```
>*.BWL^TAB
```

will get you a list of all files which have an extension starting with the characters BWL.

Another useful thing to know is this: if you are in the process of typing a file name to the Executive, and you type ESC, it will add as many characters as it can to complete a file name. If you type "?", it will give you a list of all the files which start with what you have already typed; you can then go on and finish the file name.

Here is a summary of magic characters for getting file names expanded:

ESC	completes the file name if possible; if not, completes as much as it can, and flashes the screen.
TAB	shows you all the file names which match what you have typed since the last blank, and erases what you typed.
?	like TAB, but doesn't erase anything.
X^C	retypes the command line with all file name patterns replaced by the list of file names they expand to.

There are two more simple commands for dealing with files. To delete a file, or a group of files, type

>Delete *file1 file2* ...CR

Warning: once you have deleted a file, you cannot get it back. Proceed with caution. If you have enabled version numbers and there is more than one version of a file, the one with the lowest version number gets deleted.

To get the contents of a text file printed on the screen, type

>Type *file*CR

If the contents won't fit on the display, the Alto will show you as much as will fit, then ask if you want to see more. If you do, just type a space; if you want to stop, type No.

When the Executive is running, it displays two lines of status information near the top of the screen. Included in this information is the amount of space which is left for storing files. This space is measured in *disk pages*; it takes about 5 disk pages to store one page of text. It is prudent to keep at least 150 disk pages available; if your disk has fewer, you should delete some files, perhaps after sending them to a file server (see section 6).

At this point you know enough to use Bravo to begin creating and editing text. Bravo is described in its own manual, which is part of the Alto User's Handbook. You should start reading the Bravo manual, and not try to continue with this guide until you have become familiar with the material in the first two sections of the Bravo manual. The remainder of this guide contains more information about the Alto which you won't need on the first day, but will probably want in the first week.

Because much of our day-to-day communication takes place by means of our Alto-based electronic mail system, you should also start learning to use Laurel. Begin by reading the first two sections of the Laurel manual, which is also part of the Alto User's Handbook.

5. Recovering from disasters

There are various ways in which your Alto disk can become damaged. If this does happen, the procedures described in this section will almost always allow you to recover the disk, or at worst will let you copy files from the sick disk to a healthy one. It is probably a good idea to get some help with this if you are not experienced.

Here are the symptoms of trouble:

You can't boot the disk and get to the Executive.

You are out of disk space, but you think you should have plenty; in other words, some disk space has apparently gotten lost.

You get an error message from some service which says something about disk errors or file errors, and perhaps recommends that you should run the Scavenger.

You hear a funny buzzing noise from the disk for a couple of seconds, after which the service you are using breaks in some way.

It may be that the problem is caused by an incompatibility between the disk drive on which your disk pack was written, and the disk drive on which you are trying to use it. This is a likely cause of your problems only if you have been moving the pack from one machine to another, and if you notice that it works properly on some machines but not on others. If your problem is caused by disk incompatibility, the procedures described below won't do you much good. Instead, you should report the problem to the hardware maintenance staff, so that the offending disk drive can be realigned, and make yourself a new disk pack on a machine known to be in alignment. You can transfer files from the old pack to the new one using the procedure described in section 6.5.

The first step is to run a program called Scavenger. If your disk is healthy enough to let you boot and use the Executive, you can just invoke the Scavenger by typing

>Scavenger^CR

If it isn't, you should use the NetExec to invoke Scavenger, as described in section 3.4.

If that doesn't work, hold down just the BS key and press the boot button; this should give you the dancing white square of the memory diagnostic. If it doesn't, either your Alto's Ethernet connection is broken, or the *boot server* that provides Alto programs over the Ethernet is down or unavailable. Either find another Alto without these problems, or load in a disk which is still in good shape and has the Scavenger program on it, invoke the Scavenger, and then unload the good disk and load your sick one.

The Scavenger will ask you whether you want to change disks, and give you a chance to do so if you say Yes. Then it will ask you if it can alter your disk to correct errors; say Yes.

The Scavenger will now work for about a minute. As it runs, it may ask you whether it is OK to correct "read errors". If they are "transient" errors, answer Yes fearlessly; if they are "permanent" errors, it is best to ask for advice from an expert. When the Scavenger is done, it will tell you what it found. If it has succeeded in making your disk healthy, you can go about your work. Delete the files Garbage.$ and ScavengerLog which the Scavenger leaves around. It is a good idea to go through this scavenging procedure once a month or so, just to keep your disk in good shape.

If things are still in bad shape (i.e., you can't boot and run the Executive), the next step is to boot the NetExec again and type

>NewOS^CR

This should get you a fresh copy of the operating system, which will ask you whether you want to Install. You should say Yes, and go through the Install procedure described in section 2. If all goes well, you will then find yourself talking to the Executive and can proceed normally.

If this doesn't work, there is one more step to try. Boot the NetExec again and this time type

>FTP^CR

This should get you the FTP program described in section 6.3; use it to transfer the files <Alto>Executive.run and <Alto>SysFont.al from a file server (Maxc or IFS). Then boot the Scavenger as described above and run it again. If

this fails, you should consult an expert. If no expert is available, you can boot FTP again, and use it to transfer files from your broken disk to a file server or to a clean disk on another Alto (made using the procedure described in section 2).

The Scavenger leaves all the stuff which it wasn't able to put into a recognizable file on a file called Garbage.$, and it leaves a readable record of everything it did on another file called ScavengerLog (unless it tells you that you have a beautiful disk). There are two kinds of entries in ScavengerLog: names of files removed from the directory or otherwise modified, and names of file pages which were put into Garbage.$.

5.1 Reporting problems

If your Alto itself is broken, obtain a trouble report form, fill it out, and leave it in the proper place; procedures for doing this depend on your location.

If you have trouble with Bravo, report it using the procedure in section 4.3 of the Bravo manual. If you have trouble with Laurel, see section 4 of the Laurel manual.

For other problems, consult your local expert.

6. File servers

Many uses of the Alto require you to communicate with a shared *file server*, which can store files belonging to a number of different users. A file server's disk typically has hundreds of times the capacity of your own Alto disk. Moving non-current files to a file server is a way to free up space on your own disk for other things, and a file server is a good place to put documents you want other people to be able to get at.

Each organization typically has at least one file server of its own. To make any use of your organization's file server, you must first obtain an account and password; to do this, consult your local support staff. Your account name will usually be your own last name, and your password should be six or more characters long and unpronounceable.

There are presently two types of file server, IFS and Maxc. An IFS (which stands for Interim File System, as improved facilities are under development) is an Alto with some large disks attached to it and dedicated to the one task of being a file server. Many IFSs exist at present. Maxc is a large, general-purpose time-sharing system, only one of whose purposes is to be a file server. There are only two Maxc systems (called Maxc1 and Maxc2), both of which belong to Parc.

If you are at Parc, you should obtain accounts both on Ivy (the name of Parc's IFS) and on Maxc. If you are outside Parc, you will need an account on your organization's own IFS, and you will have to find out its name. At present, many users outside Parc are also assigned accounts on Maxc because the Laurel message system uses Maxc as a central post-office, and you have to have a Maxc account in order to keep a mailbox there. This state of affairs is likely to change in the near future.

While IFS and Maxc are fundamentally different sorts of machines, in their capacities as file servers they are substantially the same. We shall first describe the facilities common to the two types of systems. Later sections will deal with additional facilities unique to one or the other.

6.1 Logging in

Before trying to access a file server from your Alto, you should first tell the Alto your account name and password. If you have given your account name to Install as the owner name for your disk, however, the Alto already knows it, and if you used your file server password as your disk password, it knows that too and you can skip to section 6.2. Otherwise, you can give the necessary information by typing to the Executive:

>Login^CR

You will now be asked for your name and password. Type in each one in turn, ending each with a CR or space. Note that it assumes your file server account name is the same as your disk owner name; if this is the case, you can just type CR to confirm it, and go on to give your password. If it isn't, type DEL, and then give the account name you want to use. Once you have done this Login, your Alto will automatically identify you to file servers whenever necessary. *If you boot your Alto, it will forget this information, and you must Login again.*

Note that Login only records your name and password; it does *not* connect you to a file server. If you don't do a Login, programs that access file servers will automatically ask you for the Login information when they first run, and will record it just as Login does.

If you wish, you can supply a password for your disk when you Install (see section 2). If you do this, you will have to type the password whenever you boot the Alto, but it will be used automatically as your file server password, unless you override it with a Login command. The password is stored on your disk in encrypted form, so it cannot be stolen by someone who paws around on your Alto disk.

6.2 About files on IFS and Maxc

IFS and Maxc file names look very much like Alto file names, but they have two more parts: a *directory* and a *version number.* On IFS, the format is

<directory>name.extension!version

whereas on Maxc the format is

<directory>name.extension;version

Each user who has an account on a file server has his own directory, named by his user name. If you name a file without specifying the directory, you will refer to your own directory. You can reference files in some other directory simply by prefixing the directory name to the file name, as illustrated.

When you put a file onto a file server, if there is already a file with the same name, the new file is added, with a version number one bigger than the old one. When you reference a file, you get the one with the largest version number if you don't specify which one. As you can see, it is almost never necessary for you to specify a version number explicitly.

There is a protection system, not described here, which allows you to control which other users can read or write your files. The usual setting of the protection, and the one you will get automatically if you don't say anything special, allows all Xerox users to read the file, but prevents anyone except the owner from writing it.

On IFS, but not on Maxc, files within a directory may be organized into *sub-directories.* For example, the file named

<Jones>Memos>ActivityReport.bravo!3

belongs in directory Jones, sub-directory Memos. You can have as many sub-directories as you wish within your own directory. You can even have sub-directories within sub-directories, to as many levels as you wish, subject to an overall limit of 99 characters in each file name.

Most commands that accept a remote file name also permit you to type a *file name pattern,* much as does the Alto Executive (section 4.2). In this case, the command is repeated for all files that match the pattern. If the file server is an IFS, the magic character "*" (but not "#") may appear anywhere in the name, just as on the Alto; if the file server is Maxc, however, "*" must appear in place of an entire field of the file name (directory, main name, extension, or version), and other uses of "*" are not permitted. Thus, "*.memo" and "<Jones>*.*" are legal file name patterns on both Maxc and IFS, but "*.BWL*" is legal only on IFS. Note that to refer to *all versions* of each file you must specify "*" for the version field explicitly (that is, "!*" on IFS and ";*" on Maxc); otherwise, the pattern will match only one version (usually the highest-numbered).

6.3 Transferring files

You can transfer files between your Alto and a file server using FTP, the File Transfer Program. This program has a fairly elaborate set of features, which are described fully in the FTP Reference Manual found near the end of the Alto User's Handbook. This section tells you enough about FTP to take care of most ordinary needs. After you have become familiar with this material, you might skim over the FTP manual once, just so that you are aware of the other facilities that are available.

After starting FTP, you will see three windows on the screen; from top to bottom, they are the *server* window, the *user* window, and the *Chat* window. Most interactions with FTP involve only the middle window—note the blinking vertical bar there, which shows where you can type. The first step is to type the name of the machine you want to talk to. For example, if your file server is named Ivy, you should just type

*Ivy^{CR}

In a second or two you should get back a response like

Ivy (Parc) file server 1.16 9-Apr-78

If the file server is not operating, there will be delay of about a minute before FTP gives up trying to contact it; you can give up sooner by striking the blank key to the right of CR.

Now you can *retrieve* a file from the file server, or *store* a file into it. To retrieve, you type

*Ret rieve remotc file Example as local file Example [New file] CR

As in the Executive, you can just type enough of the command to identify it uniquely, and then a spacc; unlike the Executive, FTP supplies the rest of the command name automatically. You then type the remote file name, followed by a space. If the file server has a file by that name on your directory, FTP will then suggest a local name for the file followed by "[Old file]" or "[New file]", depending on whether or not the file already exists on your Alto disk. If you like the name, you can just type CR. Otherwise, you can type some other name, as in the following example:

*Ret rieve remote file Example as local file Dummy [New file] CR

Note that the name originally suggested by FTP will disappear as soon as you start to type the new one. If you decide you don't want to retrieve the file after all, strike DEL to cancel the command.

After you type the terminating CR, FTP will transfer a copy of the file from the file server to your Alto disk. During the transfer, the cursor will flip its two black squares back and forth every time it transfers a block of the file, so you can tell how fast it is progressing from the frequency of flips. When FTP is done, it will tell you how long the file is, followed by "Done" and the "*" prompt. If you gave a *file name pattern* for the remote file, FTP will repeat the above procedure for each file that matches the pattern. You may strike DEL to skip over any files that you don't want to retrieve.

To store a file presently on your Alto disk onto the remote machine, you type

*St ore local file Example as remote file ExampleCR

or *St ore local file Example as remote file DummyCR

again depending on whether or not you want to use a different name. In this case, the message "[Old file]" or "[New file]" will not appear; remember, however, that storing a file on a file server ordinarily creates a new version rather than overwriting an existing file with the same name. The Store command does not accept file name patterns.

If you were not logged in at the time you started FTP, you will be asked for your user name and password when you do the first Retrieve or Store. FTP will save this information so that you won't have to provide it again until the next time you boot the Alto.

You can do as many Retrieve and Store commands as you want. When you are done, type

*Q uit

and you will be back talking to the Executive.

If you intend to do a lot of transfers to or from a directory other than your own, you can say

*Dir ectory OtherDirCR

to make <OtherDir> be the default directory for remote file names; this saves typing <OtherDir> in front of each name. Similarly, if you want to refer repeatedly to a sub-directory on an IFS file server, you can say, for example,

*Dir ectory Jones>MemosCR

Directory protections are ordinarily such that you cannot store into directories other than your own. However, if you know the password of some directory, you can *connect* to it by saying

*Con nect to directory OtherDir Password xxxxxCR

which not only directs FTP's attention to that directory (just as does the Directory command) but also gives you complete access to it, as if you were its owner. The password is not displayed when you type it, of course. File servers generally have many so-called *files-only* directories that don't belong to any particular person but rather are for use by a group or project. The Connect command is the way you gain access to such a directory, particularly when you want to store files into it.

6.4 Listing and deleting files

You can get a list of the remote files that match a file name pattern with the List command; for example,

*List *.bravo^CR

or *List <Jones>Memos>*^CR

It is quite slow, however, and there is no way to interrupt it except to SHIFT-SWAT out of FTP. You are better advised to obtain such information using the Chat program, to be described shortly.

To delete a remote file, type, for example,

*Del ete Example^CR

After displaying the full name of the file, FTP will ask you to confirm or cancel your intentions. If there are multiple versions of the same file, the lowest-numbered version will be deleted.

6.5 Transferring files to and from another Alto

In addition to accessing a file server, FTP can communicate with any other Alto that is also running FTP. By this means, you can transfer files directly from one Alto disk to another.

To do this, boot the disk and start up FTP on the second Alto. Suppose its name is Banjo. If you don't know what its name is, you can use its network address, which FTP shows at the very top of the screen (e.g., "3#326#"). Now go to the first Alto, start up FTP, and tell it to connect to the second Alto simply by typing its name (much as you would type the name of a file server); for example,

*Banjo^CR

or *3#326#^CR

Now you may retrieve and store files as usual. Of course, in this case remote file names will not contain directories or versions (unless version numbering is enabled on the other Alto). Also, file name patterns will not be accepted in any context.

When you start FTP on an Alto, it is normally ready to act as a remote machine or *server* as just discussed. If you don't say anything special, it will allow any other machine to retrieve files, and to store new files, but not to overwrite existing files. You can change these defaults by starting FTP with

>Ftp/*x*

where *x* can be: −S (no server) to prevent any such transfers; Protected to allow retrieving only, but no writing; Overwrite to allow existing files to be overwritten. Any server activity is reported in the server window at the top of the screen.

6.6 Access via Chat

The File Transfer Protocol (the means by which FTP communicates with a file server) limits itself to the basic set of operations already described. There is an escape mechanism that lets you get at some important additional file server facilities directly, using a program called Chat. Chat uses the Alto display to simulate a traditional, "dumb" computer terminal, and thereby enables you to talk directly to Executive programs running in the file server machines themselves.

Maxc, being a general-purpose time-sharing system, has a large array of commands and programs; only the ones directly concerned with Maxc's role as a file server are described here (a few additional facilities are presented in section 6.8). The IFS Executive's command repertoire is limited to operations on files. Most of the commands described here are the same on IFS and Maxc; however, there are a few differences which you should note carefully if you use both systems.

To initiate a conversation with the Executive in a file server named, say, Ivy, just type

>Chat Ivy^{CR}

If all goes well, you will see the message "Connected to:" followed by some numbers at the top of the screen, a message from the server's Executive immediately below that, and "@" at the left margin prompting you for type-in. If Chat has trouble getting connected, it will tell you its problem after trying for a few seconds. This usually means that the server is broken; you might try again in a few minutes.

The next step involves logging into the server Executive. Chat may do this for you automatically, depending on the version of Chat you are using and the file server you are accessing. In this case, if you have forgotten to Login to your Alto, Chat will first ask you for your account name and password. Otherwise, you must type

@Login (user) *name* (password) *password*^{CR}

When the server types more than a screenful at you, it will pause after every screenful and "ring the bell", which causes Chat to display a large DING at the top of the screen. After you have had a chance to read the screen, striking any key on the keyboard will get the server to produce the next screenful. If you type ahead, this feature is suppressed.

Whatever the server Executive is doing, you can force it to stop by typing C^c. On Maxc, you may have to type C^c several times in quick succession to get it to stop.

When you are finished talking to the server Executive, type

@Logout^{CR}

(or "Quit" if the server is an IFS). This will terminate your Chat session and return control to your own Alto's Executive. If you are connected to Maxc, Chat won't terminate until about two minutes after you log out; to give control back to the Alto Executive immediately, type SHIFT-SWAT.

If the file server is an IFS, you will be logged out automatically if you don't type anything for two minutes. This is because IFS can service only a small number of users (currently five) at once; the automatic logout is intended to prevent IFS from being tied up by users who aren't doing anything useful.

Chat keeps a record of your conversation on a file called Chat.scratchScript. You can read it with Bravo after a Chat session, just to see what happened, or perhaps to copy things out of it into other files, print it, or whatever. There are two funny things about this file which you need to know about:

> The file is not erased when you start a new conversation. Instead, the typescript of the new conversation starts at the beginning of the file and continues for as long as the conversation lasted. The end of the conversation is marked by the characters <=> after which you may see the remnants of previous conversations.
>
> The typescript file is only 20,000 characters long. If your conversation is longer than that, the typescript will wrap around to the beginning. It is possible to make the file larger by editing the [CHAT] section of the user profile (the file User.cm) in the obvious way.

You can also initiate a Chat-like conversation with a file server while you are running FTP. At the bottom of FTP's screen is a "Chat window" (actually labelled "User Telnet"), in which you can talk directly with a file server's Executive in much the same manner as you do with Chat. You can move the blinking cursor down into the Chat window by striking the unmarked key to the right of right-SHIFT; to get back to the middle window, strike the unmarked key to the right of RETURN. In the Chat window, after typing the server name followed by CR, you can log in and do whatever you want. This window isn't very large and doesn't offer all the conveniences of Chat itself, but at times it is nice to be able to switch very quickly between transferring files and giving commands to the server Executive. You can make the Chat window larger when you start up FTP by typing FTP/-S, which prevents the FTP server from being started and thereby eliminates the upper (server) window.

6.7 Server Executive commands

You type commands to IFS and to Maxc in more-or-less the same way (except for those commands that have different names on the two systems); however, the responses from IFS and Maxc are usually somewhat different. The examples below illustrate IFS's responses. You may type "?" at any point to obtain a brief explanation of what you are expected to type in next. Maxc normally does not display the remainder of abbreviated commands or the explanatory text in parentheses; however, you can force it to do so by terminating fields you type in with ESC rather than space.

To generate a list of the names of all your files, type

on IFS: @List^CR *on Maxc:* @Dir^CR

To list only those files matching some file name pattern, say, "Activity.*", type

on IFS: @List (files) Activity.*^CR *on Maxc:* @Dir Activity.*^CR

To list another user's directory, type

on IFS: @List (files) *<OtherUser>*^CR *on Maxc:* @Dir *<OtherUser>*.*^CR

Caution: note carefully that the command names are different on IFS and Maxc. Worse, there is also a List command on Maxc, but with an entirely different meaning (it causes hardcopy to be generated). Be careful!

You can obtain more detailed information about your files (length, date written, etc.,) by typing a comma immediately before the CR; for example,

on IFS: @List (files) Activity.*,^CR *on Maxc:* @Dir Activity.*,^CR

At this point, the server Executive will type "@@" at the left margin and permit you to type in one or more *sub-commands* that modify the action of the main command; in the case of List and Dir, subcommands are used to specify what information you wish to see about the files. Some of these are:

@@Type^CR	file type and byte size
@@Size^CR	size in pages
@@Length^CR	length in bytes
@@Creation^CR	date of file creation
@@Write^CR	date on which the file was last written
@@Read^CR	date on which the file was last read
@@Times^CR	times as well as dates
@@Author^CR	name of user who created the file
@@Verbose^CR	same as Type Size Write Read Author
@@Everything^CR	everything known about the file

After you have typed in one or more sub-commands, type just CR in response to the "@@" prompt. The server will now perform the main command and list out the files with the information you requested. The columns of printout will be aligned correctly only if the font Chat is using is a fixed-width font, such as Gacha12 or Gacha10.

You can delete one or several files (or all files matching some file name pattern), just as on the Alto, with

@Delete *file1 file2* ... ^CR

On IFS, the server Executive will now print out each file name and ask you to confirm your intention to delete it (type Yes or No); this is because once a file is deleted it is gone forever. On Maxc, all the files are deleted immediately without further confirmation; however, there is an Undelete command that you can use immediately afterward if you change your mind.

To delete all old versions of files (i.e., all but the highest-numbered version of each file), type

on IFS: @Delete *,CR *(note the comma)*
@@Keep 1CR
@@CR

on Maxc: @DelverCR
Delete oldest? YesCR
Delete 2nd newest? YesCR
File(s): CR

It is a good idea to do this fairly frequently, since old versions of files can pile up and waste a lot of space.

To find out how much space you are using on the file server, type

@DskStatCR

One IFS or Maxc page is equivalent to about four Alto pages. You will notice that you also have a *disk limit* which is the maximum number of pages you are permitted to use on the file server at one time. If you exceed your disk limit, the server won't let you store any more files until you first delete some existing ones to get you below your disk limit. To get your limit changed, consult your local support staff.

You can direct your attention to some other directory by typing

@Connect (to directory) *OtherDir* (password) *password*CR

just as in FTP. You may omit the password when connecting back to your own directory, or when connecting to a directory belonging to a project of which you are a member.

For the sake of security, it is a good idea to change your password occasionally (say, once a year). To do this, type

@Change Password (of directory) *name* (old password) *xxx* (new password) *yyy*CR

where *name* is your account name. The new password should be six or more characters long and unpronounceable (this is not enforced, however). Note that, contrary to normal practice, the new password *does* print out when you type it; this is so that if you make a typing mistake you will be able to see it.

6.8 About Maxc

Maxc's Executive is thoroughly documented in its own manual, which was written primarily for programmers and contains a large amount of information not needed by casual users of Maxc. The next few paragraphs document Maxc facilities likely to be of interest to Alto non-programmers. If you don't use Maxc you may skip this section.

If you have a file on Maxc in Press format (see section 7.1 for an explanation of file formats), you can tell Maxc to print it directly by issuing the "Press" command. This is documented in section 7.5.

Maxc provides facilities for *archiving* files onto magnetic tape, where the cost of storing them is negligible. You can get an archived file back within one day.

To archive one or several files, type

@Archive File *file1 file2* ...CR

(Note that the command name consists of the two words "Archive File"; after that you should type the names of the files you want to archive.) The files will be archived onto tape within a day or

two. After this has been done, they will be deleted from the disk automatically, and you will get a message notifying you that the archiving has been done.

Maxc keeps track of your archived files in an *archive directory* which you can list exactly like your regular Maxc directory, using the Interrogate command rather than the Directory command; for example,

@Interrogate *.bravo^{CR}

If the listing is of just one file, Maxc will ask you whether or not you want it retrieved from the tape. If you say Yes, the file will appear on your Maxc directory within a day, and you will get a message to that effect. It should be noted that the Interrogate command has some peculiarities not shared by other commands; in particular, you sometimes have to type an extra CR at the end of the command in order to get it to do anything.

Because Maxc's disk capacity is fairly small relative to the number of users who have Maxc accounts, the disk occasionally becomes full and it becomes necessary for a *forced archive* to be performed in order to make some space available. In a forced archive, all files that haven't been referenced (retrieved, printed, or whatever) in the past 90 days are written onto tape and deleted. You will be notified when any of your files are archived for this reason, and the procedure for getting them back is the same as given above.

7. Printing

The subject of printing is somewhat complicated because of the large number of variables involved. To begin with, there are many different *programs* that you can use to prepare documents for printing—Bravo, Draw, Markup, Sil, etc. Then there are various *file formats* defining the representation of documents stored in files—Bravo-format, Press-format, plain text, etc. Finally, there are several different types of *printers*—Dover, Sequoia, Slot/3100, Pimlico, etc.

This section first presents some introductory information on programs, file formats, and printers. Following that are a few of the most common procedures for printing documents.

7.1 Programs and file formats

Each of the programs you can use to prepare documents deals with information in a particular format. Bravo deals with text interspersed with special information about looks (fonts, paragraphs, etc.), and document files written by Bravo's Put command are in *Bravo format*, which only Bravo can read. The same can be said about Draw, Sil, and a number of other programs.

In order to be printed, a document generally has to be in *Press format*, which is a file format designed principally for representing printed pages. It follows that to print a document that is in some other format you must first convert it to Press format.

Programs that have their own special document formats also provide facilities for generating documents in Press format. For example, when you are using Bravo and you issue the Hardcopy command, Bravo first converts the document you are working on into Press format (this is why Hardcopy takes so long), then sends the resulting Press file to a printer. By using the File option of the Hardcopy command, you can tell Bravo to write the Press file on your disk rather than sending it to a printer.

The important point is this: a given document can be represented in several different formats. When you issue the Put command in Bravo you store the current document as a Bravo-format file, but when you issue the Hardcopy command with the File option you store the same document as a Press-format file. But while these are two different representations of the same document, only the Bravo-format file can be read back into Bravo (using the Get command), and only the Press-format file can be sent to a printer. *You can't read a Press file back into Bravo.* This is why it is important to choose file names in such a way as to identify their formats—the extension .Bravo to identify Bravo-format files, .Press for Press-format, .Draw for Draw-format, etc.

There are a few programs that deal exclusively with Press files. Markup is an illustrator that can both read and write Press files. PressEdit is a program used to manipulate Press files in various ways, such as combining several small Press files to create a single large Press-format document.

7.2 Printing servers

To print a document you send it through the network to a *printing server*, which is an Alto connected (usually) to some Xerographic printing device. The means by which you do this depend both on what kind of document you have and what type of printing server you are sending it to.

There is a bewildering array of names you will hear associated with printing servers. These names fall into three categories. There are *family names* that identify the type of printing device attached to the server Alto—a Dover is based on a Xerox 7000 copier, a Sequoia on a 3100, a Pimlico on a 6500, etc. Then there are names for different types of *printing software* used on the server Alto to control the printing device—Spruce and Press are the principal ones in use at present. Finally there are names identifying *individual printing servers*—the names by which you specify which particular

server is to print your documents. Examples of names of printing servers in Palo Alto are Clover, Menlo, Daisy, and Turkey.

To complicate matters further, while every printing server accepts Press files to print, there are different kinds of Press files and not all printers are capable of printing every kind of Press file. A Press file can contain a wide variety of information: text, straight lines, smooth curves, raster-scanned images, and even color. Obviously, if you send a Press file containing color to a black-and-white printing server you will not obtain the results you might desire; but there are other restrictions as well, most arising from technical considerations that will not be discussed here. As a rule, however, most printing servers will attempt to print any Press file as best they can and will tell you about whatever difficulties they may have encountered.

Here are some brief descriptions of the types of printing servers presently in use.

Dover is the predominant work-horse printer. It is based on a Xerox 7000 copier and is capable of continuous high-volume output (one page per second). It is best at printing documents consisting primarily of text, though it has limited capabilities for printing simple illustrations such as those produced by Sil. More complicated graphics (e.g., curves produced by Draw or raster-scanned images produced by Markup) can only be printed crudely if at all. Also, Dover prints pictures containing large solid black areas very poorly.

Sequoia is a smaller and slower printer, based on a Xerox 3100 copier. It can print more complex pictures than can Dover, and it prints solid black areas very well.

Pimlico is a Xerox 6500-based color printer. TC-200 is based on a Xerox Telecopier. Versatec is an electrostatic printer, some models of which can print on very large sheets of paper. Slot/3100 is similar to Sequoia. All these printers are capable of printing any kind of Press file (with the exception, of course, that color information is ignored by black-and-white printers).

Certain Press files contain copies of graphical information in two or more alternate forms. For example, curved lines produced by Draw are represented in the Press file both by mathematical descriptions and by Alto screen-resolution bit maps. Printing servers that understand the mathematical descriptions will use them to produce smooth curves, whereas servers that don't understand the mathematical descriptions—principally Dover and Sequoia—will print the bit maps instead, producing rather crude curves.

7.3 Fonts

There is a large array of *fonts* available for printing text. All fonts are named according to a standard convention:

family-name point-size face

The *family-name* describes the overall style of the font; e.g., TimesRoman or Helvetica. The *point-size* specifies the height of the font in points (one point is 1/72 inch). The *face*, if present, describes one or more additional properties of the font: B for bold, L for light, I for italic, C for condensed, and E for expanded. For example, Helvetica10B is a font in the Helvetica family, 10 points high, bold-face; TimesRoman12BI is TimesRoman, 12 points high, bold-face, italic.

A Press file containing text includes the *names* of the fonts to be used to print the text, but does not include any information about what the characters actually look like. Rather, each printing server maintains the printing representation of the set of fonts used most commonly by users of that printer. Many servers have a limit on the number of different fonts they can keep (related to the size of the disk attached to the server Alto). If you try to print a Press file containing text in a font the printing server doesn't know about, the server will substitute some other font and will tell you about this.

Programs such as Bravo that format text according to how wide the individual characters are obtain

the necessary information from a standard *widths* file named Fonts.widths, which must be present on your Alto disk. Additionally, in order to display text on the Alto screen you must have the appropriate *screen fonts* contained in files named *font-name*.al. You can usually obtain these files from the <AltoFonts> directory in your file server. See section 4.6 of the Bravo manual for further information on how Bravo deals with fonts.

Here are a few fonts that most printing servers know about. There are samples of some of them at the end of the Bravo manual.

Family	*Point-sizes*	*Faces (aside from normal)*
TimesRoman	8, 10, 12	B, I, BI
Helvetica	7, 8, 10, 12	B, I, BI
	6, 18	B
Gacha	8, 10, 12	I
Cream	10, 12	B, I, BI
Math	8, 10	
Hippo	8, 10	
Arrows	10	

You should consult your support staff to find out what fonts are available on your local printing servers.

7.4 Printing from your Alto

After you first initialize your disk and before you attempt to print anything, you must edit your user profile (file User.cm) to declare the name of the printing server you intend to use regularly. See section 2.4 of the Bravo manual. All programs except Laurel that generate hardcopy look in User.cm to find out where to send Press files to be printed. For Laurel, you must also edit your Laurel profile, file Laurel.profile, described in section 3.6 of the Laurel manual.

Generating hardcopy directly from Bravo is easy: you just issue Bravo's Hardcopy command, described in section 2.4 of the Bravo manual. If instead of just printing hardcopy you wish to distribute a document on-line (say, by storing it on a file server so as to make it available to other Alto users), then rather than distributing the Bravo-format document you should make a Press-format file and distribute that. To do this, use the File option of the Hardcopy command and specify a file name with extension .Press.

Once you have a Press file on your disk (having either created it yourself or retrieved it from a file server), you can send it to your printing server using the Empress program. If you just type

>Empress *filename*CR

one copy of the document will be printed by your default printing server. If you want more copies or you want to print on a different printing server, you can use the /C and /H switches; for example,

>Empress 3/C Menlo/H *filename*CR

will cause three copies of the document to be printed by the printing server named Menlo.

To print a file that is in some other format, say Draw or Sil, you must first create a Press-format version of that file. The means by which you do this are described in the appropriate manual, i.e., in the Draw manual for Draw-format files, the Sil manual for Sil-format files, etc. Once you have a Press file you can print it using Empress as just explained.

There is an additional file format called *plain-text*. Basically it is a text file containing no font information and no formatting. You can create a plain-text file using Bravo if you start with an empty window and never type any CTRL-CRs or looks, but do type ordinary CRs at the ends of lines. A non-programmer is unlikely to encounter plain-text files very often; but, for example, User.cm,

Laurel.profile, and Executive command files (see section 8.2) are plain-text files, as are documentation files with extension .tty that you obtain from a file server (section 9.2).

You can print a plain-text file using Bravo Hardcopy, but there is an easier and much faster way: just type

>Empress *filename*CR

Empress will discover that the file is plain-text rather than in Press format, and will convert it into Press format before transmitting it to the printing server. Empress will normally use a single font, Gacha8, for this purpose. You can change this to something else by editing the [HARDCOPY] section of User.cm to include a line such as

FONT: TimesRoman 10 B

The foregoing procedures for sending Press files to printing servers apply to those printers that run in *server mode*, i.e., that wait for someone to send them a Press file over the network and automatically print any files they receive. There are some printers that do not operate in server mode, usually because they cannot safely be left to run unattended. To print a document on one of these printers, you have to go to that printer's Alto, use FTP to retrieve the Press file you want to print, and type

>Press Print *filename*CR

There are sometimes additional operating procedures which you will find posted near the printer.

7.5 Printing from Maxc

If you want to print a Press document or a plain-text file that is stored on Maxc, you can retrieve it to your Alto using FTP and send it to your printing server using Empress, as already explained. However, you can alternatively tell Maxc to send the file directly to the printer.

Before you do this, you must tell Maxc the name of your printing server. You do this by creating (with Bravo) a plain-text file containing the single statement

PDEVICE *server*CR

where *server* is the name of your printing server. Then Put onto file DocGen.prt, Quit, and use FTP to transfer DocGen.prt to your directory on Maxc.

Whenever you want to print a document that is stored on Maxc, you should connect to Maxc's server Executive using Chat and issue the command

@Press *filename*CR

where *filename* is the name of a Press-format file (extension .press) or a plain-text file (e.g., extension .tty). You cannot use the Press command to print other kinds of files—in particular, Bravo-format files.

If your DocGen.prt file on Maxc also contains the line

REPORT YCR

then Maxc will attempt to notify you as soon as it has actually sent the file to the printer. Maxc will display a message on your screen if you are still connected to Maxc's server Executive at the time; otherwise, it will send you a message that you will receive next time you run Laurel.

7.6 PressEdit

You can compose the various parts of a document with Bravo, Markup, Draw, Sil, or other programs that produce Press files, and then put together the complete document with a program called PressEdit. You can also use PressEdit to extract pages from an existing Press file. PressEdit has some other features, some of which are documented here and the remaining ones in the Alto Subsystems manual.

The use of PressEdit for assembling documents has one major advantage: the resulting complete document can be left on a file server for printing by anyone who needs a copy. If you are producing a document for large-scale printing outside, on the other hand, it is probably easier to assemble it by hand than to go through all this ritual. One restriction you should be aware of is that every printing server has a limit on the size of Press file that it can handle (this is principally a function of the capacity of the disk connected to the server Alto). Most printing servers can handle Press files up to about 50 pages long (printed pages, not Alto disk pages), though some can handle documents substantially larger than that. If you have a very large document, you should distribute it as several Press files, each containing no more than 50 pages.

The simplest use of PressEdit is to append together two or more smaller Press files to create a single large one. For example,

>PressEdit Manual.press ← Chapter1.press Chapter2.press Chapter3.press^{CR}

creates Manual.press by concatenating the three other Press files in the order given. Be sure to type a space both before and after the "←". You should also remember that the new Press file will occupy as much disk space as the three existing Press files combined; check that you have enough free disk pages before you start.

You can copy selected pages out of a Press file and put them into a new Press file by a command such as

>PressEdit Short.press ← Long.press 3 6 10 to 14 18 to 22^{CR}

This extracts pages 3, 6, 10 through 14, and 18 through 22 from Long.press and puts them in Short.press. Note that the numbers refer to consecutive pages in the source Press file, counting from 1, and have nothing to do with any page numbers that actually appear on the pages themselves.

The concatenate and extract operations may be combined in one command to produce a document with pages from two or more source documents interleaved. This is particularly useful for inserting illustration pages into text documents. For example,

>PressEdit Report.press ← Text.press 1 to 3 Figures.press 1 Text.press 4 to 8 Figures.press 2 Text.press 9 to 14^{CR}

This produces a document consisting of pages 1 through 14 of Text.press, with page 1 of Figures.press inserted between pages 3 and 4 of Text.press and page 2 of Figures.press between pages 8 and 9.

If you want to make a document that has pages containing both text and illustrations, there are two ways to *merge* selected pages of different Press files. Both techniques are unfortunately rather cumbersome. The first method involves interleaving the pages of the text and illustration Press files, as just described, and then using Markup to copy material from one page to another in the resulting file. This procedure is documented in section 4 of the Markup manual. It is very slow and requires a lot of manual labor, and it does not always work for illustrations produced by any program besides Markup.

The second technique requires you first to put special marks in the text and illustration Press files to show how you want the illustrations to be positioned. You then run PressEdit, which merges the source Press files automatically to produce the desired document.

Each illustration must be contained in a separate, one-page Press file. Somewhere in the illustration must appear an "arrow" consisting of the following piece of text:

<==<<

There should be no spaces or other characters either before or within this piece of text; you must position the arrow using the text positioning facilities of the illustrator you are using.

The main text document must also contain an arrow to show the position of every illustration. Inside each arrow must appear the name of the Press file that contains the illustration to be inserted there; for example,

<<=<Fig3.press<

Again, there must be no spaces or other characters either before or within the arrow; you must position it by setting the left margin appropriately and possibly by using the vertical tab feature—see sections 3.2 and 3.6 of the Bravo manual.

Having prepared all the Press files, you merge them by typing a command of the form

>PressEdit/M Document.press ← Text.press Fig1.press Fig2.press Fig3.press^CR

After the "←" you must type the name of the main text file followed by the names of all the illustration files you are inserting into the final document. You may list the illustration files in any order, and if you are inserting a particular illustration into more than one place in the final document, you need type its name only once.

Now, each time PressEdit encounters an arrow in the text document, it merges into that page the illustration contained in the Press file whose name is inside the arrow. PressEdit positions the illustration on the page by aligning the two arrows, then removes the arrows.

PressEdit also has a facility for adding fonts to Press files. This is useful primarily when you want to edit the Press file with Markup and add text in some font that does not already appear in the file. For example, to add fonts Logo24 and Helvetica12 to file Example.press, type

>PressEdit Example.press ← Example.press Logo24/F Helvetica12/F^CR

The next time you read Example.press into Markup, the new fonts will appear in Markup's font menu in addition to the ones that were there before.

8. Other things

This section describes various facilities and procedures that you will probably find useful at some point. Browse through them now just so you know where to find them.

8.1 Copy and Rename

To copy one file to another, say

>Copy *new* ← *old*^CR *don't leave out the spaces*

The "←" is to remind you of the direction the copying is done.

To change the name of a file, say

>Rename *new* ← *old*^CR *don't leave out the spaces*

or: >Rename *old new*^CR

There must not already be a file called *new*. However, if *old* and *new* differ only in capitalization, Rename may be used to change the capitalization.

8.2 Command files

If you have a sequence of Executive commands that you wish to execute repeatedly, you may put them into a *command file*, then invoke the command file at any later time. To create a command file, use Bravo to enter the exact commands that you would issue to the Executive, then Put the document onto a file with the extension ".cm", the standard extension for command files.

To execute a command file named, say, Cleanup.cm, type

>@Cleanup@CR

The Executive will display the first command in the command file, perform the command, and then go on to the next command, continuing until it has executed all the commands in the file. To abort execution of a command file, type C^C, which will stop it at the end of the current command, or SHIFT-SWAT, which will stop it immediately.

Actually, you may substitute the contents of a command file for any part of an Executive command line. For example, if file ListOfFiles.cm consists entirely of the text "Alpha Beta Gamma Delta" (with no CR at the end), and you type

>Delete @ListOfFiles@ EpsilonCR

the effect will be exactly the same as if you had typed

>Delete Alpha Beta Gamma Delta EpsilonCR

A number of commonly-used programs, most notably FTP, are capable of accepting their own commands from the Executive command line used to invoke them. Normally, when you start up FTP simply by typing

>FtpCR

FTP then expects you to type commands to its own keyboard command interpreter. But if you start it up with, say,

>Ftp Ivy Store/C Memo1.bravo Report2.pressCR

FTP will make a connection to the Ivy file server, store the files Memo1.bravo and Report2.press on your Ivy directory, and return control to the Executive, with no further action on your part, except to type in your password if you haven't already logged in.

This capability may be used in conjunction with command files to permit you to deal with large groups of files all at once. For example, if you have a set of Bravo documents comprising one large report, you can create a command file, say, Report.cm, containing the names of those files, then issue the command

>Ftp Ivy Store/C @Report.cm@CR

to transfer a complete, consistent set of those files to a file server.

You should be aware that the language used to control FTP from the command line is *not* the same as that used to control it from the keyboard, though it is similar. You should read section 4 of the FTP manual before attempting to create FTP command lines.

8.3 Dump files

The Executive's Dump command gives you a way to package up a number of files into a single, so-called *dump file*. You can then transport the dump file around as a unit, and later recover one, a few, or all of the files from it using the Load command. This is especially useful in maintaining consistent sets of related files.

To make a dump file, type

>Dump Alpha.dm *file1 file2* ... CR

Here "Alpha.dm" is the name of the dump file; by convention it has the extension "dm." You can list as many files as you want to be dumped. Often the * feature of the Executive is useful here, and of course you may obtain the list of files from a command file.

To get files back from a dump file, type

>Load/v Alpha.dmCR

You will get a list of the files in Alpha.dm, and after each one you will be asked whether you want it loaded or not. If you leave out the /v all the files which don't already exist will be loaded; if you say /c instead, all the files will be loaded whether or not they are already on your disk.

The FTP program has facilities for accessing dump files on a file server: you may transport a collection of files on your disk to or from a remote dump file without ever having to put the dump file on your disk. That is, the command

>Ftp Ivy Dump/C Alpha.dm *file1 file2* ... CR

will package together files *file1*, *file2*, etc., and store them as Alpha.dm *in the file server*, not on your Alto disk. Similarly,

>Ftp Ivy Load/C Alpha.dmCR

will access Alpha.dm on Ivy and load the constituent files onto your Alto disk. You will probably find that this is more convenient than using the Executive's Dump and Load commands. See sections 3 and 4 of the FTP manual for complete information.

8.4 Neptune and DDS

The Neptune program provides convenient facilities for managing files on your Alto disk. Basically, it displays the names of all the files on your disk in a window that you can scroll just as in Bravo, and it permits you to specify operations on individual files simply by pointing at their names with the cursor. It can delete files, display the contents of text files, and (if your Alto has two disk drives) copy files from one disk to another.

The Neptune manual is included at the end of the Alto User's Handbook. If you read sections 1 through 4 of that manual you will know enough to start using Neptune.

There is another program called DDS which is considerably more powerful than Neptune and provides a number of useful capabilities that Neptune lacks. However, it is rather slow and takes up a great deal of space on your disk, so it is not included on the BASIC NON-PROGRAMMER'S DISK. You should try it out and see whether its features are valuable to you. The DDS manual appears as part of the Alto Subsystems manual (see section 9.2).

8.5 Illustrators

There are currently three major programs for drawing pictures on the Alto:

Markup: good for pictures involving images, free-hand drawing or painting. Markup is also useful for adding pictures to a text document produced by Bravo; these pictures can come from Draw or Sil, or they can be drawn by Markup itself.

Draw: good for pictures which contain lines, curves and text.

Sil: good for forms and pictures with only horizontal and vertical lines. For such pictures it is much faster than either Markup or Draw.

Manuals for Markup and Draw are included in later sections of the Alto User's Handbook. Sil is also suitable for general use; unfortunately, the present Sil documentation is rather terse and is oriented principally toward users of the Design Automation system, of which Sil is a part. You can obtain this documentation, such as it is, by printing <Sil>SilManual.press.

8.6 CopyDisk

The simplest use of the CopyDisk program is copying the contents of one disk pack to another on an Alto equipped with two disk drives; it was described in section 2. CopyDisk can also copy the contents of a disk pack from one Alto to another over the Ethernet. To use it in this mode, you need two Altos; in the example below they are called Banjo and Flash.

Put the disk you want to copy *from* into one Alto (say, Banjo), and use the NetExec to invoke CopyDisk (see section 3.4). Put the disk you want to write *onto* into the other Alto (Flash), and start CopyDisk on that Alto also. You will type all commands on Flash, i.e., the Alto containing the disk you want to write *onto.*

You should now go through the following dialogue:

```
*Copy from: [Banjo]DP0^CR                    the digit zero, not the letter O
Copy to: DP0^CR
Copying onto DP0 will destroy its old contents.
Are you sure this is what you want to do? [Confirm] Yes
Are you still sure? [Confirm] Yes
```

In the above example, in response to "Copy from:" you type the name of the *other* Alto (the one you are copying *from*) in square brackets, followed immediately by "DP0" (with no intervening spaces). If you don't know the name of that Alto, you can instead type its Ethernet address (displayed in the black bar on that Alto's screen) followed by a #. In response to "Copy to:" you type simply "DP0", meaning the disk pack in drive 0 of the Alto on whose keyboard you are typing.

After you have confirmed your intentions, the copy should proceed. When CopyDisk is done, if all went well you will see the message "Done. [Banjo]DP0 and DP0 are identical" followed by the "*" prompt. You may now type Quit^CR to each Alto.

8.7 Version numbers

There is an optional *version number* facility that permits you to keep multiple versions of a particular file on your Alto disk without having to invent a different name for each one. This can be particularly valuable when you are making a number of successive changes to a document but want to keep earlier versions around in case you change your mind. Files stored on file servers always have version numbers, but use of version numbers on Alto disks is optional.

Unfortunately, the version number capability has not been integrated fully into all Alto programs (in particular, the Laurel message system). Furthermore, the relatively small amount of storage available on an Alto disk makes wholesale maintenance of multiple versions impractical. For these reasons, the version number facility is *disabled* on the BASIC NON-PROGRAMMER'S DISK. If you wish to use it, you may enable it by doing an Install, asking for the "long installation dialogue", and answering the questions appropriately.

If the version number facility is enabled, a file name may end with an exclamation point followed by a number: for example, "Alto.Manual!4" is version 4 of the file Alto.Manual. The basic rule for version numbers is this:

> When you read a file, you get the one with the largest version number (the *current* version), unless you include the version number you want in the file name.
>
> When you write onto a file for which the current version is n, a new version $n+1$ is created, and becomes the current version, unless you include the version number in the file name. Furthermore, if version $n-1$ was around, it gets deleted, so that just two versions of the file are kept, the current one (with the largest version number) and the next earlier version. The number of versions kept may be changed at Install time.

For example, if version 4 is the current version of the file Alto.Manual, there will probably be "Alto.Manual!4" and "Alto.Manual!3" around. If you write onto "Alto.Manual" (e.g. by doing a Put from Bravo), "AltoManual!3" will disappear, and "Alto.Manual!5" will appear with the new information on it. "Alto.Manual!4" will still be around unchanged, so you can get the old version back from there if you need it. On the other hand, if you write onto "Alto.Manual!4", that file will be changed, and no new versions will be created.

If a file name doesn't have a version number, most programs will not make any new versions, but will just write on the single version. Bravo is an exception; it always makes new versions if the version number facility is enabled.

9. Software distribution and documentation

Alto software and documentation are publicly available from file servers. Most file servers maintain duplicate copies of common files. In Palo Alto, most software and documentation of interest to non-programmers is stored on Maxc; in other places, on the local IFS. You should consult your support staff for the exact maintenance policies used in your organization.

9.1 Obtaining new software releases

When new versions of the various programs are released, they are normally announced by messages to all registered Alto users. You can obtain a new version of a service called, say, Alpha as follows:

> If the release announcement includes instructions for installing the new version of the program, follow those instructions. Otherwise:
>
> Using FTP, attempt to retrieve <Alto>Alpha.cm from your file server. If this succeeds, leave FTP and type to the Executive
>
> >@Alpha.cm@CR
>
> This will cause FTP to be invoked again, some files to be transferred from your file server, and perhaps some other activity. When everything settles down, you will have the new version.
>
> If there is no <Alto>Alpha.cm, retrieve <Alto>Alpha.run. This will be the new version of the program. You don't have to do anything else.

It is a good idea to keep fairly up-to-date. New versions of programs are sometimes released to fix serious bugs or to reflect important changes in operating procedures. If you run into trouble while running an obsolete version of some program, you are unlikely to receive much help or sympathy from the program's maintainer or from your local support group.

The best way to obtain a complete set of new software, and clean up your disk at the same time, is to obtain a fresh disk, initialize it from the BASIC NON-PROGRAMMER'S DISK as described in section 2, and then use FTP to transfer all the files you want to keep from your old disk to the new one, as described in section 6.5. If you have an Alto with two disk drives, you can put the old disk in one drive and the new one in the other, then use Neptune to copy files between disks. See sections 1 through 4 of the Neptune manual.

An alternative way to make a BASIC NON-PROGRAMMER'S DISK is to put the disk you want to initialize into an Alto, boot the NetExec as described in section 3.4, then type

>NewOS^CR

You will get a fresh version of the operating system, which will ask you if you want to Install. Say Yes, ask for the "long installation dialogue", and say that you want to erase a disk. After a minute or so, you will have a clean disk with nothing on it except the Executive and FTP. Use FTP to retrieve the file <Alto>NewNpDisk.cm from your file server. Then type

>@NewNpDisk^CR

This will automatically transfer all the needed files from the file server, and do any other necessary initialization. It takes about 20 minutes, and puts a significant load on the file server, so use this procedure only when you can't find the BASIC NON-PROGRAMMER'S DISK. During the operation, there will be an automatic Install of the operating system; answer its questions appropriately. You will have to type your name and password at various points in the procedure. There will also be an automatic initialization of Bravo, and you should do a Quit when it is finished.

9.2 Documentation

Documentation for all the standard Alto software can be found in the <AltoDocs> directory on Maxc and other file servers. As a rule, each major piece of documentation appears as a Press file which you can obtain and print by means of the procedures explained previously (section 7). Short documents are available as files with the extension "tty"; these are plain text files that you can transfer to your Alto and read with Bravo or print with Empress.

To see what is available, you can Chat to your file server and type

on IFS: @List <AltoDocs>*.press <AltoDocs>*.tty

on Maxc: @Dir <AltoDocs>*.press <AltoDocs>*.tty

Here is a short guide to on-line documentation likely to be of interest to non-programmers but not already contained in the Alto User's Handbook. All are stored on the <AltoDocs> directory except where noted otherwise.

Alto Subsystems, files Subsystems1.press and Subsystems2.press.

This manual, which is about 150 pages long, contains documentation for a large number of Alto programs. Included is comprehensive information on Chat, CopyDisk, Empress, and the Executive, which are only partially described in the Alto User's Handbook. An additional program of interest to non-programmers is DDS, which enables you to display and manage your Alto disk's file directory in ways much superior to Neptune.

Alto Subsystems Catalog, files SubsystemsCatalog.press.

This is a summary of Alto subsystems, organized by function.

Sil, Analyze, Gobble, Build Reference Manual, file <Sil>SilManual.press.

Sil is an illustrator specialized for very rapid composition and editing of pictures consisting of straight lines and text. It is part of the Design Automation system for digital logic development, but Sil is useful in its own right as a general-purpose illustrator.

Printing at Palo Alto, file Printing.press.

This is a comprehensive summary of Press printing facilities, formats, and programs. Despite its title, it is applicable more-or-less everywhere.

How to Use IFS, file <IFS>HowToUse.press.

This is a complete user's manual for IFS, presenting a fair amount of material not covered in the Alto User's Handbook.

The Alto User's Primer, file AltoUsersPrimer.press.

This document contains some introductory material on Alto hardware and software. It is oriented towards newcomers to the Whole Alto World, and its main purpose is to describe what exists and how to get it.

Whole Alto World Newsletter.

This is a monthly newsletter that serves as a vehicle for communication of Alto-related information among Alto users throughout Xerox. Each month's edition is stored as WAWNews*m-yy*.press; for example, WAWNews8-78.press is the August 1978 edition.

BRAVO

by BUTLER W. LAMPSON

Bravo Manual

Table of Contents

Preface

This manual describes the Bravo system for creating, reading and changing text documents on the Alto. It is supposed to be readable by people who do not have previous experience with computers. You should read the first four sections of the Non-Programmers Guide to the Alto before starting to read this manual.

You will find that things are a lot clearer (I hope) if you try to *learn by doing.* Try out the things described here as you read.

Material in small type, like this, deals with fine points and may be skipped on first, or even second, reading.

This manual is written on the assumption that you have the user profile, fonts and other Bravo-related material from the BASIC NON-PROGRAMMER'S DISK. If this is not the case, some of the things which depend on that stuff will not work the same way.

There is a one-page summary of Bravo at the end of this manual. It is intended as a memory-jogger, not as a complete specification of how all the commands work.

Bravo was designed by Butler Lampson and Charles Simonyi, and implemented mainly by Tom Malloy, with substantial contributions from Carol Hankins, Greg Kusnick, Kate Rosenbloom and Bob Shur.

1. Introduction

Bravo is the standard Alto system for creating, editing and printing documents containing text. It can handle formatted text, but it doesn't know how to handle pictures or drawings; for these you should use Draw, Markup or Sil.

When you start up Bravo (do it now, by saying Bravo/e^{CR} to the Executive), you will see two *windows* on the screen, separated by a heavy horizontal bar. The top one contains three lines with some useful introductory information; it is called the *system window.* The bottom one contains a copy of the material you are reading, which was put there because of the "/e" you typed to the Executive. If you had omitted the "/e", as you do when using Bravo normally, the bottom window would be empty, except for a single triangular **endmark** which indicates the end of a document. In the bar separating the two windows is the name of the document in the lower window.

As you do things in Bravo, the first two lines of the system window will give you various useful pieces of information which may help you to understand what is going on and to decide what you should do next. Usually, the top line tells you what you can do next, and the second line tells you what you just did, and whether anything went wrong in doing it. *Make a habit of looking at these two lines while you are learning Bravo, and whenever you are unsure of what is happening.*

No matter what is going on in Bravo, you can stop it and get back to a neutral state by hitting the DEL key. You can leave Bravo and get back to the Executive by typing

Quit^{CR}

The characters which you type (Q and CR) are underlined in this example; the characters which are not underlined are typed by Bravo. This convention is used throughout the manual. Notice that you only type the first character of the Quit command; this is true for all the Bravo commands.

Each Bravo window (except the top one) contains a *document* which you can read and change. Usually you read the document from a file when you start Bravo, and write it back onto a file after you have finished changing it. Later, you will find out how to do this (section 2.3). It is possible to have several windows, each containing a document; this too is explained later on (section 4.2).

Bravo is controlled partly from the keyboard and partly from the *mouse*, the small white object with three black buttons which sits to the right of the keyboard. As you push the mouse around on your working surface, a *cursor* moves around on the screen. Pushing the mouse to the left moves the cursor to the left, pushing the mouse up (away from you) moves the cursor up; and so forth. You should practice moving the mouse around so that the cursor moves to various parts of the screen.

The three buttons on the mouse are called RED (the top or left-most one, depending on what kind of mouse you have), YELLOW (the middle one) and BLUE (the bottom or right-most one). They have different functions depending on where the cursor is on the screen and what shape it has. Don't push any buttons yet.

Mouse lore:

You will find that the mouse works better if you hold it so that it bears some of the weight of your hand.

If the cursor doesn't move smoothly when the mouse is moving, try turning the mouse upside down and spinning the ball in the middle with your finger until the cursor does move smoothly as the ball moves. If this doesn't help, your mouse is broken; get it fixed.

You can pick the mouse up and move it over on your work surface if you find that it isn't positioned conveniently. For instance, if you find the mouse running into the keyboard when you try to move the cursor to the left edge of the screen, just pick the mouse up and set it down further to the right.

2. Basic features

This section describes the minimum set of things you have to know in order to do any useful work with Bravo. When you have finished this section, you can read the other parts of the manual as you need the information.

2.1 Moving around in a document

Move the cursor to the left edge of the screen and a little bit below the heavy black bar. Notice that it appears as a double-headed arrow. It will keep this shape as long as you stay near the left edge, in a region called the *scroll bar.* If you move it too far right, the shape will change. Keep the cursor in the scroll bar for the moment.

Now push down the RED (top or left) button and hold it down. Notice that the cursor changes to a heavy upward arrow. This indicates that when you let the button go, the line opposite the cursor will be moved to the top of the window. Try it. This is called *scrolling* the document up.

Next push down the BLUE (bottom or right) button and hold it down. Now the arrow points down, indicating that when you let the button go, the top line on the screen will be moved down to where the cursor is. Try it. This operation takes a few seconds, so don't get impatient. Practice scrolling the document up and down until you feel comfortable with it. It is useful to know that if you don't move the mouse, scrolling with RED and BLUE are symmetrical operations: one reverses the effect of the other.

You may have noticed that the text on the screen doesn't fill up the window, but that more text appears when you scroll up. The reason for this is that in addition to space on the screen, Bravo needs space inside itself (in the Alto's memory) to display lines of text on the screen. When a line has only a few characters, it doesn't take up much internal space, but when it runs all the way across the page, like the lines in this document, it takes a lot of internal space. When Bravo runs out of internal space, it stops displaying text and leaves the rest of the window blank. You can tell that there is more text in the document (i.e., that you aren't seeing the end), because when Bravo gets to the end it displays a triangular **endmark** as the very last thing to mark the end. If you don't see the **endmark** at the bottom of the displayed text, you can be sure that there is more text in the document which isn't being displayed.

If you keep the cursor in the scroll bar, near the left edge, and hold down YELLOW (the middle mouse button), you will see the cursor change into a striped right-pointing arrowhead. Think of it as a thumb, and the entire left edge of the window as the pages of a closed book, corresponding to your whole document (*not* just to what is displayed). If you stick the thumbnail into the book and flip it open, you will find yourself someplace in the book. If the thumb is near the middle, you will be about in the middle. If it is all the way at the top, you will be at the beginning; if all the way at the bottom, you will be at the end.

The tip of the arrowhead acts like the thumbnail, and letting go of YELLOW is like flipping open the book. You will also see another striped arrow, enclosed in a box. This one is called the *bookmark*; it points to your current location in the document. After you let up YELLOW, if you hold it down again without moving the mouse, the thumbnail and the bookmark should coincide exactly, making a solid arrowhead; this happens because the thumbing operation moved the document exactly to the place indicated by the thumbnail. To move forward a little, push the thumbnail down a little below the bookmark and thumb again; to move back, push the thumbnail up a little above the bookmark. To get to the beginning, push the thumbnail up until the arrowhead overlaps slightly the horizontal bar at the top of the window. Try thumbing your way through the document until you feel comfortable with it. Also try thumbing and then scrolling up and down.

2.2 Changing the text

In order to make a change in the text of your document, you have to:
say where you want the change made, by making a *selection;*
say what you want done, by giving a *command.*
You always make the selection first, then give the command. If you change your mind about where you want the change made, you can always make another selection. Making a selection is just like pointing with a pencil: it doesn't have any effect on the document. Only commands can change the document. You never have to worry about getting rid of a selection, since it never does any harm. If you make a selection, and then give a command that doesn't require any selection, that is perfectly all right; the needless selection will be ignored.

You make selections by pointing with the mouse and pushing one of the buttons. To try this out, move the cursor into the region of the screen where the text of the document is displayed. Notice that the cursor is displayed as an arrow which points up and slightly to the left. Point the arrow at a character (any character) in the document, and click RED. The character you pointed at should be underlined; if it is, you have just selected it. If it isn't, look nearby and see if some other character is underlined. If you find one, then that is the one Bravo thought you were pointing at. Experiment until you feel confident that you can point easily at characters.

You should note that each selection erases the previous one; there is only one selection at a time, and it is the most recent one. Also, you can make a selection at any time, except when you are in the middle of a command. Once you have started a command, you must either finish it normally, or abort it by striking DEL, before you can make another selection.

Something useful to know: if you hold RED down, you can move the cursor around and the selection will follow it. The selection won't freeze until you release RED (or move the cursor out of the text area). Try this too.

Now try a selection using YELLOW instead of RED. Notice that instead of underlining a character, Bravo now underlines a whole word. A word is defined as consecutive letters and digits, or consecutive punctuation characters. For convenience, apostrophe is counted as a letter. Also, a number containing a decimal point is a single word.

There is one more thing to learn about selecting text: how to select more than one character or one word. To do this, first select a character with RED. Then point to another character and click BLUE; Bravo will underline all the characters between the one you selected with RED and the one you pointed at with BLUE. This is called *extending* the selection. Try holding down BLUE and moving the cursor around. The selection will change continuously so that it includes the characters between the one you originally selected with RED and the one you are pointing at now. As before, when you let up the button, the selection will freeze. You can change the extension as many times as you want by using BLUE over and over; Bravo will remember the original selection you made with RED until you make another one.

Finally, try selecting a word with YELLOW and then using BLUE to extend the selection. Notice that the end of the selection will be a word also. To select the entire document, issue the Everything command.

Space, TAB and carriage return (CR) characters in the document simply appear as white space on the screen, just as they do on paper. You can, however, select them like any other characters. Try it. You will notice that not all the white space on the screen can be selected; in fact, the space on a line after a CR, and the space to the left of the left margin, cannot be selected. Bravo's handling of white space is discussed in detail in section 3.5.

Now that you know how to say where you want a change made, it's time to make a change. Select something (for instance, a word). Now type D (for Delete). The word you selected is *deleted* from the document, and the selection moves over to the character after the original selection. The rest of the text is adjusted to make up for the deleted material; if necessary some words may be brought up from the next line to fill up the one which contained the deleted material.

You can *undo* the deletion by typing U (for undo). Try it; you will see the stuff you deleted reappear, and it will be selected again, just as it was before you deleted it. Do several deletions, followed by undos, until you are sure you know what will happen. Try deleting larger pieces of text by extending your selections. Be sure not to move the selection between doing the Delete and the Undo.

Delete and Undo are *commands*. Like all Bravo commands, they are given by typing just the first letter of the command name. You can type the letter in either upper or lower case.

To add new text, select something in front of which you want the new text to go (if you want it to go at the very end of the document, you can select the **endmark**). Then type I (for Insert). You will see that a blinking caret appears just before the selection. This marks the place where the new text will go. Anything you type will appear where the caret is, and as you type each character, the caret will move over to make room for it. Try typing a few characters, and notice that the rest of the text is automatically rearranged to make room for the new stuff.

If you strike the wrong key while typing, you can erase the mistake by striking the BS key (on the right side of the keyboard). You can erase as many typed characters as you like using BS. You can also use A^c (hold down the CTRL key and type A) to erase a character; it works just like BS, and may be more convenient to type with your left hand, if your right hand is on the mouse. To erase typing on a larger scale, you can use W^c (hold down the CTRL key and type W) to erase a word and its following spaces or punctuation characters. When you have typed as much as you care to, hit ESC to finish the insert. Notice that the caret disappears, and that the inserted material is selected. You can undo the insertion with Undo. Then you can undo the undo and get the insertion back. Try it.

Sometimes it is more convenient to insert *after* a selection, rather than *before*. You can do this with the Append command (remember that you just type the A). Except for where the new material goes, Append is exactly like Insert.

If you want to change one word into another, or correct a typo, you have to delete some text and insert other text in its place. This can be done by a Delete followed by an Insert, but it is more convenient to use the Replace command, which combines these two functions into one. Replace can also be undone.

Whenever Bravo first displays the blinking caret, you can insert a *copy* of some existing text rather than typing in new text. You do this by making another selection, called a *copy selection*, instead of typing. The copy selection is made exactly like an ordinary selection, and you can even use the scroll bar to move around in the document in order to find the text you want to copy. You can distinguish a copy selection from an ordinary one by its dotted underline, which contrasts with the solid underline of an ordinary selection.

You can change your copy selection as many times as you like. When you are satisfied with it, type ESC, and a copy of the copy selection will be inserted in place of the blinking caret. You can't do anything else while you are making a copy selection, except to scroll the document.

A copy selection can be used to move text from one place to another: first copy the text, and then delete the original.

There is one more useful thing to know about insertion. If you just type an ESC for an insertion, without making a copy selection or typing anything else, a copy of the last thing you inserted or deleted will be inserted. This is called repeating or *defaulting* an insertion; it is very convenient for inserting the same thing in several places, e.g., a dollar sign in front of several numbers. It also gives you another way to move text: first delete it, and then insert it in its new place by selecting the new place and typing Insert followed by ESC.

You now know all three ways of doing an insertion: *typing* the text, *selecting* some existing text to be copied, or *defaulting* the previous insertion by simply typing ESC. These three ways of inserting text can be used whenever a Bravo command needs some text. You will see many references to "inserting text" as you read on.

Before going on to learn anything more about Bravo, you should practice the Delete, Insert, Append and Replace commands, and copy selections, until they are quite familiar.

2.3 Filing a document

Whether you use Bravo simply to read or browse through a document, or to create or change it, you will need to fetch the document from a file before starting, and to file it away again afterwards if you have changed it. This section tells you how to do these things.

To fetch a document from a file, give the Get command. You will see the blinking insertion caret appear in the heavy black bar above the window. Insert the text of the file name, usually by typing it in, and ending it with an ESC just as for any other insertion. The document will appear in the document window, and there will be a note in the system window telling you how long it is. A Get will erase the old contents of the window, if any.

To file a document away, give the Put command, and type the file name as you did for Get. If the name you want is already in the black bar, you can just type ESC to default the name. It is also possible to edit the file name in the black bar, exactly like an ordinary document. Put always files away the entire document, regardless of what the selection is; when it is done, you will see a note which tells you how long it is. Bravo turns most of the screen black while executing a Put; this makes the Put run faster. Do not be alarmed at this.

Warning: If you make some changes to your document and then attempt to Quit from Bravo without having done a Put, Bravo will warn you that the document has not been filed and will ask you whether you still wish to Quit. If you want to save the document, strike DEL to cancel the Quit command, then file the document using Put. If you want to quit without saving the document, type Yes. If you do this, you will lose any changes you have made to the document. If this does happen to you, read section 4.3 on replaying to see if you can still be saved.

If you Get a document from a file and Put it back on the same file, Bravo will save the original on a *backup* file. Normally the backup file's name will be the name of the original file followed by a "$". The backup file is sometimes useful if you discover that some of the changes you made are not to your liking after all. If you have enabled version numbers at Install time (not recommended), the backup file will be the old version of the file from which you did the Get, and Bravo will make a new version for the Put (see section 8.7 of the Alto Non-programmer's Guide for a discussion of file versions).

You can do an "unformatted Get" with the Z^C command (type Z^C instead of Get); this treats the formatting information at the end of each paragraph as ordinary text. The main use of Z^C is for patching up a file which has been damaged by hardware failure or cosmic rays. In particular, if Bravo refuses to Get the file because "End of file not in Bravo format", you can usually correct the problem by doing an unformatted Get of the file, deleting the last line or two, and Putting it back. Then Quit, restart Bravo and try again to Get the file.

2.4 Hardcopy

Printed copies of a document may be obtained using the Hardcopy command. Before using Hardcopy for the *first* time, you must tell Bravo the name of the printer you intend to use regularly. To do this, Get the file User.cm. In that document, you will find an entry that looks like this:

```
[HARDCOPY]
PREFERREDFORMAT: PRESS
PRESS: Name-of-your-Press-printer
```

Replace the words "Name-of-your-Press-printer" with the name of the printer you intend to use (every printer has a registered name such as Clover, Menlo, or Daisy). Then Put the document, Quit, and type Bravo/i^{CR} to the Alto Executive.

To print one copy of the document you are editing, simply give the Hardcopy command followed by CR. This will print the entire document, regardless of what the selection is. While doing the hardcopy, Bravo displays in the cursor a count (modulo 10) of the number of pages it has processed; hardcopy takes about 8 seconds per full page, like this one. After sending the document to the printer, Bravo will report success. If there is a problem, Bravo will leave a note in the system window. If the printer is not responding, Bravo will leave a note to that effect, and keep trying. You can abort the Hardcopy by typing DEL, as always.

The hardcopy may fail for several reasons. If there is an EFTP error, trying again will usually work. If the problem is that there is a character in your document which is in a font for which there is no printable representation, Bravo leaves one of the offending characters selected, and puts it at the top of the screen. You can try again after modifying the looks of the selected character. If you have a page with so many different fonts that it exceeds the capacity of the printer, Bravo leaves the first character of the page selected and at the top of the window. There is no remedy for this problem except to simplify the offending page. See section 4.6 for more information about fonts.

You may want more than one copy of a document. The Hardcopy command has an option called Copies, which allows you to specify the number of copies you want; you type in the number, and it will appear in the leftmost buffer in the system window, much like a file name. You must give the Copies option right after the Hardcopy command, every time you want more than one copy.

If you compare the hardcopy of your document with Bravo's display, you will see that although the text is identical, the hardcopy has more words on each line, so that the two versions look quite different. In order to see a nearly exact facsimile of the hardcopy on the screen, you can give the command

Look hardcopy (note the lower-case h)

You are now in *hardcopy mode* on the screen. Until further instructed, Bravo will represent the printed version of your document as faithfully as it can, by positioning each character on the screen within one-half screen dot (about .007 inches) of its position in the final hardcopy. The screen representation is 10% larger than the printed one. To turn off the hardcopy simulation, type

Look Hardcopy (note the upper-case H)

You can edit normally in hardcopy mode. In fact, if your document contains tables whose appearance is critical to you, it is advisable to stay in this mode, because in the normal mode text will take up much more space on the screen than it will in the final hardcopy (if you have such tables, you should also read section 3.5 on white space and tabs). In hardcopy mode it is also possible to see exactly where lines will be broken, so that you can insert hyphens by hand if necessary.

Bravo provides a number of facilities for controlling page formatting, which you can read about in section 3.6.

The Hardcopy command has options for printing on the Diablo printer, and for producing a Press file which can be combined with drawings into a larger printable document, or sent to a file server for public distribution. These are described in section 4.8.

2.5 Miscellaneous

As you edit, Bravo keeps track of the changes you make to the document. In doing this, Bravo consumes space in the Alto memory. During a long editing session, it is possible to consume all the available space, in which case Bravo will leave a warning note ("Core storage getting low") in the system window, and will refuse to execute any more editing commands. If this happens, you should Put your document onto a file immediately, and then Quit, restart Bravo, and Get the document back from the file. Now you can continue with another editing session.

When you have finished editing one document and have filed it away, you can Get another file, and continue working. It you are making extensive changes, however, it is better to Quit and restart Bravo when you start to work on a new document. If you do this, you are less likely to provoke a bug in Bravo, and you will be able to recover from a crash with the replay feature (section 4.3) much more quickly.

The maximum size of a Bravo document is 65,536 characters. Whenever Bravo Gets or Puts a document, it leaves a note telling you how long the document was. When your document has reached 65,536 characters, you won't be able to add any more text, and peculiar things may occur if you do try to add more text. It is a good idea to split the document into two parts well before this happens. To encourage you to do this, Bravo will flash the screen and display a warning message after every command if the length of the document exceeds 60,000 characters.

If you type a character which has no printable representation, Bravo will display it as a black rectangle. The best thing to do with such a character is to delete it.

Depending on exactly what Bravo is doing, the amount of text it can display on the screen will vary. You can always get the maximum amount of text displayed by doing a scrolling operation; if you scroll up with the cursor at the top of the scroll bar, the text won't move, and Bravo will just display as much more as it can. If you then give a command, some of the text may disappear from the screen, but you can always get it to reappear by doing another scrolling operation.

Bravo keeps copies within itself of information in your user profile (file User.cm; see section 4.6) and in various files on your disk: font files (named *.al and Fonts.Widths), and the files containing the Bravo system and its temporary storage (named Bravo.*). It refreshes these copies whenever you start it up with

>Bravo/i^CR

This is called *initializing* Bravo. It is necessary to initialize whenever you get a new version of Bravo or the Alto Operating System and when you change your user profile or any font file. Initializing is just like starting Bravo up normally, except that it takes about a minute. *If you are in any doubt* about whether something has changed since the last time you initialized Bravo, or if your Bravo is crashing with messages which refer to disk or file errors, *you should initialize Bravo* by starting it with Bravo/i.

You now know enough to edit unformatted documents. Take a rest.

3. Formatting

This section describes the Bravo facilities for creating formatted text and pages. It you are not interested in formatting, you don't have to read it. If you are interested, be sure to read sections 3.3 and 3.4, where you will find a lot of good advice.

Bravo normally describes character sizes and distances on the page in *points*. A point is a unit of distance used in the printing industry; there are 72 points per inch. Thus 36 points is 1/2 inch, and 18 points is 1/4 inch. In many cases, you can also specify distances in inches or centimeters, as described in section 3.2.

3.1 Making pretty characters

Bravo allows you to say how you want your text printed: in italics or bold face, underlined, in various sizes and type styles, superscripted or subscripted, etc. You can change the way existing text is printed, or you can say how you want the characters to appear as you are typing them in. We will begin by describing how to change the looks of existing text.

First, select the text you want to mess with. Then give the Look command. This command has a large number of options, each specified by a single character, which is sometimes followed by some additional information:

bold	SHIFT B to un-bold
italic	SHIFT I to un-italicize
– to underline	SHIFT – to remove the underline
← to subscript (text down 4 pts) SHIFT ← or ↑ to superscript (up 4 pts)	Down 0 ESC to remove subscript or superscript
0 to 9 to set the typeface	
visible to display spaces, tabs, and CRs	SHIFT V to stop this.
Down followed by a distance (see below) to move the text down that distance, relative to the baseline. Subscript is Down 4.	Up followed by a distance to move the text up. Superscript is Up 4.

CLR (the blank key to the right of BS on an Alto-I, or the key labelled BW on an Alto-II) to restore the standard looks: font 0; not bold, italic, underlined, visible, graphic, up, or down.

The typeface is usually called the *font*. For Bravo, each different size of the same style is a different font, but bold and italic are considered to be in the same font. The choice of fonts is specified by your *user profile* in a way which is described later (in section 4.6), but the standard choice provided on the basic non-programmer's disk is:

0 Times Roman, 10 pt. This is the standard font.
1 Times Roman, 8 pt.
2 XEROX logo (only the capital letters E O R and X)
3 Math, 10 pt. A large set of mathematical symbols. No bold or italics on hardcopy.
4 Greek, 10 pt. No bold or italics on hardcopy.
5 Times Roman, 12 pt.
6 Helvetica, 10 pt.
7 Helvetica, 8 pt.
8 Gacha, 10 pt. This is a fixed-pitch font.

9 Helvetica, 18 pt. The bold-face version of this font is especially good for making view-graphs.

You will find tables at the end of this manual which give the correspondence between ordinary characters and the Math and Greek fonts, and some samples of the various fonts.

There is another Look option which is very convenient. It is Look Same, followed by a copy selection. In this case, what is copied is the looks, rather than the characters. This is the way to get one piece of text to print in the same style as another piece.

Like most commands, Look can be repeated with ESC. This is useful if you want to change the looks of several pieces of text in the same way. You can also undo a Look with Undo.

You can find out what the looks of a character are by selecting it and giving the Look ? command. Bravo will tell you (in the system window) all the looks of the selected character. You may have to scroll the system window up in order to see all the looks.

LOOKS DURING TYPING

When you start typing, the looks which will be attached to the characters you type are set to the looks of

the first character of the selection if the command is Insert or Replace;
the last character of the selection if the command is Append;
the standard looks otherwise.

To change the looks while you are typing text, use the CTRL key instead of the Look command: hold down CTRL and type the look you want. The only things described above which you can't do this way are Look Up and Down; you can get the standard superscript and substript offsets with ↑ and ←, though. To restore the standard looks, you can just strike the CLR key; it is not necessary to use CTRL in this case.

3.2 Paragraphs

In addition to changing the looks of the characters, you can also change the shape of the text: the margins, space between lines, justification, centering, etc. The Bravo facilities for doing this are based on the idea of a *paragraph.*

A paragraph in Bravo is all the text between two CTRL-CR characters. You can tell when you have one by selecting it. To do this, move the cursor into the *line bar*, which is between the scroll bar on the far left, and the text area. You can tell that you are in the line bar, because the cursor will appear as a rightward-pointing arrow. Once you are in the line bar, use the YELLOW button to select a paragraph. Note that the cursor changes to a paragraph symbol; it keeps this shape as long as the selection is a paragraph.

The CTRL-CR that ends a paragraph carries the paragraph looks described below. It can also carry character looks, and if you are setting up a standard paragraph, it is a good idea to attach to its CTRL-CR the character looks which you want as the standard ones for the paragraph. Thus, for example, the CTRL-CR for a standard heading like the one at the start of this section would carry the italic look. Of course, this is just a convenience, and not essential; you can always set the character looks during typein as described above, e.g. by typing ic for italics.

If the text at the end of a paragraph is in a font smaller than the standard one, as this one is, the CTRL-CR ending the paragraph should carry the same font looks. Otherwise, the inter-line spacing of the paragraph may appear uneven.

The YELLOW button selects exactly *one* paragraph, so by looking at what is underlined you can tell where the paragraph starts and ends. Note that the second CTRL-CR (the one that ends the paragraph) is counted as part of the paragraph; the first CTRL-CR is part of the previous paragraph. You can use BLUE to extend the selection to several paragraphs.

To merge two paragraphs into one, just delete the CTRL-CR that separates them. You will probably want to replace it with a couple of spaces, or maybe with an ordinary CR. To break one paragraph into two, insert a CTRL-CR; it is just like any other character, except that you can't backspace over it.

If you select a paragraph and then give an Append, Insert or Replace command, a blank paragraph with the same looks as the selected one will be created for you to type into.

To change the looks of a paragraph, you can use some more sub-cases of the Look command. Select the paragraph (or any text in it) first, and then say Look, followed by:

center; turns off justification	SHIFT C to stop centering
justify (even right margin); turns off centering	SHIFT J to stop justifying
nested to indent the whole paragraph (36 pts, or 1/2 inch, more)	SHIFT N to un-indent
open up more white space in front of the paragraph (12 pts, or 1/6 inch, more)	SHIFT O to close up the white space
q to open up half as much more white space in front of the paragraph as Open does (6 pts more)	SHIFT Q to close up the white space

All of these can be invoked during type-in; hold down the CTRL key and strike the appropriate key, just as you do for character looks.

In the following Look cases, *d* is a *distance* on the page, which can be specified in several different ways, as described below. Distances are measured from the left edge of the paper (except for Up and Down, which measure from the baseline of the line of text). These looks cannot be used during type-in.

Left *d* to set the left margin. The default is 85 points, or about 1.2 inches from the left edge of the paper.

First *d* to set the left margin of the first line. Use this to control indenting or un-indenting of the first line. A Look Left cancels a Look First, since it sets the left margin for all the lines of the paragraph.

Paragraph *d* to set the left margin of all the lines except the first. A Look Left cancels a Look Paragraph, since it sets the left margin for all the lines of the paragraph.

Right *d* to set the right margin. The default is 527 points. Since an 8.5" x 11" page is 612 points wide, this results in 85 points, or 1.18", of white space on the right. Thus, the default margins center the text on the page.

X *n* to set the space or *leading* between lines. The leading should be at least 1 point (as it is in this document) to avoid a squashed effect. If you want a less dense appearance, try larger leadings. The default is 6pt, which gives double spaced text.

Y *n* to set the leading in front of the paragraph. The default is 12pt, which gives a blank line between paragraphs. Look open increases the paragraph leading by 12 pts, and Look q increases it by half that, or 6 pts. On hardcopy, both line and paragraph leading are suppressed for the first line of a page or column. Leading must be less than 64 points.

Here are the ways to specify the distance. Try them out until you are quite comfortable with them.

By typing a number in one of the following forms:

123 or 123pt	a distance in *points*. A point is a printer's unit equal to 1/72 of an inch. A number without a decimal point and with no explicit units is assumed to be in points.
1.71, 1.71in or 1.71"	a distance in inches. A number with a decimal point and no explicit units is assumed to be in inches.
4.34cm	a distance in centimeters.
a sequence of blanks	a distance equal to the width of that many blanks.

By typing a number *n*, as above, preceded by + or −. The distance specified by *n* is added to, or subtracted from, the current value of the look being changed. Thus, to indent a paragraph by an additional one-half inch, type Look Left +.5 ESC.

By using BLUE to point to a place on the screen. The horizontal position of the place you point at is displayed in the system window. If you hold down BLUE and move around, the displayed position is updated continuously.

By using RED to select a character. The horizontal position of the left edge of the character is displayed in the system window.

By typing \ (back-slash, not /), which displays a default value for the look being changed in the system window.

By just typing ESC, which uses the value already in the leftmost buffer of the system window.

You can select, point or default as many times as you want, just as with an ordinary copy selection. Then you can type a number, if you like. When the leftmost buffer in the system window has the value you want, type ESC to complete the command. Of course, if you get disgusted you can always type DEL to cancel the whole thing. Note that pointing is a convenient way to measure horizontal positions on the page.

Look All, followed by a copy selection, will copy all the paragraph looks of the paragraph in which the copy selection is made, to the paragraph containing the current selection. Note that Look All copies paragraph looks whereas Look Same copies character looks.

If a paragraph is selected (using YELLOW in the line bar; the cursor will be a paragraph symbol when a paragraph is selected), the Look ? command will display the paragraph looks in the system window; if the selection is not a paragraph, the command displays character looks, as described in the previous section. You may have to scroll the system window to see all the information. Note that it appears in a buffer (see section 4.5) which is made current, and you can insert it into a document with a default insertion.

If you have a paragraph whose left margin is less than the default (normally 85 pts), any characters in the paragraph to the left of the default margin will fall off the left edge of the window and will not be displayed. Try setting a left margin to some values less than 85, and see how this works. You can change the setting of the left edge of the window, so as to make these characters visible on the screen, with the command

Window Edge *d* *d* is a distance, which must be typed in and cannot be obtained by pointing.

The distance *d* is the distance from the left edge of the page at which the left edge of the window should be set. It should be smaller than any paragraph left margin if you want to see all the characters on the left. For instance, if *d* is 0, the window edge will be at the paper edge; if the text has the usual 85 pt margin, this will result in 1.2" of white space in the window (in addition, of course, to the white space in the line and scroll bars). The default value for *d* is the default left margin.

HINTS

You can select several paragraphs (using BLUE to extend your selection) and apply the same Look command to all of them. You can change the looks of every paragraph in the document by doing an Everything to select the whole document before the Look. A Look command involving a distance of the form $+n$ or $-n$ adds or subtracts n from the look value for each selected paragraph. Thus, Look Left +5 ESC will indent each selected paragraph by five more points.

If you use several different formats (e.g., for section headings or for indented material) you can copy the formatting from an existing example of a particular style to a newly created one with Look All. Often it is convenient to put a set of sample paragraphs at the head of your document, each containing one line which explains what it is a sample of. Then you can split the window (as described in section 4.2) and have the samples readily available to copy from with Look All. This is highly preferable to entering all the new looks manually every time you switch to a new format.

An alternative technique for creating a new paragraph in a specific style is to select the paragraph before or after the point at which the new paragraph is to appear, then Append or Insert and make a copy selection of the desired sample paragraph. Now select the text of the newly-created paragraph, *not including the CTRL-CR at the end*, and Replace it with new text that you type in. This method copies both character looks and paragraph looks from the sample paragraph.

When you are setting up the format for a document, you should put a few blank paragraphs (just CTRL-CRs) at the end, and set the formatting on all of them to your standard format (it is convenient to do this by copying the formatting from a paragraph which already has your standard format). This might include indenting the first line of a paragraph, setting the leading, leaving space between paragraphs, justification, and even the font. Now when you add material to the document by inserting into one of these blank paragraphs, you will automatically pick up all of the formatting you have preset. As you type along, each time you use a CTRL-CR to start a new paragraph, it will acquire the same formatting as the old one.

3.3 Formatting style

This section is intended to provide you with some guidance in using all the different ways Bravo gives you for controlling the appearance of your document. Many of the rules are based on the customs of the printing industry. There are two advantages to following these customs:

they are the result of many years of experimentation, during which many people have tried to find out what looks good on the page;

readers are accustomed to seeing text presented in this style.

You will notice that some of the rules are contrary to the usual practice for preparing documents on a typewriter. There are good reasons for this: when you are printing with variable-pitch fonts, italics, boldface, justification, and precisely controlled leading, some of the things which work well for fixed-pitch, single-font documents are no longer appropriate.

EMPHASIS

Use italics for *emphasis* in text. You can also use boldface, but this is usually less desirable, and it is better to reserve boldface for words which play some special role, e.g., **begin** and **end** in computer programs. You should also use italics for the names of variables, e.g., "Suppose there are *n* items."

Don't use underlining for emphasis; it is not compatible with the use of italics and boldface. Use underlining only when you want a different *kind* of emphasis, e.g., to distinguish the characters a user types from the ones the machine types, as is done in this document.

Don't capitalize a whole word for emphasis. In fact, try not to capitalize a whole word at all; it usually looks terrible in a variable-pitch font because the capital letters are so much wider than the small ones. If you have words which you think should be set in capitals for some reason, try SMALL CAPITALS. In this example, the S and C were 10 point (font 0), the rest of the letters 8 point (font 1). Compare this with the appearance of FULLY CAPITALIZED words and you will see the point.

SECTION HEADINGS

In general, use left-justified rather than centered headings, and don't use all capitals, for the reasons just discussed. Here is a satisfactory list of styles for the headings of successively larger portions of your document:

smallest	*Italic*	18 pt paragraph leading (Look Y 18, or Look q if your standard leading is 12 pts).
next	**Bold**	24 pt paragraph leading (Look Y 24, or Look o if your standard leading is 12 pts).
largest	**12 pt bold**	36 pt paragraph leading (Look Y 36, or Look o twice if your standard leading is 12 pts).

Note that you can switch from the standard leading to the 1.5, 2 or 3 times standard leadings for headings during typein, using o^c and q^c. For the largest units, you can center the heading and/or use all caps instead of, or as well as, switching to a 12 pt font. It is best not to have more than three levels of heading, but you can extend to four or five levels using these tricks. Helvetica 18 bold (font 9 bold) is sometimes nice for chapter or document titles.

Use Look Keep 80 (see section 3.6) on headings to make sure that the heading doesn't end up all by itself at the bottom of a page.

LEADING

The standard printing fonts are designed in such a way that they need some extra space between the lines to avoid a cramped appearance. You put this space in with Look X, and you should use 1 pt for ordinary single-spaced text. If you want a less dense appearance, experiment with more leading. For double-spacing of the text, try Look X 6 (the default).

Use double spacing (Look o) between paragraphs. When you have indented material which is fairly short, try 6 pt leading (Look q), as in the example two paragraphs back. *Don't* use extra carriage returns to get blank space between paragraphs. However, the maximum leading you can specify is 63 points; if you need more (e.g., to leave space for a figure) you will have to put in blank paragraphs.

Note that both line and paragraph leading are suppressed for the first line of a page or column. The height of a line of text (in points) is equal to the point size of the largest font used in the line, provided there are no characters which have been superscripted, subscripted or offset with Look Up or Look Down. If any character in the line is offset Up, the minimum line height, *including* leading, is given by the font size of the character, plus its offset; i.e., characters offset Up are allowed to eat into the leading. If a character is offset Down, the largest such offset must be added to obtain the line height; i.e., characters offset Down are *not* allowed to eat into the leading.

INDENTING

Use Look nested to indent material, and Look Nested to cancel the indentation. Note that this also works when you are typing in. For example, if you type

CR^c n^c Here are three points: CR^c n^c First ..CR^c Second ..CR^c Third ..CR^c N^c Now we continue ...

the document will look like this:

Here are three points:
    First ..
    Second ..
    Third ..
Now we continue ..

Use Look First if you want to indent the first line of a paragraph, rather than tabs. When you have a list of items, it is often nice to *un*indent the first line by about 15 pts, especially if the items are numbered. For example:

1. This paragraph was formatted with Look Left 120, Look First −15, in order to make the number hang out to the left.
2. To get the first word of the first line to line up with the left margin on subsequent lines, set a tab stop at that point (see section 3.5).
3. The easiest way to specify the position of the tab stop is to select the first character of the second line, using RED. In this case, of course, the stop is at 120.
4. Indended paragraphs sometimes look better balanced if the right margin is indented as well. Unfortunately, Look nested doesn't do this for you; you have to change the right margin yourself using Look Right *d*.

OFFSETS UP AND DOWN

Use the smallest offset you can get away with for subscripts and superscripts, since large offsets result in wide ugly spaces between the lines. The offset used by Look ↑ (superscript) and Look ← (subscript) are defined in your user profile (see section 4.6); the standard profile sets it to 4 pts.

3.4 Forms

Although Bravo will let you begin with a completely empty window and start typing into it, this is a bad practice and should be avoided. Instead, you should start out by Getting a *template* or *form* which will guide you in constructing the document you want.

An obvious example is a memo form, and you will find one on the file Form.Memo. Start Bravo, and Get Form.Memo into the window. You will see that it has spaces for the sender, receiver, date and subject, and that these are filled in with words which indicate what should go there. To fill in the form, select each of these words, and Replace it with the proper text. Then do the same with the MEMOBODY. When you are done, you have a completed memo which you can file under a suitable name using Put. Be careful not to Put the document back onto the file from which the form came. The best way to avoid doing this is to edit the file name in the black bar above the document immediately after Getting the form.

Your disk comes equipped with a few forms; you can see their names by typing form. TAB to the Executive. You should construct your own forms for other kinds of documents which you find yourself creating frequently. As you have seen in the description of Bravo's formatting features above, a form can contain a great deal of information in addition to standard text and spaces to be filled in. You will find that your life is easier and your work is more uniform and of higher quality if you use forms consistently, and take the trouble to carefully design a new one when necessary.

3.5 White space and tabs

When you type on a typewriter, you can get white space to appear between characters by typing spaces or TABs. You can get blank lines by typing carriage returns. In Bravo, you can do exactly the same things, with exactly the same results. Space, TAB and CR are characters which are in your document exactly like "a", "b" or "c". You can get Bravo to display them as special, visible characters by selecting the text in which you want to see them, and typing

Look visible (this must be a lower-case v).

To turn off the display and just see the usual white space, type

Look Visible (this must be an upper-case V).

Normally you don't have to type any CRs; Bravo will automatically end a line when there is no room for the next word. You can force a line to end by putting in a CR; this is appropriate when you want to control the layout of the text precisely, as in a table. Otherwise, don't put in CRs. You should use CTRL-CR to end a paragraph, as described in section 3.2.

Bravo allows you to set up to 14 tab stops, which are named by the digits 1-9 and the letters abcde. The tab stops are paragraph looks, just like the margins; hence they can be different for each paragraph. You can set a tab stop with the command

Look TAB *t d*

where *t* is a digit or one of the letters abcde, and *d* is a distance (see section 3.2).

When you strike the TAB key during typein, the caret moves to the next tab stop, just as it does on a typewriter, and a TAB character is added to the document. This TAB character is called a *plain-tab*, because it moves the caret to the next tab stop, not to a specific named tab stop.

For example, suppose you have a line like this:

Column 1 Column 2 Column 3

Tab stops 1, 2, and 3 are at 180, 265 and 400 points, and there is a plain-tab between each digit and the following C. If you now append some x's to the digit 1 to get past tab stop 1, the result will look like this:

Column 1xxxxxxx Column 2 Column 3

That is, the point to which a plain-tab jumps depends on the width of the preceding text. This can vary both when you change the text and when you switch to hardcopy mode; thus, the appearance of hardcopy may not match the screen if you are using plain-tabs.

You can turn a plain-tab into a *named-tab* by selecting it and issuing the command Look , *t* (Look comma *t*), where *t* is the name of a tab stop. A named TAB character will always make the following character print at the correspondingly named tab stop. If printing has already passed that tab stop, Bravo will start a new line, and display a heavy black rectangle at the end of the previous line, to warn you that something is wrong.

To continue the above example, suppose you name the first TAB 1 and the second 2. Now the result will look like this:

Column 1xxxxxxx▮
Column 2 Column 3

When you switch from normal display mode to hardcopy mode, there will usually be more white space occupied by the TAB (perhaps enough to permit printing all the text on one line), but everything will continue to be positioned horizontally in exactly the same way.

You can find out the name of a tab stop by selecting it and giving the Look ? command.

Caution: the Look comma command should be applied only to TAB characters. If applied to a character other than TAB, it will invoke some unsupported features for color printing.

For compatibility with old versions of Bravo, and with the programmer-oriented tab conventions of the Alto and Maxc, you can set unnamed or *plain* tab stops spaced at equal intervals with the command

Look TAB = *d*

where the distance *d* specifies the interval. If you don't set any tab stops, you get plain tab stops spaced at 36 pts (this parameter comes from the user profile, and can be changed; see section 4.6).

One final word about white space: Bravo has formatting features, described in the section on paragraphs above, which allow you to indent the first line of a paragraph, and to put blank space above a paragraph, without using spaces, TABs or extra CRs. It is good practice to use these features, since you can control the spacing much more precisely and don't have to worry about having extra characters cluttering up your document.

3.6 Page formatting

There are a number of features to help you in controlling the layout of your document on printed pages. Unlike the horizontal layout, the location of page breaks and the headings, page numbers etc. for the most part cannot be displayed on the screen. There is, however, a *page boundary* command which allows you to see on the screen where the page boundaries will appear in the hardcopy. The command is invoked by the LF key. It assumes that the first character of the current selection is the first character on a hardcopy page, and it moves the selection to the first line of the next page. By applying the page boundary command repeatedly, you can move through the document, page by page (or column by column, if your document profile specifies multiple columns; see below). Alternatively, if you know where one page break is (perhaps because of a control-L in the previous line; see below), you can start there. If you want to start at the beginning of the document, you can use the Everything command to make the first character of the document be the first character of the selection.

As a convenience, the page boundary command leaves the original selection at the top of one subwindow, and the first line of the next page as the third line of the next subwindow (which it creates if necessary). Among other things, this makes it easy to do some editing near the end of the page, and then reselect the beginning of the page and repeat the command. Try it.

Normally, Bravo will start a new page when it runs out of room on the current page, i.e., when the next line to be printed would intrude on the bottom margin, or at the beginning of a paragraph if the amount of space left before the botom margin is less than the paragraph's keep value. You can force a page break by including a L^c (control-L) in the text; the line containing the L^c will stay on the same page, but the next line will start a new page. This character is displayed as a lower-case L with an over-bar. You can't type it in simply by holding down CTRL and typing L, but instead you can type an L followed by S^c. You do this *during an insertion*, not as a command. The L^c is treated just like any other character during editing.

You can also force a *paragraph* to start a new page by giving it a keep property of 11". If you want to position the paragraph precisely on the new page, give it a vertical tab property as well.

You can exercise some control over where page breaks occur with the command

Look Keep *d* *d* is a distance

This sets the paragraph property called *keep*, which has the following meaning. During hardcopy,

when printing of the paragraph is begun, the amount of space left on the page before the bottom margin must be at least the keep distance, or a new page will be started. For instance, by setting the keep of a heading to the total height of the heading (including its leading) plus the height of the first two lines of the next paragraph, plus the paragraph leading, you can ensure that the heading will never end up alone at the bottom of a page. Good values to use, with standard fonts and leading as in this document, are 40 pts on ordinary paragraphs and 80 pts on headings.

You can set the vertical position of a paragraph precisely on a page using the vertical tab property, which is set by the command

Look Z *d* *d* is a distance.

When a paragraph with a vertical tab is printed, its upper edge (including leading, if any) will be positioned at the vertical tab value, measured from the *bottom* of the page (i.e., use 10.5" to put it .5" from the top). Unlike a horizontal tab, which may start a new line, a vertical tab never starts a new page; instead, it may cause overprinting. Vertical tabs are useful for positioning headings and footnotes, and for precisely aligning text to meet some physical constraint, such as a pre-printed form or a window envelope. The first line of a paragraph with a vertical tab will be printed on the current page, even if it runs into the bottom margin (but not if the paragraph also has a keep property which forces it off the page).

Vertical tab and keep properties are not visible on the screen, but you can always use Look ? to find out whether a paragraph has them, and what their values are.

Note that both line and paragraph leading are suppressed for the first line of a page of column. If you want white space in front of such a line *l*, you can use vertical tabs, or introduce a blank line in front of line *l*, and adjust the leading of *l* to compensate for the height of the blank line.

The remaining aspects of page formatting can be controlled by an optional *document profile* which you can put at the very beginning of the document. The document profile is a sequence of paragraphs, each of which must have the *profile* property. This property is set and cleared by a Look command:

Look ; sets the profile property Look SHIFT ; clears it

A document profile has the following form (this one is the profile for this part of this manual):

Page Numbers: Yes X: 527 Y: −.5" First Page: 40 Not-on-first-page
Private Data Stamp: No X: 3.5" Y: −.6"
Columns: 1 Edge Margin: .6" Between Columns: .4"
Margins: Top: 1.3" Bottom: 1" Binding: −5
Line Numbers: No Modulus: 5 Page-relative First Line: 1
Odd Heading: Not-on-first-page
BRAVO MANUAL
Even Heading:
Section 3: Formatting

Any of the lines may be omitted, and in general any of the fields on a line may be omitted. Fields on a line are separated by one or more spaces. Distances, shown in inches in the example, may be given in points or centimeters, as described in section 3.2. X coordinates are from the left edge of the paper, Y coordinates from the bottom; negative coordinates are measured from the right edge or top of an 8.5" x 11" page. Bravo's measurements on the page are exact to less than .01". Actual printers, however, can make errors in positioning the text on the page of as much as .25" in any direction. These errors do not affect the *relative* positions of characters (e.g., the length of a line cannot be affected) but they can cause the text to shift around on the page as a whole.

We now proceed to explain the various options.

PAGE NUMBERS

The coordinates of the page number are the coordinates of the upper right corner of the number. You can add Roman to the line if you want Roman numerals for your page numbers, and Uppercase if you want the Roman numerals in upper case. If Not-on-first-page is present, the page number is not printed on the first page of the document. If First Page is not specified, it is assumed to be 1, and Not-on-first-page is also assumed for both page numbers and heading; i.e., there will be no page number or heading on page 1 in the default case.

Each page number's looks are copied from the first character of the heading printed on the same page, if there is one, or from the first character of the document otherwise.

The coordinates of the private data stamp are for its upper right corner. Do not use a private data stamp without proper authorization. You will need to supply a password on each hardcopy to get the private data stamp applied; see your laboratory manager to learn the password if you have a legitimate need.

MARGINS

The top margin specifies the amount of white space at the top of the page. The bottom margin specifies the minimum amount of white space at the bottom of the page; a line will start a new page if any part of it intrudes into the bottom margin. Exception: if a paragraph has a vertical tab, its first line will be printed without regard to the bottom margin, and it may be positioned without any regard to the top margin.

If Binding appears, it is assumed that the pages are eventually to be printed on both sides of the paper, with odd-numbered pages on the right side of the resulting double spreads. Page numbers of even pages will be reflected left-to-right; in the example, even page numbers will have their upper *left* corner at X: .5" Y: .5". The binding distance is the amount of extra margin to be supplied on the inner side of the page, which abuts the binding. This amount is added to all the X coordinates on odd pages and subtracted from all the X coordinates on even pages. For example, if you want 98 pt (1.36") outside margins and 72 pt (1") inside margins, use a left margin (Look Left) of 85 pt (the default), and a right margin (Look Right) of 612 (8.5")−85=527 (also the default) to center the text on the page. Then use a Binding of 72−85=−13. In general, the rule is:

Let *d* = (desired outside margin + desired inside margin)/2
Look Left *d*
Look Right 612 (8.5") − *d*
Binding: desired inside margin − *d*.

This rule will lead to a negative binding if the inside margin is less than the outside margin; that is perfectly all right.

MULTIPLE-COLUMN PRINTING

The columns line is relevant only for multiple-column pages. It says that the hardcopy should have the specified number of columns, with the nominal edge margin (at both edges) specified (.6" in the example), and the amount of space between columns specified (.4" in the example). If the number of columns in the example is changed to 2, the nominal horizontal layout of an odd page will be:

.6" edge margin; 3.45" text, column 1; .4" between columns; 3.45" text, column 2; .6" edge margin

for a total of 8.5". The text is centered on the page; if a Binding is specified, the text will be displaced in opposite directions on odd and even pages, just as for single-column text. The width of the text in the columns (3.45 in this example) is determined by subtracting all the other space from the 8.5" page width. If there are *nc* columns, the column width is

col width = (8.5" − 2*(edge margin) − (*nc*−1)*(between cols))/*nc*

The text width and position specified above is only nominal: the actual width and position is determined by the specified left and right margins in the following way. The first column is printed exactly as its left and right margins specify. The second column is moved right by (col width + between cols) from what its left margin specifies (i.e., that amount is added to all its X coordinates). This means, for example, that you can get a double-column page with some text at the top which runs all the way across by setting the right margin of the full-width text appropriately, and using a vertical tab to position the first paragraph of the second column below the full-width text. The appearance of the resulting page will be

Full-width text .....
first-col text second-col text

Note that to do this you must manually find the end of the first column (easily done using the page boundary command), and put a suitable vertical tab property on the first paragraph of the second column.

A consequence of this laissez-faire approach to column formatting is that you must supply the proper left and right margins yourself. To keep the text within the nominal boundaries defined above, the left margin should be greater than or equal to the edge margin specified in the document profile, and the right margin should be less than or equal to the edge margin plus the column width.

The edge margin specified in the example, which would be much too small for single-column pages, is good for double-column. It is also desirable to reduce the top and bottom margins when you are printing double-column, e.g., to .8" and .4" respectively.

When you are printing more than one column, a L^c in the document starts a new *column* rather than a new page. To start a new page, use two consecutive L^c characters.

LINE NUMBERS

If there is a line which says

Line Numbers: Yes Modulus: *n* Page-relative First Line: *f*

every *n*th line will be numbered, slightly to the left of the standard left margin. Thus, if *n* is 5, the numbers will be 5, 10, 15 .... If Page-relative appears, numbering starts over on each page; otherwise it continues throughout the document. If First Line appears, the first line is numbered *f*, and numbering continues from there; otherwise the first line is numbered 1.

HEADINGS

If a Heading line appears, it must be followed by a paragraph, also with the profile property, which is used as the heading on each page. This paragraph should have a vertical tab which positions it correctly (for example, 10.5" for the heading on this page) and appropriate margins, centering or whatever to produce the desired effects. It may have more than one line. It is also possible to have separate Odd Heading and Even Heading paragraphs. If Not-on-first-page is present, the heading will not be printed on the first page.

4. Other things

In this section you can learn about a wide variety of other useful things Bravo can do. They are described more-or-less in order of cost-effectiveness: the earlier ones will probably give you more payoff per unit of effort to learn them.

4.1 Some useful features

This section describes a number of features which are easy to learn and helpful to use. As always, it is a good idea to try them out as you read about them.

You can select entire lines of the document by moving the cursor into the *line bar*, which is to the left of the text area and to the right of the scroll bar. You can tell that you are in the line bar because the cursor will appear as a right-pointing arrow when it is in the line bar. To select the entire line pointed to by the cursor, use the RED mouse button. To extend the selection, use the BLUE button. Both of these work very much like selecting a character and extending. The YELLOW button selects a *paragraph*; you can read about paragraphs in section 3.2.

To put the current selection at the top of the screen, say Normalize.

To insert the current *date and time* in front of the current selection, say Time. Notice that it leaves just the time selected, so if you follow it immediately with a Delete, you will be left with just the date. To replace some text with the current date, select it and say Delete Time Delete; be sure you understand why this works.

You can *search* the document for the next occurrence of some text with the Jump command. After you say Jump, you have to specify the text you want to search for, and you do this exactly the way you make an insertion: by typing it in (ending with ESC), by making a copy selection, or by typing ESC to default to the same text which was used for the last Jump (*not* the last insertion or deletion). Notice that if you type in text, it appears between the right-most set of curly brackets in the system window; this is called the *search key buffer*, and it normally contains the last text you searched for. However, the contents of this buffer are destroyed by the Look ? command and some of the Calculator commands. The search starts with the second line displayed in the window. If it succeeds, it brings the first occurrence of the text to the top of the window; if it fails, a note in the system window informs you. Jump does not affect the current selection. The search ignores the looks of the characters.

You can *substitute* one text for another using the Substitute command. It will ask you (in the top window) for the information it needs. In looking for substitutions it will examine only the text in the current selection, so if you want to substitute throughout the document, do an Everything first; this will make the entire document the current selection. For reasons you don't want to know about, it is not a good idea to do a Substitute in which the old text contains a CR.

Most Bravo commands can be *repeated* by simply typing ESC in command mode. When you do this, Bravo uses the *current* selection, not the one which the previous command used. For example, you can append a carriage return after each of several words by selecting the first one and Appending after it, and then selecting successive words and simply typing ESC. Or, you can search through the document looking for occurrences of a word by Jumping to it once and then just typing ESC.

The Undo command will *undo* the action of most Bravo commands which change the document, provided you haven't moved the selection. You can only Undo the most recent command; it will still work if you have scrolled, however.

The (command will put parentheses around the current selection. You can put other kinds of brackets around the current selection with the commands [, {, <, ' and ".

The) command expands the current selection as little as possible to make it balanced with respect to parentheses. This is useful primarily while editing algebraic expressions or programs. For example, if the current selection is the underlined character in

$$X_{n+1} = ((aX_n) \bmod (m+1) + c) \bmod m$$

then one application of the) command will extend the selection thus:

$$X_{n+1} = ((aX_n) \bmod \underline{(m+1)} + c) \bmod m$$

and another application will do this:

$$X_{n+1} = \underline{((aX_n) \bmod (m+1) + c)} \bmod m$$

Again, the], }, and > commands do similar things.

4.2 Windows

So far you have worked with a single document in a single window. Bravo will let you work on several documents at the same time, each in its own window. This is convenient if you want to compare two documents, or copy something from one into another, say from an address list into a letter. You can also have several subwindows looking into the same document, which is nice when you want to copy something from one part of the document to another, or to check something on another page without losing your place.

At the top of each window, separating it from the one above, is a heavy black bar. Inside this bar is the name of the file for the document in the window; this name is set by Get and used by Put. It can be edited like any other text. Subwindows, created only by the split operation described below, are separated by horizontal black lines. All the subwindows of a window are looking at the same document, although usually at different parts. If part of the document happens to be displayed in several subwindows, any changes to it will appear in all of them, and so will the selection underline or the insertion caret. Two different windows, on the other hand, are always looking at different documents, and no change to one window can affect the other. You can copy text freely from one document to another with a copy selection.

Some commands, like Jump, Everything, Get, Put and Hardcopy, work on the *current* window, which is the window containing the current selection.

There are two commands for windows, one for creating and re-arranging windows, and the other for destroying them. Each has three options, selected by the three mouse buttons.

To *create* a new window, type Window, move the cursor so that it marks the point where you want the new window boundary to be, and hold down BLUE. The new window will appear. As long as you keep BLUE down, you can move the cursor around and the top boundary of the new window will follow it. When you let go of the button, the boundary will freeze. Try it. The new window will be empty, but you can insert or Get into it.

To *split* a window and create a new subwindow, type Window, put the cursor where you want the split, and hold down YELLOW. The new boundary will appear, and it will follow the cursor until you release YELLOW. It is important to understand that after a split you have the *same* document in each subwindow. Scroll one of the subwindows so that some of the same text appears in both subwindows, and select some of the common text. Notice that the selection appears in both subwindows. If you make changes to the document, you will see them in both subwindows. This is very different from creating a new window and Getting the same file into it; that is equivalent to taking another copy out of a file cabinet.

To *move* a window or subwindow boundary, type Window, put the cursor over the boundary you want to move, and hold down RED. The boundary will follow the cursor until you let go of RED.

You can get rid of a window or subwindow by typing Kill, putting the cursor in the doomed window, and holding down RED or BLUE for about a second. RED will give the space of the window to the window above; BLUE will give it to the window below. Kill YELLOW will clear the window; it is equivalent to Everything Delete (except that you can't Undo it).

If you attempt to Kill the only window showing a document that you have changed and haven't filed with Put, Bravo will pause with the message "Type CR to confirm". If you really want to lose the changes you have made to the document, type CR; otherwise, type DEL and then file the document using Put.

In summary:

	Window	Kill
RED	move boundary	give space to window above
YELLOW	split; new subwindow	clear
BLUE	new window	give space to window below

You can remove the *system window* from view by striking the unmarked key to the right of RETURN. This provides more space on the screen and in Alto memory for your document, and also makes subsequent commands execute slightly faster. You can restore the system window by striking the same key again. The system window is restored automatically when you make an error, and at certain other times when Bravo wants you to see what is in the window.

4.3 If Bravo breaks

When Bravo breaks or *crashes*, what usually happens is that Swat gets called; the manifestation is a couple of seconds of whirring from the disk, followed by a mostly blank display on the screen, with the words Swat version *xx* at the top. If this happens, look at the bottom of the screen, where there will be a more or less intelligible message. In some cases this message may describe a condition you can do something about, e.g., that your disk is full. Or it may inform you of a parity error; if this happens repeatedly, you should file an Alto trouble report to get your Alto repaired (see section 5.1 of the Alto Non-programmer's Guide). A third possibility is some fairly meaningless message describing an internal Bravo malfunction. In any case, after looking at the message you should type K^c (if that doesn't work, boot the Alto). Then, if you want to recover your work, you can proceed as described below.

Bravo makes a record of everything you do during a session; the record is called the *transcript*. It is useful for three reasons:

If Bravo crashes because it has a bug, the transcript can be used to report the problem to the people responsible for fixing bugs.

If Bravo crashes because of a hardware failure of your Alto, a power failure, you accidentally pushing the boot button, or whatever, you can recover your work by *replaying* the transcript. In this case, the last few characters you typed may be lost.

If you make a mistake, like deleting half of the document you have been editing for several hours, you can replay the transcript up to the error and then save the document.

You can do all these things using a system called BravoBug. Thus,

>BravoBug^{CR}

will start replaying your transcript. As the replay proceeds, Bravo will report each command, just as it does when you type a command in the usual way. When it is finished, Bravo will say Ready, and you can go on editing. It is a good idea to save your work with a Put and start Bravo again.

Warning: you can only do a replay if you haven't started Bravo up again. Once you restart Bravo normally, your chance to replay is lost.

To *report* a bug in Bravo and then do a replay, type

>BravoBug/R^{CR}

This will deliver to the Bravo maintainers copies of all the files involved in the bug, including the transcript, and then start Bravo to do the replay.

You can control the replay, step by step, as described in this paragraph. To stop a replay which is going on, type a space. As soon as the command currently being replayed is finished, Bravo will stop and tell you the *number* of the next command. At this point you can type

Quick to make typed-in text go in all at once during the replay, rather than one character at a time. This is faster, but you don't get to see what is going on. Quick is the normal mode.

Slow to make typed-in text go in one character at a time.

space to replay one more command. Note: only commands which change the text or windows are recorded in the transcript, not scrolling operations.

Proceed to continue replaying at full speed. You can stop the replay again at any point by typing a space.

Break before command *n* to make the next Proceed stop before command *n* (of course, it will still stop right away if you type a space). This is useful if you know that the first 50 commands are good, but want to proceed more cautiously after that. Note that after a Bravo crash, the Swat display usually tells you the number of the command during which the crash occurred.

Terminate CR to terminate the replay. After terminating, you can proceed to give ordinary Bravo commands. Don't do this unless you are sure that you want to stop replaying, since there is then no way to resume it.

You should try replaying a Bravo session and using these commands, so that you feel comfortable with them. You will then feel much more in control when you have a problem with Bravo or your Alto, or make a serious blunder while editing.

4.4 Arithmetic

Bravo incorporates a simple calculator, modeled after the Hewlett-Packard 35. The calculator has a *stack* with room for four numbers; while you are using it the top of the stack is displayed in the search key buffer, in the lower right corner of the system window.

To enter a number in your document onto the stack, select it and type \ (enter). To add a number in your document to the top of the stack, select it and type + (or =, which is the lower-case character on the same key). Similarly, you can subtract with −, multiply with * (or ←), and divide with /. The % command is just like *, except that it divides the result by 100. After any of these operations, the top of stack is the current buffer, which means that you can insert its contents in a document by defaulting the text of the insertion with ESC.

If you want to type in a number instead of selecting it, just type the number, and end it with one of the calculator commands. The number will appear in the middle buffer while it is being typed.

You can operate on the top two stack elements, rather than on the current selection and the stack, by prefixing the operation with the Calculator command. Thus, to compute $(a + b) * (c + d)$, you

select *a*	\ (enter)	Stack: *a*	-	-	-
select *b*	+	Stack: *a*+*b*	-	-	-
select *c*	\	Stack: *c*	*a*+*b*	-	-
select *d*	+	Stack: *c*+*d*	*a*+*b*	-	-
C *		Stack: (*a*+*b*)*(*c*+*d*)	-	-	-

This also works for enter: Calculator \ duplicates the top of the stack; if the stack was *a b c d*, it becomes *a a b c*.

There are a few more calculator commands which are occasionally useful:

~ exchanges the top two elements of the stack: *a b c d* becomes *b a c d*.

↑ rotates the stack, bringing the second element to the top and the top to the bottom: *a b c d* becomes *b c d a*. Four repetitions of ↑ leave the stack where it was.

Calculator *n* sets the number of digits after the decimal point to *n*. It is initialized to 2. All calculator arithmetic is rounded.

Calculator Fixed sets the display to fixed point (the normal mode).

Calculator Scientific sets the display to scientific notation.

Calculator Engineering sets the display to engineering notation.

Calculator Radix *n* sets the radix. *n* can be a digit, or Binary, Octal, Decimal or Hexadecimal.

4.5 Other useful features

BUFFERS

The system window contains three pieces of text enclosed in curly brackets. These are called buffers, and they are used for a variety of purposes, some of which you have already encountered. The three buffers are numbered, as follows:

1 {last deletion} 2 {last insertion} 3 {search key}

One of the buffers is always marked with a "*"; that one is the *current* buffer, and its contents are usually what is inserted when you default a text insertion by simply typing ESC.

Commands which insert into buffers, like Jump and Substitute, default to the old contents of the buffer being loaded. Get and Put default to the file name already in the window.

The text in the buffers is always in visible mode, i.e., with spaces, TABs and CRs shown explicitly. Furthermore, TABs and CRs don't have their usual effect of leaving white space, because there is no room in the system window for these effects. Finally, if there is too much text to fit in the space allowed for the buffer on the screen, the middle of the text is replaced by an ellipsis (...).

You can force buffer *n* to be the current buffer with the command Buffer *n* ESC. You can set the contents of buffer *n* with the command Buffer *n* followed by typing or a copy selection.

PARTIAL SUBSTITUTION

If you want to substitute "that" for "this" you can use the Substitute command described in section 4.5. If you want to change *some* of the occurrences of "this" to "that", however, it is useful to know about the Find and Yes commands.

Find is exactly like Jump, except that

the search starts from the end of the current selection, not from the top of the display;

the occurrence of the key which is found becomes the current selection (Jump leaves the selection unchanged).

Yes is equivalent to Replace ESC Find ESC. So, to change some "this"es to "that"s, proceed as follows:

select something before the first place you want to start looking;

Find this ESC;

ESC to repeat the Find until you get the one you want to change;

Replace that ESC;

Find ESC;

Now at each point type ESC (i.e., repeat the Find) to make no change and go on to the next "this", or Yes to make the same change you made last time and then go on.

CONTROL AND SPECIAL CHARACTERS

Bravo normally displays a control character as the corresponding lower-case letter (or whatever) with an overbar. If you turn on Look graphic it will try to display the control character from the font (if there is something in the font for it; otherwise it will display a black rectangle). This is a character Look, just like Look Visible.

You can't type a control character in directly, but you can type the corresponding regular character, *followed* by a S^c, which converts the preceding regular character into a control character.

Certain font families (notably TimesRoman and Helvetica) contain extra characters, such as accents, ligatures, and several sizes of dashes and spaces, which are useful in creating very high-quality documents. These may be entered as control characters in graphic mode (use Look graphic). The correspondence between control characters and their printed representations is as follows.

Accents. Each of these has a width of zero so should be typed in immediately *before* the character above which it is to appear.

d	d s^c	example:	do
e	e s^c		eo
k	k s^c		ko
p	p s^c		po

Ligatures:

ff	f s^c	replaces:	ff
ffi	q s^c		ffi
ffl	r s^c		ffl
fi	t s^c		fi
fl	u s^c		fl

Quotation marks:

‘	g s^c	open single quote
’	’	close single quote (standard apostrophe)

Note that you can make open and close "double quotes" by using pairs of the corresponding 'single quotes'.

Spaces and dashes:

||	space bar	letter space; here are 10 of them: | |
| |	y s^c	figure space: | |
| |	o s^c	em space: | |
-	-	hyphen: ----------
–	v s^c	en dash: ––––––––
−	n s^c	minus sign: − − − − − − − − − −
—	s s^c	em dash: ————————

Miscellaneous:

¿	b s^c
ç	c s^c
ï	h s^c

At the time this manual was prepared, the special characters had not yet been incorporated into the *.al display fonts. Until this has been done, these characters will all appear as black rectangles on the screen. To find out which control character one of these black rectangles corresponds to, turn off its graphic look.

4.6 The user profile and fonts

The file User.cm is your *user profile*, which contains information for various systems about how you want them to be set up for your use. The information for each system starts with the name of the system in brackets, e.g. [BRAVO]. Then follow a series of lines of the form

label: information

Bravo currently accepts three kinds of information in the user profile: initial and quit *macros*, described in the next section, and font definitions and default parameter settings, which are discussed here. Look at your User.cm file now, to see how this works.

Each line of font definition tells

the *number* of the font (0 to 9)

the *name* and *size* (in points) of the hardcopy font

the name and size of one or two screen fonts which can be used to represent that hardcopy font on the screen. Bravo will use the first one normally but the second one (if present) in hardcopy mode.

For example, the line

FONT:0 TIMESROMAN 10 TIMESROMAN 12 TIMESROMAN 10

says that font 0 is to print as 10 pt Times Roman when you generate hardcopy.

The rest of the line says that font 0 can be represented on the screen by the screen fonts stored on files TimesRoman12.al and TimesRoman10.al, which must be present on your disk. Bravo will use the 12 pt version in normal display mode and the 10 pt version in hardcopy mode. The extension "al" is used for screen fonts. There are no files for the bold or italic versions of screen fonts,

because Bravo can construct them from the regular version.

In addition to all the *.al files, you must also have a file called Fonts.widths, which contains information about the widths of all the characters in the hardcopy fonts.

Warning: A Bravo document file does not contain the association between the font numbers and names used to create that document. If you modify the font definitions in your user profile, you may encounter problems when you attempt to share Bravo documents with other people.

The user profile also contains settings for

default left and right margins;

tab interval;

Look nested parameters;

default line leading (used by Look X) and paragraph leading (used by Look Open, Look Q and the default for Look Y);

standard offsets (used by Look ↑, Look ←, and the default for Look Up and Look Down).

You can change all these settings by editing User.cm in a way which should be obvious; after doing this, be sure to initialize Bravo. Except for the default margins, all these settings affect only the process of editing the document, and not the document itself. In other words, once a Look nested, superscript or whatever has been done, the margins, offset, etc., have been set in the document and cannot be affected by subsequent changes in the user profile.

There is also a [HARDCOPY] section in the user profile that is used both by Bravo and by other subsystems that generate hardcopy. This contains entries for

preferred printing format (Press is the only reasonable choice at present);

name of your Press printer;

the name to be inserted in the "Printed by" field on the break page of the printout (an occurrence of "$" is replaced by the name by which you are known to the Alto Executive).

4.7 Startup and Quit macros

This section is only for programmers, and is not recommended even for them.

You can put into your user profile sequences of Bravo commands, called macros, to be executed automatically when you start up or quit from Bravo. Each macro is named by a letter. Startup macro *x* will be executed if you start Bravo with

>Bravo/x . . . CR

Quit macro *x* will be executed if you type an "x" instead of CR after typing Quit.

You can see the format of a macro definition by looking at your user profile. The command sequences are just like those which Bravo writes into the transcript (the file Bravo.ts), and can be constructed by actually executing the desired sequence of commands, and then copying Bravo.ts to another file and copying the sequence out of that file. There are two exceptions:

* in place of a selection (which looks like {6,2,123,648}) means the current selection;

@n in place of typed-in text (which looks like 'text{) means the *n*th *parameter.* For startup macros the parameters are strings on the command line separated by blanks; the first one after Bravo/x is numbered 1. For quit macros, parameters 1, 2 and 3 are the three buffers, 4 is the file name for the first window, 5 the file name for the second window, etc.

4.8 Press and Diablo hardcopy

To make a Press file on your disk instead of sending your hardcopy directly to a printer, use the File option of the Hardcopy command. It will ask you for the name of the file; the recommended name is the name of your document, with the extension replaced by "Press". Once you have made this file, you may print it using the EmPress subsystem, store it on a file server to make it available to other people (see sections 6 and 7.4 of the Alto Non-programmer's Guide), or give it to PressEdit to combine it with other files into a large document (see section 7.6 of the Guide). *Warning*: the Press file is typically 50% larger than the document file; be sure you have enough room on your disk.

To send hardcopy to some printer other than your regular one (specified in User.cm), use the Hardcopy option

@ *name* ESC

where *name* identifies the printer you wish to use.

For both Press and Diablo printing, you can start printing at a specified page number with the Hardcopy option

Start at page *n*

This is mainly useful for Diablo printing. There will usually be a substantial delay while Bravo figures out where page *n* starts.

To print on the Diablo printer, you can use the Diablo option to the Hardcopy command. Before doing this, be sure a printer is plugged into your Alto; when you plug or unplug it, *turn the Alto power off first.* This option has an array of sub-options, which you can invoke when it pauses before printing each page. At the pause, the system window says "Ready to print page *n*", and the beginning of the text for page *n* is displayed in the document window (if there is room in Bravo's memory). You can then say:

Repeat last page to prepare to reprint the previous page, instead of the page which was going to be printed next.

Continuous print, to suppress the pause after each page; this is useful if you have continuous forms in your printer. You can still stop the printing by typing space during printing, as described below.

space to start printing the page which Bravo says it is ready to print.

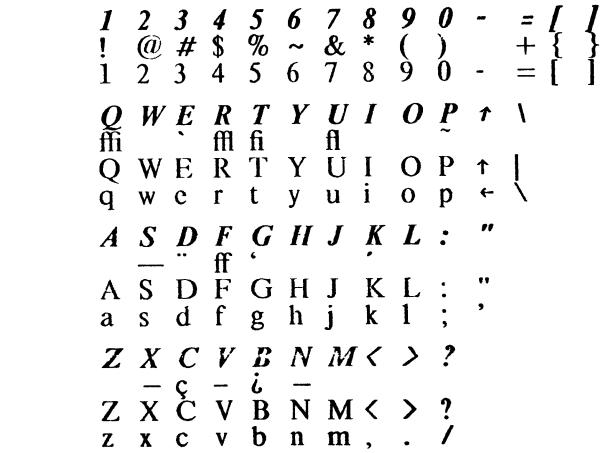
During printing, you can abort printing of the current page at any time by typing a space. You can then use the options just described to restart the current page (with space) or reprint the previous page (with Repeat). If you want to start at another page, use DEL to leave the Hardcopy command, give another Hardcopy command, and use the Start at page *n* option.

There is a command Look Magnify Diablo that displays the document as closely as possible to the way it will appear when printed on the Diablo printer. Look Hardcopy terminates this mode.

Fonts

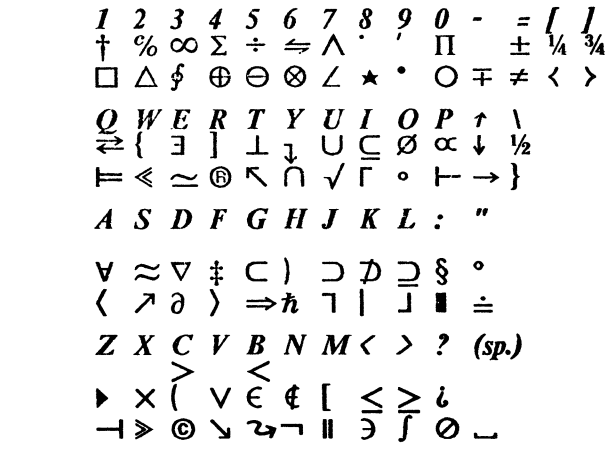
Times Roman

Font 0 = 10pt., Font 1 = 8pt., Font 5 = 12pt.
Available in 6, 7, 8, 10, and 12pt. in Reg., **Bold**, *Italic*, and ***Bold Italic***.
Available in 18pt. in Reg. and **Bold**.
↑O = Em Quad; ↑Y = En Quad & Fig. Space.
Space Bar = 1/4 Em Quad.



Math

Font 3 = 10pt.
Available in 8 and 10pt. in Reg. only.
↑D = 1 pt. space



Logo

Font 2 = 24pt.

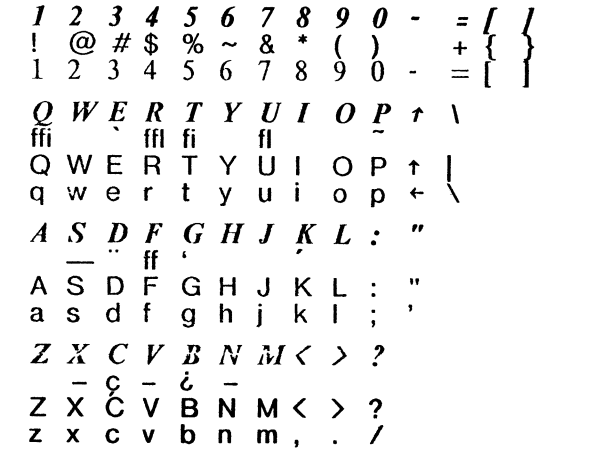
XEROX

NOTES:

1. In Times Roman, Helvetica, and Hippo;
   ↑N = ↑X = Minus sign,
   ↑V = En Dash,
   ↑S = Em Dash.
2. The Bold Italic characters on this page represent the keys on the keyboard.
3. Three lines of characters below the keys are Control, Upper Case, and Lower Case respectively.
4. Two lines of characters represent Upper Case and Lower Case respectively.
5. Blank lines are not blank but contain spacing characters.

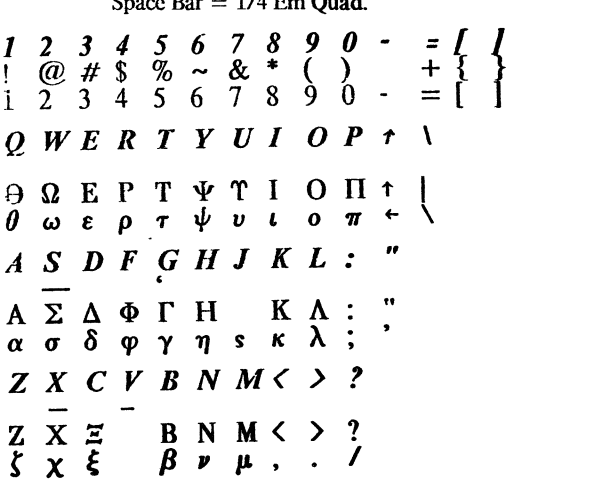
Helvetica

Font 6 = 10pt., Font 7 = 8pt., Font 9 = 18pt.
Available in 6, 7, 8, 10, and 12pt. in Reg., Bold, Italic, and Bold Italic.
Available in 18pt. in Reg. and Bold.
↑O = Em Quad; ↑\ = En Quad; ↑Y = Fig. Space
Space Bar = 1/4 Em Quad.



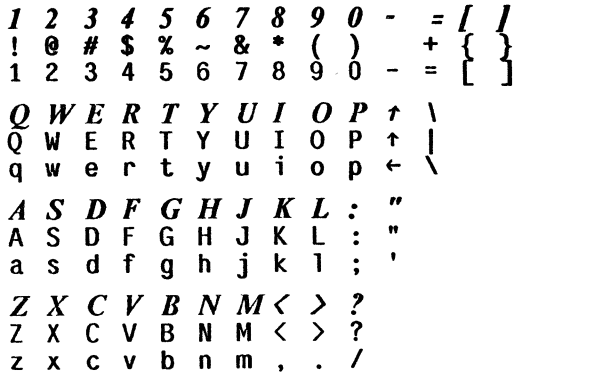
Hippo

Font 4 = 10pt.
Available in 8 and 10pt. in Reg. only.
↑O = Em Quad; ↑Y = En Quad & Fig. Space.
Space Bar = 1/4 Em Quad.



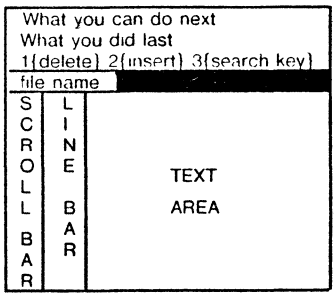
Gacha

Font 8 = 10pt.
Available in 8 and 10pt. Reg. only.

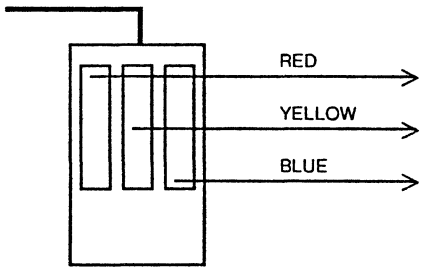


BRAVO VERSION 7.2 SUMMARY

THE SCREEN



THE MOUSE



WHAT MOUSE BUTTONS DO

	Scroll bar	Line bar	Text area
RED	Scroll up	Select line	Select character
YELLOW	Thumb	Select paragraph (Sets cursor to ¶)	Select word
BLUE	Scroll down	Extend	Extend

The CLR key is the blank key to the right of BS on Alto-I or the key labelled BW on Alto-II.
The blank key to the right of RETURN removes or restores the system window.

COMMANDS

CHANGING TEXT

Delete selection
Insert *text* before selection
Append *text* after selection
Replace selection by*text*

(These make a new paragraph if a paragraph is selected.)

During type in: Backspace character with BS or ctrl-A
Backspace word with ctrl-W
Separate paragraphs with ctrl-CR
Enter ctrl-*char* as *char* ctrl-S

text may be typed or selected. ESC gets the current buffer Buffer n ESC makes buffer n current

FILING

Get to read in a file
ctrl-Z for unformatted text
Put to write out a file
File name goes into bar above window. ESC gets you one already there. Can edit file name like any other text.

Hardcopy to print. Options:
Copies n Start on page n
File name.press (no printing)
@ printer-name

CR to do it
Diablo output, with sub-options: Space to start or abort
Continuous printing
Reprint last page

MISCELLANEOUS

Undo works on most recent command.
ESC repeats most recent command, using current selection.
Everything selects whole document.
Normalize moves current selection to top of window.
([{ < ' " put indicated brackets around current selection.
)] } > extend current selection to closest matching brackets.
LF takes current selection as top of page, and moves selection to top of next page.

DEL cancels what you are doing.
Quit CR exits from Bravo
Time inserts current date and time.
Buffer 123 *text* sets buffer and makes it current.
ESC instead of*text* just makes it current.

LOOKS

Basic looks. Mostly, SHIFT means NOT. Thus, Look b turns on bold, Look B turns it off.
During type-in, use ctrl-b for bold, etc. In command mode, give the Look command, then the letter alone (no ESC).

Text:			Paragraphs:	
	bold	· underline	centered	justified
	italic	←/↑ sub/superscript	q - add 6 pts	open - add 12 pts space before paragraph
	visible	0-9 font	nested (indent)	
	graphic	CLR to reset	; document profile	

Look has more cases, which take a distance as parameter. These cannot be used during type-in.

Text:
- Up raises text; default 4 pts.
- Down lowers text; default 4 pts.
- n names a tab (n = 1-9, a-e)

Distances (measured from left or bottom page edge):
- As a number: in points (72/inch) - 123 or 123 pt.; in inches - 4.5 or 4.5" or 4.5 in; in cm - 4.5 cm; in blank widths - type that many blanks
- As an increment to current value: + or - followed by a number as above
- As the left edge of a character: select with RED
- As a position on the screen: point with BLUE
- Defaulted to a standard value: type \.

Paragraphs:
- Left margin; default 85 pt
- F - left margin of first line
- P - left margin of other lines
- Right margin: default 527 pt
- X - space between lines; default 6 pt
- Y - space between paragraphs; default 12 pt
- Z - vertical tab; default none
- Keep on current page; default 0; 11" forces new page
- TAB n d sets tab stop n (n = 1-9, a-e); default not set
- TAB = d sets even tab stops, default 36 pt

Standard Fonts

0 Times Roman 10
1 Times Roman 8
2 Logo 24
3 Math 10
4 Hippo 10
5 Times Roman 12
6 Helvetica 10
7 Helvetica 8
8 Gacha 10
9 Helvetica 18

There are samples in the manual

Look Same *selection* sets all text looks of current selection to be the same as those of *selection*.
Look All *selection* sets all paragraph looks of current selection to be the same as those of *selection*.
Look ? displays looks of selected text or paragraph in buffer 3.
Look hardcopy makes screen match hard-copy with 10% magnification. Look Hardcopy clears it.
Look Magnify Diablo matches Diablo harcdcopy.

PAGE FORMATTING

The following can be in the document profile; it must be the first paragraph, and have the profile property (Look;):
Private Data Stamp: Yes/No X:d Y:d
Columns:n Edge Margin:d Between Columns:d
Margins: Top:d Bottom:d Binding:d
Line Numbers: Yes/No Modulus:n Page-relative First Line:n
Page Numbers: Yes/No X:d Y:d First Page:n Roman Uppercase Not-on-first-page
Heading or Odd/Even Heading: Not-on-first-page followed by a heading paragraph, also with profile property

d is a distance: use - to measure from top or right
ctrl-L causes page (or col) break after current line
Use two ctrl-Ls for page break with multiple cols
Also note Look Keep and Look Z (vertical tab)

WINDOWS

Window			Kill	
	RED	to move boundary	RED	to merge with window above
	YELLOW	to split (make new subwindow)	YELLOW	to erase contents, leave the window
	BLUE	to make new window	BLUE	to merge with window below

Window Edge *distance* sets the left edge of the window at the specified point on the page; default is 85 pt.

SEARCHES

Jump *text* starts search at second line, doesn't move selection
Find *text* starts search after current selection, moves selection to the string found
Substitute *text* for *text* · works on current selection. Usually you want to do Everything first.
Yes is equivalent to Replace ESC Find ESC; use Yes when you want to confirm substitutions.

REPLAY

> Bravobug will start a replay. > Bravobug/R will report a bug to the Bravo maintainers first, then replay.
Space will stop the replay and show you the number of the next command. You can then type:
Slow to slow down typein Space to replay one more command Break n to stop before command n
Quick to speed it up Proceed to continue replay full speed Terminate CR to stop replaying

ARITHMETIC

Operators are + = - * ← % / \(enter) An operator as a command combines the selection with the top of stack.
You can also type a number, followed by an operator. The stack is just like the one in a Hewlett-Packard calculator.
Also ~ exchanges x and y
↑ rotates the stack

Calculator, with options

Fixed	operator to operate on x and y
Scientific	0-9 to set digits after decimal pt
Engineering	Radix n to set the radix (n = H for hex)

LAUREL

by D. BROTZ
R. LEVIN

Laurel Manual

Table of Contents

1. Introduction

What is Laurel?

Laurel is an Alto-based, display-oriented, message manipulation system. It provides facilities that permit its users to display, forward, classify, file, and print messages, and to compose and transmit messages and replies. Laurel is an initial component of what will ultimately be a distributed message system. Although the distributed nature of this system has inherent technical interest, it is largely transparent to the users of the system, who see a collection of logical facilities resembling those provided by MSG on Maxc. Eventually, the services of Laurel will surpass those of MSG, but at present, the two are roughly equivalent in function. The important distinction for now is that Laurel executes on an Alto and uses the display in a fashion befitting Alto-based software. It also produces files that can be manipulated by other Alto subsystems.

Many initial users of Laurel will be familiar with MSG and will naturally be interested in the functional differences between MSG and Laurel. This manual, in addition to presenting the facilities of Laurel, points out some of the incompatibilities with MSG. *Prospective Laurel users with strong ties to MSG should read section 5 carefully before committing themselves to Laurel.* The Laurel team fully expects that some potential users will find the transition to Laurel too uncomfortable to undertake at present. Accordingly, we have provided a "free sample" of Laurel in the form of a tutorial (see section 2), which can be run without any commitment to further use of Laurel or danger to MSG files. We would much prefer that Laurel users "go in with their eyes open" and not be unpleasantly surprised when confronted by incompatibilities.

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 probably use Laurel quite competently after reading only the introductory sections of this manual. (There is also a tutorial available to guide new users; see section 2.) However, Laurel has features that are not immediately obvious, and after becoming acquainted with the system, you will want to read about these facilities. The Laurel team will be very happy to hear any suggestions you may have, and is particularly interested in your experiences in using the initial system. Comments should be sent to LaurelSupport (using the facilities provided by Laurel!)

The version described in this manual is Laurel 2.0. Laurel displays its version number in the upper-left corner of the screen. Future versions of Laurel containing major new capabilities will change the integer part of the version number, whereas maintenance releases will change the fractional part.

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, at present, that your mail files reside on your own Alto disk(s). Laurel provides facilities for manipulating mail files, examining and responding to messages, composing and sending new messages, and cataloging, filing, and printing messages. These functions are discussed below.

How to obtain Laurel

We recommend strongly that new users obtain a Non-Programmer's disk to be used primarily for processing and managing messages (see section 5 for the reasons behind this recommendation). The BASIC NON-PROGRAMMER'S DISK already has Laurel installed, so if you have obtained a copy of this disk (by the procedure described in the Alto Non-programmer's Guide), you may skip the next paragraph.

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

This obtains Laurel and starts it up in a tutorial mode. If you are already familiar with Laurel, you should retrieve and execute <Laurel>Laurel.cm, which merely retrieves the files necessary to run Laurel.

Note to Mesa programmers: Laurel 2.0 is a Mesa 4.1 program and therefore requires the Mesa 4.1 version of RunMesa.run to be on your disk. For this reason it can't coexist on a disk with a version of Mesa other than 4.1.

Before using Laurel's **hardcopy** command for the first time, you must edit your *Laurel profile* (file Laurel.Profile) to declare the name of the printing server to which your hardcopy is to be sent. See section 3.6 for information on the Laurel profile.

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 section 3.5.

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

>Laurel Help^CR

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.

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 displayed command by moving the mouse until the screen 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 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.

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 typing ESC, CR, Y, or space, or by clicking YELLOW. If you type 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!

Type-in conventions

Laurel observes most of the standard Alto type-in conventions. When it expects you to supply text, it displays a blinking caret at the appropriate place on the screen. Striking BS or A^c deletes the character to the left of the caret; W^c deletes the word to the left of the caret. Type-in may always be terminated by typing ESC; sometimes other characters will terminate input as well (see below). Commands that do not permit you to make a secondary selection may also be terminated by clicking YELLOW.

Brackets

Several Laurel commands require text arguments. In particular, 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 with 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, the following rules apply: If you type ESC, CR, TAB, or space, or click YELLOW, the caret disappears and the contents of the brackets remain unchanged. If you type BS or W^c the contents of the brackets will be edited as if you had just typed that text. If you type anything else (except as described above), it replaces the complete text within the brackets. Type-in is terminated by ESC, CR, TAB, space, or YELLOW. Typing DEL aborts the entire 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

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 commands on the Laurel screen, 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 *selection 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 *composition and delivery menu.* As the name implies, this menu contains commands that facilitate the composition of new messages and responses to old ones. The composition region is separated from the feedback region by a single horizontal line.

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 behave as they do in Bravo. 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 that separates the region from the menu above. When the cursor is positioned just below this bar and YELLOW is depressed and held, a short vertical line segment appears on the bar. Moving the mouse to the left or right causes a corresponding movement in this line segment. The position of the line segment relative to the left edge of the horizontal bar identifies a location within the text associated with the region. Thus, moving the line segment to the extreme right edge of the bar identifies the end of the associated text; positioning it in the center of the bar specifies the middle of the text. When YELLOW is released, the identified position is brought to the top of the region.

A portion of the thumbing bar appears as a dashed line, whose position and length correspond to the text displayed in the window. Thus, positioning the line segment at the left edge of the dashed line identifies 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 selected entities in each region 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 line segment 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 segment on the thumbing bar. Releasing the mouse button then causes the selected text to be moved to the top of the region.

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 two 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 box will capture the cursor 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—we will discuss this in section 3.1. 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). This pointer is used to identify messages that you wish to manipulate, and can be repositioned with the mouse buttons. We will discuss it in detail shortly.

2.4. The upper menu

The upper menu contains some obvious status information: the date and time, the version of Laurel you are running, and the amount of free space remaining on your Alto disk. There is also a space reserved for posting the status of your in-box—see section 3.4. Of the remaining items, which are all commands, we discuss only **Mail file** and **New mail** here. The other commands are documented in sections 2.8 and 3.3.

Mail files

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 file folder labelled "Active.mail", and if not, it creates an empty file and labels it 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 3.1.

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 the description of {*brackets*} in section 2.1.) 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 the facilities described in subsequent sections to manipulate this file.

Mail file {*file*} always reads from the file whose name is displayed in the 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.

A fine point: Laurel acts on deletions (see section 2.5) 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.

New mail

Messages that are waiting for you reside in your *in-box*. The *in-box* is also called the *mail box*, but we will avoid this term to prevent possible confusion with *mail file*. 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 hour-glass, confirming that Laurel is busy shuffling messages. You will also observe the free page counter changing. When all messages have been transferred, the gray background and hour-glass 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.

As the **New mail** command is completed, the table-of-contents region is scrolled so that as much as possible of the new mail is displayed, and the selection pointer is set to point at the first new message. You will also observe that each new entry has a "?" to its left—this indicates that the

contents of the associated message have not yet been examined. The "?" is a *mark character*; you can change it to provide a (primitive) classification of the message—see section 3.1.

2.5. The selection commands menu

Displaying messages

Normally, to examine the contents of a message, you must first select its table-of-contents entry with the selection pointer (see below). 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** 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 (see below), you need not do so. Simply click **Display** repeatedly and Laurel will advance the selection pointer by one entry each time.

A fine point: 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**.

Selecting messages

Entries in the table-of-contents region are selected in the same way that lines of text are selected 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. The selection pointer will move to the indicated entry. By performing explicit selections, you may examine the contents of your mail file in any order you wish.

More generally, you can select any consecutive group of entries in the table-of-contents. This is not particularly useful if you only wish to display messages, but can be convenient if you wish to delete a group of messages (see below), move a group of messages to another file (section 3.1), or generate hardcopy (below). You first select a single message, as described above, then extend the selected range by moving the cursor either up or down and clicking BLUE. Thus, selections are made with RED and BLUE in much the same way they are in Bravo. YELLOW has no effect.

Deleting and undeleting messages

After examining some of the messages in your mail file, you may wish to delete them. The **Delete** command, when activated 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.

If you discover that you have inadvertently deleted some messages that you want to keep, simply reselect them if necessary, position the cursor over **Undelete**, and click RED. The lines drawn through the table-of-contents entries of the selected messages will be removed.

Deleted messages are expunged (i.e., removed permanently) from the current mail file whenever a **Quit** (see section 2.8) or **Mail file** command is executed.

Generating hardcopy

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. Laurel will generate hardcopy of each message in the form of an inter-office memorandum, one page (or more if necessary) per message, and send it to be printed on your default printing server. 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.

There are no options that you can exercise at the time you issue the **Hardcopy** command. However, by changing your *Laurel profile*, you can control certain aspects of the hardcopy formatting as well as a few other things such as the name of your printing server (see section 3.6).

2.6. Composing and delivering messages

Composition is the process of building the header and body (see the beginning of section 2) of a particular message. *Delivery* is the process of transmitting a composed message to its specified recipients. Laurel separates these two actions and supplies distinct commands appropriate to each in the composition and delivery 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* an existing one to a new recipient, or *get* a previously-composed form or message from a file on your Alto disk. In the composition and delivery menu (below the message display region), 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 (see below) 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. **Get** will be described in more detail later (section 3.2).

Composition

New form gives you a new message form in the composition region. The form contains "To", "Subject", and "cc" fields, which you should fill in as appropriate. Laurel provides a subset of Bravo's editing commands to permit you to compose these fields and the message body. Laurel provides the Append, Delete, Insert, Replace, ESC (repeat) Undo, and secondary selection functions of Bravo, but does not handle italics, bold-face, multiple fonts, or any "looks". For important additional editing functions, see section 3.2. In short, 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.

At present, the Laurel editor is not quite compatible with Bravo, though it may be in the future. In particular, you should be aware of the following differences. An Undo will undo the previous command. Undo always undoes the previous command *exactly where it was given*. To move text from one place to another use Delete followed by Insert ESC. **New form**, **Answer**, **Forward**, and **Get** may also be undone.

In general, 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 each of the three 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 (see below), Laurel will supply your name and

the date, so you need not include them explicitly.

Answer and **Forward** initialize a message form in different ways, but both take information from the message *currently displayed in the message display region* (which is not necessarily the one selected in the table-of-contents region). **Answer** fills in the "To" field with the sender of the currently displayed message and sets the subject to be "Re: *sender's subject*". It also sets the "cc" field to include all of the recipients of the message being answered. If you don't like these substitutions, you may, of course, change them using the editing facilities. **Forward** copies the message body from the display region into the composition region. After clicking either **Answer** or **Forward**, you must complete the message by editing any remaining uninitialized fields in the message header and body.

Laurel identifies fields that it expects you to replace by supplying a keyword bracketed by black rectangles, e.g., ▮Recipients▮. Laurel will refuse to deliver any message whose header contains one of these fields. This protects you from simple oversights, such as forgetting to supply a "Subject" field.

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 hour-glass. 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 hour-glass 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.

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 3.3.

Deliver tells you the number of recipients to whom it will deliver the message. If this number exceeds 30, you must confirm the delivery. This is intended to minimize unintentional deliveries to large distribution lists. **Deliver** also tells you the size of the message (in characters).

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.

2.8 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. If you invoke **Quit** with BLUE, Laurel will omit the confimation prompt. When you re-enter Laurel at a later point, these messages will no longer appear in the table-of-contents. They are gone forever.

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

3. Additional facilities

3.1. Filing and classifying messages

Laurel provides two facilities to assist you in classifying your mail. Remember that your default mail file should be viewed as holding those messages upon which you have yet to act. Naturally, you may want to file away those messages on which you have *already* acted. The command **Move to** {*file*} in the selection menu allows you to do this. *Mark characters* provide a simple means of attaching labels to individual messages within a mail file.

Move to {file}

To invoke **Move to** {*file*} proceed as follows. First, select (with selection pointers—see section 2.5) the messages you wish to transfer to a separate file. Next, point the cursor at **Move to** and proceed as you would for **Mail file** (see sections 2.1 and 2.4). After you have supplied the file name, Laurel will append the selected messages to the indicated file. Laurel will draw a line through the table-of-contents entries of the messages that have been moved so that the corresponding messages will be deleted when you leave Laurel (or change mail files). If you wish to retain any of these messages in the current mail file *as well*, you may use "Undelete" as described above. Laurel will also *mark* (in the mark character field—see below) moved messages with an "m".

Move to does not move any messages that are indicated as deleted. If a range 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.

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 the mark character, position the cursor to the left of the desired entry but to the right of the line bar and selection pointers. 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 it is moved to another file (using **Move to** {*file*}). Note that "?" and "m" are valid mark characters but have additional semantics assigned to them by Laurel. Other mark characters may be assigned special meanings in the future, e.g., to permit selective display of the table-of-contents You may change the mark character by positioning the cursor over the mark to be changed and repeating the preceding steps. A space is a valid mark character.

3.2. Additional editing and delivery facilities

Secondary selection

Laurel supports secondary selection (for Insert, Append, and Replace commands) in the same manner as Bravo does. That is, you may identify the text string to be inserted by using the mouse buttons to select characters, words, lines, or paragraphs. You may select text in either the message

display region or the composition region, and you may scroll or thumb within either region while making the secondary selection.

Paragraphs

Although Laurel does not implement Bravo's notion of paragraphs, it does provide a selection mode that gives similar results in simple cases. Laurel defines a paragraph terminator to be two successive CRs. You may select a paragraph by positioning the cursor in the line bar to the left of the desired text and clicking YELLOW, just as in Bravo. The main visible difference is that Laurel will not display Bravo's paragraph symbol in the cursor.

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, no CR at the end). 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

Alpha users: Brown, Doe, Jones, Smith;

Name this file "AlphaUsers.dl".

Now, when composing a message in Laurel, insert the file name in the appropriate field of the message header (e.g., following "To" or "cc"), followed by an up-arrow character:

To: AlphaUsers↑

While delivering the message, Laurel will read the file to determine the recipients. Multiple distribution list specifiers may be used within a single field and may be intermixed with names of individual recipients—separate them all with commas. If you give your distribution list files the extension ".dl", you may omit typing the ".dl" to Laurel.

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

The present distribution list mechanism is a temporary one, and has some inconvenient aspects. In particular, since distribution lists are presently stored on your own Alto's disk, they can get out-of-date rather easily. Commonly-used distribution lists are maintained centrally, presently as files <Secretary>*.dl on Maxc. We have tried to streamline the task of keeping distribution lists current by providing a command file, DLUpdate.cm, which updates local distribution lists from the central source. Registered Laurel users (i.e., those listed in LaurelUsers.dl) are notified when DLUpdate.cm should be run. DLUpdate.cm is automatically retrieved by Laurel.cm and LaurelNewUser.cm (see section 2). Note that DLUpdate.cm updates *only* those distribution lists that you *already* have on your disk. To obtain a copy of a distribution list you don't already have you must use FTP to retrieve it from the <Secretary> directory on Maxc. Thereafter, running DLUpdate.cm will keep it up-to-date along with all the others.

Get and Put

The composition and delivery menu includes the commands **Get**, **Put** {*file*} for filing and retrieving the contents of the composition window. **Put** writes out the entire contents of the composition window onto the file whose name is inside the brackets. (If a file with that name already exists, Laurel requires an extra confirmation before overwriting it.) **Get** replaces the contents of the composition window with the text contained in the file. As usual, if you click RED over **Get** or **Put**, a blinking caret appears within the brackets and you may fill in or edit the file name before confirming the command, whereas if you click BLUE, Laurel will execute the command immediately using whatever name is already inside the brackets.

Put is 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. Obviously, you can use Laurel to edit arbitrary text documents in much the same fashion as Bravo; however, since Laurel does not provide any of Bravo's formatting capabilities (multiple fonts, looks, etc.), you will find Laurel to be of limited utility in this respect.

In addition to the **Get** and **Put** *menu commands* just described, there exist Get and Put *keyboard commands*. These work somewhat differently from the menu commands. In particular, whereas the **Get** and **Put** menu commands operate on the entire composition window, the Get and Put keyboard commands operate only on the *current selection*. Get and Put do, however, make use of the file name contained in the **Get, Put** {*file*} menu command.

The Put keyboard command first prompts you either to confirm the file name already in the brackets or to type in a new file name, then writes a copy of the *currently selected text* onto the file by that name. This text is not removed from the composition window.

When you issue the Get keyboard command, Laurel first puts a copy of the *currently selected text* into the file name brackets. You may now either confirm that file name (with ESC or whatever) or type some other name followed by ESC. Laurel then *replaces* the currently selected text by the contents of the file, leaving the replacement text selected.

The Get keyboard command can be used as a file substitution device anywhere within a composed message. To include the contents of a file within a message, insert the file name (including the extension) at the desired point. Then select the file name with the mouse buttons as you would for a Replace command, but type Get ESC instead of Replace. Laurel will replace the file name with the contents of that file.

It is sometimes useful to expand a distribution list *before* sending a message. You can do this by inserting the distribution list name in the appropriate header field, selecting it, then typing Get. In addition, there is a short-cut that permits you to omit the explicit ".dl" extension. If you type the file name followed by a "↑", select it, and type Get, Laurel will append the ".dl" (if no extension is present) and perform the Get. You may take advantage of this feature to peek at the contents of a distribution list: use Get as just described, then undo the expansion by typing Undo.

Put produces a file that can be read and manipulated by Bravo. In fact, you can create a Bravo document, use the Get keyboard command to read it into a composed message, then send the message. The recipient can perform a **Forward** to move the received message into the composition window, select the portion of the message consisting of the Bravo document, and issue the Put keyboard command to write it onto a file. It should be noted that Bravo formatting information in the document may contain unprintable characters, which are displayed by Laurel as black rectangles. However, Laurel does not actually alter these characters, so formatting information is not lost. You must take care to Put the entire Bravo document (and nothing else), or else Bravo won't be able to read it.

Message header format

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. However, two features should be noted. First, comments may be included in a header item by surrounding them with parentheses. Second, recipients at Arpanet sites other than Parc-Maxc may be specified as either "*name* @ *site*" or "*name* at *site*".

You should avoid typing extra CRs in message headers, since the ARPA standard specifies some less-than-obvious behavior in such cases. In practice, this is rarely a problem, because Laurel will supply line breaks for you (both on the screen and in the transmitted message).

3.3. Authentication and logging In

In order to perform **New mail**, you must be an *authenticated user.* This means your name and password must be accepted by a centralized authenticator (currently Maxc). When you start up Laurel, it obtains your name and password from the Alto Operating System and submits them to the authenticator for verification. If the authentication fails, the message "Name or password invalid" will appear in the upper menu, just below the Laurel version number. Laurel will not allow you to receive messages until you have been authenticated successfully.

To enter 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 (see 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. A fine point: At present, the name you supply to the **User** command is used as the directory on Maxc where your mailbox is located. This may not be true in the future, but it will always be possible for Laurel to locate your mailbox from the user name you supply.

Laurel normally requires that you be an authenticated user before it will allow you to receive or send messages. If, for some reason, you are not an authenticated user and you select **Deliver**, Laurel will prompt you for confirmation. If you confirm, Laurel will append the phrase "(not authenticated)" to the sender identification it includes in the outgoing message. This makes it difficult for others to masquerade as you (or vice-versa).

When you leave Laurel, your name and password are passed back to the Alto Operating System.

3.4. Polling for new mail

Laurel checks your in-box periodically to determine if new mail has arrived. If it finds messages waiting in your in-box, it stops polling and displays "You have new mail" below the Laurel version number at the left-hand side of the upper menu. You may then use the **New mail** command (see section 2.4) to move the messages in your in-box to the current mail file. After you have done so, Laurel will remove the "You have new mail" message and resume its periodic polling.

Because this polling is completely automatic, you can leave your Alto running Laurel when you are not using it for other things, and glance at the screen occasionally to see if new messages have arrived. The polling interval at present is about 5 minutes.

3.5. 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 use ".mail".

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**. Upon the completion of the **New mail** (if any), 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 determine whether your in-box has messages, it ignores the /c switch.

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 2.4.

3.6. The Laurel profile

Laurel obtains certain configuration information and options from your *Laurel profile*, contained in file Laurel.Profile on your Alto disk. A standard Laurel profile is installed when you obtain Laurel using the LaurelNewUser.cm or Laurel.cm command file (section 2). The only things in the Laurel profile you are likely to want to change at present are the name of your printing server and perhaps some of the hardcopy format options. For completeness, however, nearly all possible options are documented here.

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 file must adhere to this format. This implies that there may not be any blank lines in the file. Furthermore, the last character in the file must be the CR of the last parameter line. Bravo formatting is *not* permitted. You should probably use only Laurel to edit Laurel.Profile. Note, however, that Laurel configures itself according to Laurel.Profile when it is started up, so after editing Laurel.Profile you must **Quit** and restart Laurel before the changes will take effect. The Laurel program processes this file very quickly, with only minimal error checking. If Laurel.Profile is not properly formatted, the only indication from Laurel will be the exception message "The file Laurel.profile is missing or bad."

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:

Send: Maxc
Retrieve: Maxc
Authenticate: Maxc
Hardcopy: Clover
PrintedBy: $

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

Send: *host name*

The name or network address of the mailbox server machine to which you will send all outgoing mail. (Default: Maxc) Network address constants of the form *number# number#* must be used at sites where there is no gateway providing name lookup services.

Retrieve: *host name*

The name or network address of the mailbox server machine from which you will retrieve new mail. (Default: Maxc)

Authenticate: *host name*

The name or network address of your authentication server, i.e., the machine that verifies user name and password combinations. (Default: Maxc) Normally the values specified for Send, Retrieve, and Authenticate will all be the same.

Hardcopy: *host name*

The name or network address of the hardcopy server machine to which all hardcopies will be sent. (Default: an illegal server name)

PrintedBy: *text*

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: $)

Herald: *text*

A line of text to be printed at the top of the first page of each hardcopied message. (Default: Inter-Office Memorandum). You should take care to ensure that the herald will fit on one line when hardcopied.

HeraldFont: *font*

The font that will be used to print the herald on hardcopied messages. A font value must take the form:

family-name point-size face

Examples:

TimesRoman 14 B
Helvetica 10

The optional B or I following the point size indicates bold or italic face. If the CR immediately follows the point size, the regular face is used. (Default: TimesRoman 12 B)

Logo: *text*

A line of text to be printed at the logo position between the header and the message body on the first page of each hardcopied message. (Default: XEROX). You should take care to ensure that the logo will fit on one line when hardcopied.

LogoFont: *font*

The font that will be used to print the logo on hardcopied messages. The format for the font value is given above under HeraldFont. (Default: Logo 24)

4. 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 (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, 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 contain within the form, after which Laurel will again be available for normal use.

5. Laurel and MSG

This section addresses the relationship between MSG and Laurel. As mentioned in section 1, 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.

Before proceeding to the gory details, we should remind you that these comments apply *only* to Laurel versions 1 and 2. Most of the difficulties described below are artifacts of a temporary implementation of a number of Laurel facilities. Future releases will cure these ills, so if Laurel appears insufficient for your present needs, don't give up.

Mail files

Laurel keeps nearly all of its files on your Alto disk. The only exception is your in-box, which is currently file Message.txt in your directory on Maxc. Getting new mail (with Laurel) moves messages from Maxc to your Alto disk, *emptying Message.txt.* This makes it very inconvenient to continue to use both Laurel and MSG, and it makes Laurel inconvenient to use if you switch Alto disks frequently. For now, our best advice is to designate a disk on which you will use Laurel exclusively, and stick to it. You will need about 500 disk pages to hold Laurel (including RunMesa.run), a moderate-sized mail file, and several distribution lists.

Once you have mail files on your Alto disk, it becomes impossible to process them using MSG, since they are on the Alto. 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

are willing 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.* If you are *not* willing to convert completely, we recommend that you wait to use Laurel until it is able to deal with remotely-stored mail files.

On the positive side, your correspondents never need to know that you have converted to Laurel, since Laurel is able to process your Message.txt file directly. All that has happened is that you have started using that file in a different way (as your in-box), but the change is invisible to people who send you messages.

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 3.1). 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. After you have read sections 2.4 and 2.5, it should be evident to you why this works.

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 appear with lines through them (i.e., marked as deleted), and the examined/unexamined status of each message will be reflected in the mark field in Laurel's table-of-contents entry.

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 (admittedly tedious) procedure should be followed for each mail file:

1. On Maxc, append the desired mail file, say Archive.txt, to Message.txt:
   @Append Archive.txt Message.txt^CR

2. On the Alto, in Laurel, click RED over **Mail file** and supply the file name "Archive".

3. Select **New mail**. 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 1.) Move any extraneous messages to "Active.mail".

4. **Quit** Laurel and go back to Maxc for the next file. You may delete Archive.txt on Maxc at this point.

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.

6. 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.)

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. If you don't like what you get when you insert text from a displayed message, you will have to replace explicit CRs by spaces before Laurel will reformat the text.

Tabs

Laurel does not presently implement true tabs. Instead, a tab is converted to a fixed amount of white space. Thus, if you use tabs only at the beginning of a line, things will line up. In other cases, you are out of luck.

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 specifier 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 specifier.

File version numbers

Laurel does not support the use of Alto file version numbers. This restriction stems from the underlying implementation of the Mesa run-time system, and is not likely to change. Therefore, mail files and distribution lists should not have version numbers.

Space required for mail files

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 *not* suitable for obtaining hardcopy archives of all your messages, say, to put in your filing cabinet. Because Laurel prints each message on a separate page, using Laurel hardcopy for such archival purposes would produce unwieldy quantities of paper.

If you do wish to maintain a hardcopy archive, we recommend that you use the EmPress subsystem to print your mail file as continuous, unformatted text.

7. Things a casual user doesn't really need to know

Laurel currently executes almost exclusively on your local Alto. Eventually, a large fraction of its facilities will be provided remotely in a transparent fashion, but for now everything except in-box storage is local. This means that all mail files reside on the local Alto disk. 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. Deleting them lengthens the time required to start up Laurel.

If Laurel generates an automatic error report directed to LaurelSupport (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.

In-boxes are *not* stored locally. Laurel assumes that your in-box resides on the server machine indicated in the "Retrieve" line of your Laurel profile, and uses that machine's mail server to read and empty them. Messages sent via the **Deliver** command are forwarded to the mail server specified in the "Send" line of your Laurel profile. Arpanet addresses, specified as "*name* @ *site*" or "*name* at *site*", will work correctly only in messages delivered to Maxc. All this is strictly temporary—eventually, Laurel will understand about mailboxes in multiple places and will introduce its own notions about naming message recipients.

MARKUP

by WILLIAM M. NEWMAN

Markup User's Manual

Table of Contents

1. Introduction

Markup is an Alto program for document illustration. Its basic purpose is to permit you to add illustrations to an existing formatted text document. It can also be used to prepare visual aids, and to manipulate illustrations prepared with other illustration programs, such as Draw.

The purpose of this manual is to explain how to use Markup. Effective use of Markup involves three different kinds of issues. In the first place, there are things you must know *before* you start to use Markup; then there are the individual commands of Markup, and their effects, which you must learn; finally, there are a great many "tricks" that you will find useful if you make extensive use of Markup. This manual is concerned mainly with issues of the first two kinds, those that you will *need* to understand in order to use Markup. The final section of this manual discusses some of the "tricks" that fall into the third category.

2. Things to know before you start

Markup manipulates dots. The commands in Markup are almost all oriented towards helping you to manipulate the *dots* that make up the picture on the screen. With enough patience and skill, you could use Markup to build entire pictures out of individual dots, as if you were a Pointillist painter. On the assumption that you don't have the time for this, Markup gives you more powerful commands for manipulating whole collections of dots. Thus you can with a single command lay down a straight line of dots, or an entire rectangular area. Even after you have used one of these commands, the picture remains a collection of dots, each of which can be modified individually.

Markup treats dots and text separately. Markup knows how to deal with text: it will accept formatted text from Bravo and other such programs, or simple text strings placed with the mouse. Text is, however, treated differently from dots. You use different commands to manipulate the two different types of information.

Markup is for marking up documents. In other words, Markup expects you to provide the initial document for marking up. If you don't have anything with which to start, Markup will provide you with something analogous to a single blank sheet of paper. If you do have a document from which to start, Markup will let you add more pages to it. Basically, however, Markup expects you to provide a document as a starting point.

Markup works with Press files. In the world of the Alto, there are many different sorts of file formats: Bravo files, Sil files, Draw files and plain ordinary text files. Most of these are for use with a specific program—Bravo files with Bravo, Sil files with Sil, and so on. Press format is rather different: it is supposed to be suitable for *all* kinds of program, whether text editors, page composition systems, printers, or illustrators. As time goes on, we expect more and more of our programs to deal solely in terms of Press files. Markup is the first of these.

The implications of the use of Press files are as follows:

1. Any document to be illustrated with Markup must be converted to Press file format. You can tell Bravo to produce a Press file by issuing the File option of the Hardcopy command. Draw, Sil, and some other programs are also capable of producing Press files.
2. Once in Press format, the document will be in the form of separate pages, each corresponding to a page of the final printed document. It will no longer behave like a continuous scroll, as in Bravo.
3. Any illustrated document generated with Markup will also be in Press format, and must be printed by a process that accepts Press files. Examples of systems that will print Press files are the Spruce program for the Dover and the Press program for the SLOT/3100.

3. Use of Markup

This section of the manual starts by explaining how to get hold of a copy of the Markup program, and how to get the program running. It then covers each of the commands of Markup, one by one.

How to obtain Markup

The program you need is called Markup.run, and is to be found on the <Alto> directory on Maxc and on most IFS servers. A copy of Markup.run is usually included with the other programs on the BASIC NON-PROGRAMMER'S DISK. If your disk does not include a copy, transfer it from your local file server by means of the FTP program; for example:

>Ftp Maxc Ret <alto>markup.run^{CR}

How to start Markup

Simply type "Markup", followed by $\underline{^{CR}}$. Markup will ask you for names of the input and output files. If you omit the input file name, and type only $\underline{^{CR}}$, Markup will supply a one-page blank Press file; if you omit the output file name, Markup will assume that you just want to play, and do not want to generate any output for printing. For convenience, you may type both file names on the command line:

>Markup InputFile OutputFile^{CR}.

Markup takes several seconds to get started. Eventually the display screen will go blank, the first page of the input file will appear, and the cursor will change to a small cross. Markup is now ready for your use.

If the input and output file names are the same, Markup will save the original input file on a backup file whose name is the original file name followed by "$". This operation is performed at the time you "quit" from Markup.

The mouse

The mouse is the most important input device to Markup. Its three buttons are used throughout in a consistent manner:

RED button (top or left-hand): use of this button generally results in the *addition* of information to the document.

YELLOW button (middle): this button invariably controls the display of the *command menu* that you use to select your next command.

BLUE button (bottom or right-hand): use of this button generally results in the *removal* of information from the document.

There are exceptions to these rules, which are noted below wherever they occur.

As a minor but important point, it should be understood that the so-called "ball mouse", with the three large buttons arranged from left to right, is much less satisfactory for use with Markup than the older-style mouse with three bar-shaped buttons, arranged from top to bottom. The old-style mouse can be moved much more smoothly than the ball mouse, and this is essential for some of the freehand drawing operations of Markup.

The menu

Whenever you press the YELLOW (middle) button, one row of the command menu appears. You

may move the mouse around with this button held down, and select the menu symbol corresponding to the command you want—the cursor symbol will change as you move from one menu symbol to the next. If you move up or down, more rows of the menu will appear. When you have selected the command you want, release the button, and the menu will disappear.

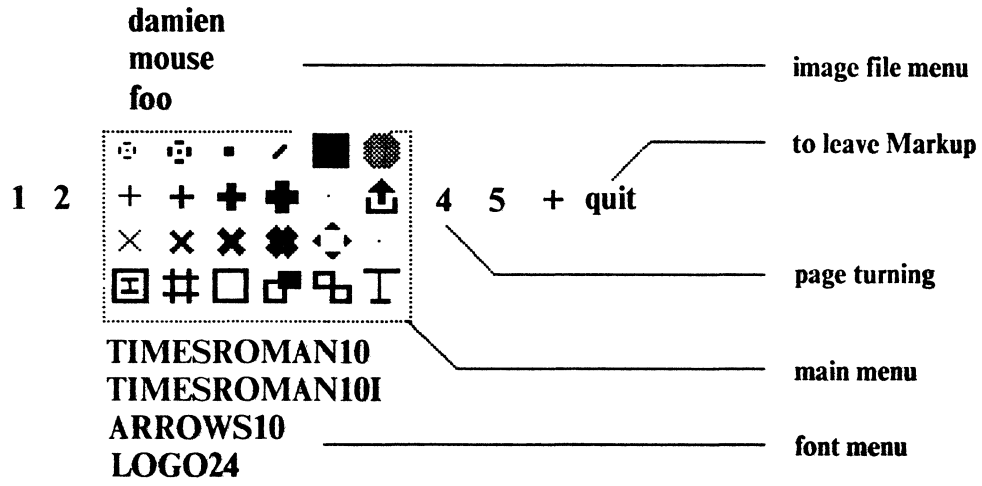
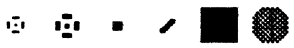


Figure 1. The Markup menu.

The complete menu is shown above. The four rows of six symbols constitute the principal commands of Markup. Below these symbols are the names of the fonts available to you for adding text to your illustrations (the choice of fonts depends on the fonts defined in the input Press file). Above the menu are the names of other Press files containing images that you can add to your illustrations. To the left and right are the numbers of the other pages in the document, and the symbols "+" and "quit" for adding more pages and for leaving the Markup program when you are finished.

The top row of menu symbols: freehand drawing



The top row of the menu contains six different "brushes" for freehand drawing. When you select one of these six, you can "draw" by pressing down on the RED (top or left) button, and holding it down as you move the mouse; dots will appear as long as you hold the button down. The pattern of dots laid down matches the shape of the brush, except in the case of the two left-hand brushes in the top row, which are surrounded by broken circles to make them easier to follow.

The BLUE (bottom or right) button is used for erasing, in an exactly analogous manner to the RED button. As long as the BLUE button is held down, any black dots over which the cursor passes will be changed to white.

The second and third rows: line drawings

The middle two rows of the menu are devoted mainly to line-drawing brushes. When these are selected, use of the RED button dispenses black dots in straight line segments, and the BLUE button does the same with white dots. The upper of the two rows is for use when only horizontal and vertical lines are required; the lower row will also draw lines at 45 degrees. Four different line thicknesses are provided, measuring 1, 2, 4 and 6 screen units across.

In contrast to the freehand drawing brushes, these brushes do not dispense dots in a continuous stream. Instead, a short straight segment is laid down every time the cursor is moved to the next intersection of an invisible *grid.* Thus if you press and release the RED button, nothing will happen;

if you press, and then move the cursor about 1/4 inch, a line segment will appear. As you move the mouse further, a connected sequence of line segments will be laid down, following the path of the cursor as closely as the grid will allow. Note that as long as the button is held down, the corners of connected lines are filled in neatly; if you lift the button between line segments, however, the corners will look messy. To rectify this, you can redraw the line segments that meet at the corner in question.

At the start, Markup sets the line drawing grid to 16x16 Alto screen dots. If you are a novice, you will find this relatively large spacing convenient, but for many purposes it is too coarse—8x8 is a more useful setting. The command for changing the grid setting is described below.

Text

To add text to your picture, simply type it. You need not position the cursor before you type, for Markup expects you to position the text afterwards. While you type, the text will appear wherever the cursor happens to be. During type-in, a^c or BS erases the most recent character typed, and DEL or either the YELLOW or BLUE mouse button erase the whole string. When the text is complete, position the cursor where the center of the text string should go, and press on the RED button. The original text will disappear from the screen, and will reappear, centered about the cursor position.

Text is positioned in this way to the nearest half-grid unit. If you want to position it more accurately than that, do not release the RED button immediately, but hold it down and move the mouse in the direction you would like to move the text. You will have to move through a small "dead zone", and then the text will begin to move, somewhat more slowly than the cursor. If you move the mouse beyond a certain distance, the text will disappear altogether. This is useful, for it allows you to change your mind about inserting the text.

Text may also be erased, using the BLUE button. To do this, you must first select the "text brush", i.e. the T-shaped symbol in the lower right-hand corner of the menu. When you point at a string of text and press on the BLUE button, the text will be erased. It may now be inserted elsewhere with the RED button. Only the text brush and the text-and-image brush (see below) can be used to remove text. It is not always possible to erase text created with other programs, such as Bravo or Draw.

Markup "remembers" the text you erase, or type in and insert, and allows you to insert this text any number of times. In between insertions, you may draw with the freehand and line-drawing brushes, change fonts (see below), and move to other pages of the document.

You may add text in any of the fonts in the font portion of the menu. To select a font, press the YELLOW button and move the mouse down until the font menu appears; lift up when the cursor is within the appropriate font name. You may change the font of a piece of text in this way: erase it, switch fonts, and insert it.

Image areas

Markup provides a number of commands for manipulating rectangular *image areas* on the screen. You can erase an area, insert its contents somewhere else, rotate the image, make a mirror image, and scale it up or down. You may read in images from files on your disk, and may scan images in using a TV camera. You have the option, when erasing or moving an image area, of taking the text within the area too.

In order to use all these commands, you must understand that Markup can "remember" image areas, in much the same way that it "remembers" text. Thus when you erase an area, it doesn't vanish forever, but goes into Markup's memory; when you give the command to insert an image, it

is the image in this memory that gets inserted. The fact that you can't *see* the image in Markup's memory is sometimes inconvenient; but there is only one display on the Alto, and Markup uses it just to show you what's on the page.

Erasing and inserting image areas

There is a small open square symbol in the bottom row of the menu: selection of this symbol allows you to erase and insert image areas. To erase, you must indicate the rectangle to be erased, by tracing out a diagonal of the rectangle. Position the center of the cursor at one corner of the rectangular area, press down with the BLUE button, move to the opposite corner, and lift up. As you move the mouse, the area you will erase is shown by *inverting* its contents, i.e. by turning black dots white and white dots black. Note that the boundaries of the area are constrained to the nearest grid lines.

If, during the erase operation, you change your mind or realize that you started in the wrong place, simply retrace your steps until the black inverted area disappears; you can then lift up with no side-effects. If you should lift up, and erase some priceless gem of an image, there is a special command to put it back exactly where it was (see below, *retrieving from memory*).

To insert an image, select the image area symbol in the menu. Then press down with the RED button, and whatever image is in Markup's memory will be added to the displayed page. Thus an image can be moved by selecting the image area command, erasing it and then inserting it in its new position.

When you press the RED button to insert an image, Markup shows you the boundary of the rectangle where the image will be added. You may keep the RED button down and move this rectangle around until you are satisfied, and then lift up. If you decide you don't want to add the image at all, move the cursor to the edge of the screen, so that the rectangle disappears; you may now release the button without causing insertion.

Markup includes a feature to help you position images exactly where you want them—until this feature was provided, it was very difficult to align parts of an image with the picture on the screen. The feature allows you, before erasing an image area, to copy a small square region of the image into the cursor; this is done by selecting the image area command, positioning the cursor over the area of interest, and pressing the BLUE button down and up without moving the mouse. Now you may erase the area. When you try to insert it with the RED button, the cursor will show you where the area of interest will appear, and you can align it appropriately before lifting the RED button.

Images with dimensions

The third row of the menu includes a symbol made up from four arrow-heads: this symbol can be selected for image manipulation with dimensions. The command operates in all respects exactly like the image-area brush, except that *dimensions* are shown. During erasure, two numbers are displayed near the cursor, indicating the *size* of the area selected for erasing. During insertion, the two numbers show the *displacement* of the image from its original position when erased; this is useful for bar charts, etc.

Retrieving from memory

The symbol at the right-hand end of the second row, depicting an arrow emerging from a bucket, can be selected to retrieve the image, or string of text, from Markup's memory. Whatever was last erased, whether a text string or image area, will be replaced on the screen in its original position, and will be expunged from Markup's memory. This command is useful now and again: for example, to copy an existing text string, you can erase it, insert it at the new position, and retrieve

from memory to put back the original text.

Images and text

At the left-hand end of the bottom row is a symbol made up of the image area and text symbols combined. If this symbol is selected, text and image areas can be manipulated together. This command is very useful for moving entire illustrations.

The image-and-text command works in a similar fashion to the image area command: the BLUE button is used to define the rectangular area to be picked up, and the RED button is used to insert the image and text in the desired position. As in the image area command, a rectangle shows where the information will be inserted; also the same cursor alignment feature is available. Images and text may be moved between pages by erasing them, selecting another page, and inserting them; this is useful for adding existing illustrations to a text document, and applies to pure images and pure text as well as to combinations of the two.

The only unusual feature of this command is its criterion for picking up text. Any text strings placed on the page with the aid of Markup will be picked up provided they lie within the boundary of the overall enclosing rectangle. Formatted text from Bravo, however, is picked up only in units of a paragraph at a time—it is not possible to pick up part of a paragraph and leave the rest. This is sometimes useful, for it means that you can pick up an illustration without accidentally including part of a nearby text paragraph.

Changing the grid

The symbol resembling a number-sign, next to the image-and-text symbol, can be selected in order to change the grid spacing. This command uses the BLUE mouse button for size changes, and the RED to change the grid alignment. Either button, if pressed, will cause lines of dots to appear on the screen showing the horizontal and vertical grid spacing.

To change the grid spacing, press down on the BLUE button. Two numbers will appear near the cursor, showing the horizontal and vertical spacing. As the mouse is moved, the spacing will change. It can be adjusted to as high a value as you like, and to as low as 5 units in each direction, independently in the two directions. Note that 45-degree lines cannot be drawn if the horizontal and vertical spacings are different.

You may wish, after changing the grid spacing, to make sure that lines drawn with the new grid are aligned with those drawn previously. To do this, you may have to change the grid alignment, using the RED button. When the RED button is held down, two numbers show the coordinates of the grid *origin*; you can think of these numbers as defining the position of the "squared paper" on which the grid is drawn. With the aid of these numbers, plus the dots showing the grid itself, it is usually possible to get things lined up.

Inversion and fast erasing

The bottom row includes a command symbol made up of a black box together with an open box. This command provides two functions. With the RED button, image areas may be *inverted*, i.e. black dots turned white and *vice versa*. With the BLUE button, fast erasing can be done, at the expense of retrieval from mistakes—once the dots have gone, they can't be retrieved or inserted elsewhere. If you should ever need to erase a large area rapidly, you will find this command saves you many seconds of waiting. This command operates only on dots, not on text.

Rotation and scaling

Once an image has been erased and stored in Markup's memory, it can be manipulated in a number of ways, using the one remaining command symbol on the bottom row yet to be described—the one consisting of two open boxes. This command differs from all the others in its use of the mouse buttons: both buttons cause the image to be inserted. The RED button is used to scale the image, to rotate it through multiples of 90 degrees, and to make mirror images of it. The BLUE button is used to rotate the image through arbitrary angles.

When you press down on the RED button, a double cross appears, marking the "origin" for scaling and rotation. You may now choose between the various functions available, by moving the cursor in various different directions. As shown below, the upper right quadrant is used for scaling, while the two adjacent quadrants provide mirror-imaging, and the lower-left quadrant provides 180-degree rotation. If the cursor is positioned directly above or below the origin, the image will be inserted rotated 90 degrees anticlockwise, and if the cursor is positioned on the same horizontal line as the origin, the image is rotated 90 degrees clockwise. In every case, a rectangle shows the position where the image will be inserted; in the case of scaling, tick-marks are placed at intervals of half the size of the original.

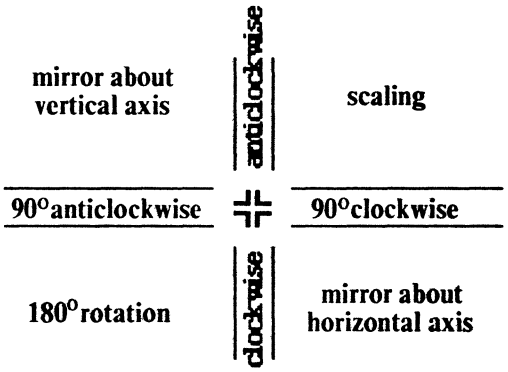


Figure 2. Control of scaling and rotation by quadrant

Pressing down on the BLUE button also displays the double cross origin, and when the mouse moves, three small dots show the positions of the other three corners of the rotated image. Rotation is zero when the cursor is directly to the right of the origin.

In both cases, the image will appear when the mouse button is released. To prevent any insertion, simply return the cursor to the vicinity of the displayed origin and release the button.

Image files

Images may be stored in Markup's memory either by erasing them from the displayed picture, or by reading them in from Press files stored on the disk. The names of the files that you can access in this way are shown above the main menu, and you read them in by moving the cursor so that it coincides with the name of the image you want; when you lift the YELLOW button, the image is placed in Markup's memory, whence it can be inserted, rotated or scaled as described above.

Markup selects images for inclusion in the menu at initialization time, by scanning your disk directory for files with extension ".press", and then checking to see which of these include an image on the first page. A maximum of 20 such images may be included in the menu. You may inhibit the construction of the image file menu by invoking Markup thus:

>Markup/n InputFile OutputFile^{CR}

Turning the page

The numbers to the left and right of the main menu may be selected in order to turn to other pages of the document. Markup's memory is not affected by page turning. The "+" symbol creates a fresh blank page. There is no command for erasing a page, but if all the text and dots on a page are erased, Markup will delete the page from the document when it writes out the output file.

Finishing

When you have finished using Markup, select the word "quit" to the right of the page number display in the menu. Markup will think for a while, and then return you to the Alto Executive.

Camera input

The menu contains a few blank spaces, including two on the third row. However, if you happen to use an Alto with a TV camera attached, a special symbol will appear on this row, depicting a square with a round hole in it. This symbol controls the use of the camera for scanning in images.

Before you select the camera command symbol, you should define a rectangular area of the screen into which to scan the image; do this by erasing an area with the image-area or fast-erase brushes. Then when you select the camera command, the normal screen picture will disappear, and you will see instead the view through the camera lens, with an area of the required size highlighted within a black border. You can move the border around until it encloses the desired image, and then press the RED button to scan the image onto the page and return to the normal display. The mouse may be moved while the RED button is held down, in order to make accurate adjustments to the position of the border.

The camera hardware performs what is known as *thresholding* on the image—dark areas are made black, and light areas white. The resulting image quality is not always perfect. To improve it, you can adjust the *threshold level* while the camera image is visible, by pressing the BLUE button and then moving the cursor up and down: upward movement will darken the image, and downward will lighten it, at a rate proportional to the cursor's horizontal position.

Use of the TV camera unfortunately involves learning how to switch it on, and how to adjust the various knobs on the camera control box. This is best understood by watching someone else do it.

4. Markup Techniques

This section is for serious users of Markup, who want to learn how to produce good-quality illustrated documents with the minimum of effort. The list of techniques described here is by no means exhaustive. After all, Markup is really a "box of tools" for manipulating the image on the screen, and there are no well-defined limits to the utility of these tools. Each tool has an explicit purpose, like line-drawing or area manipulation, but there are generally several other useful things you can do with each tool; for example, you can erase with the line-drawing brushes in order to square off an irregular black area. The more you use Markup, the more techniques and strategies for illustration you will learn.

This section deals with two sorts of technique: those for managing your document as a whole, and those for creating the individual illustrations. Document management techniques are discussed first.

Disk space

The first thing you must understand before trying to illustrate a large document (i.e. a document with more than half a dozen illustrated pages) is that Markup consumes large quantities of disk space. As a rule of thumb, you should make sure there are at least 100 free pages on your disk for *every illustrated page of the document.* Thus if you intend to put illustrations on 12 pages, you should have 1200 free pages before you start. The exact amount you will need depends on the size of the illustrations: if you intend to put just a few tiny images on each page, in the same manner as this manual, then your disk space requirements will be much less.

The reason for all this caution is that Markup cannot be relied upon to recover gracefully if it runs out of disk space, unless it runs out after you have "quit". Whenever you turn to a different page, Markup checks to see if you have 100 or more disk pages available; if you do not, it writes out the pages you have created or modified so far, and quits to the Executive. This is not fool-proof, however, and it is still possible for you to spend three hours illustrating twenty pages, and then run out of disk space while working on the twenty-first, leaving you with no means of recovering your lost work. When you "quit", Markup goes through a cleaning-up operation, combining the new versions of the pages you have created or modified with the original versions of the pages that have not been changed. This, too, consumes disk space in large quantities, but if Markup fills up the disk during this operation, or quits when it finds fewer than 100 disk pages, there is a simple recovery procedure (see below, *Recovery from a full disk*).

For reasons such as these, it is generally worthwhile to split large documents into a number of separate Press files, and combine them only when all illustrations are complete. This is easily done with the PressEdit program (see the Alto Non-Programmer's Guide or the Alto Subsystems Manual). Each separate Press file should include no more than six to ten illustrated pages.

Use of existing illustrations

Documents must often be edited many times before they are ready for distribution. Markup has been designed to allow you to prepare the illustrations for such documents on a separate Press file or files, and then to combine them with the text. You may also use illustrations prepared with other programs, such as Draw, Sil and Flyer.

This technique amounts to the preparation of one or more Press files that form a *portfolio* of illustrations for inclusion in the document. It is generally more convenient if each Press file page contains just one illustration. If, however, you are *sure* that a certain group of two or more illustrations will be placed on the same page, then you will save time by keeping them on one page of the portfolio.

The PressEdit program is again the means whereby the formatted text file (in Press format) is combined with the portfolio file or files. Once the files have been combined, you can commence the rather tedious operation of copying illustrations from the portfolio pages to the appropriate text pages using the image-and-text brush. Make sure that, at the end, nothing is left on the portfolio pages, so that they will be expunged from the file when you write it out.

Use of Draw output

Press files generated with the Draw program may be used as input to Markup. You may define curves with Draw, and then read the resulting Press file into Markup, in order to add image information or text.

Although Draw allows you to add text to the curved pictures you draw, it is often more convenient to omit the text at the Draw stage, and add it with Markup. If you include text in the Draw file,

you will find the resulting picture very slow to edit with Markup, particularly if you mix several fonts.

For faster copying

Markup uses a rather out-of-date method of copying image areas, which means that this operation is sometimes rather slow. In particular, if the grid is not set to a horizontal spacing of 16, with the origin at zero, then images will be inserted very slowly. *Before you start copying illustrations from the portfolio to the text pages, therefore, make sure that the horizontal spacing and grid origin are at 16 and zero.* For ease of vertical alignment, however, you may set the vertical spacing to a smaller value, such as 8, without affecting performance.

Recovery from a full disk

If Markup exhausts disk space during during a page turn or the "quit" operation, it prints a message on the screen telling you what to do. In essence, Markup saves all the new and modified pages of the document on a Press file called MARKUP.SCRATCH, and this allows you to reconstruct the desired output file.

The most difficult part of this operation is to figure out which pages are included in MARKUP.SCRATCH. You *may* use Markup itself to help you here, but with caution, for Markup erases MARKUP.SCRATCH and creates a new version. Therefore you should copy MARKUP.SCRATCH to a file with another name (such as FOO.SCRATCH), and look at this new file with Markup. When you have figured out which pages this file contains, you can use PressEdit to put together the required document out of this and the original file. Before doing all this, you must of course delete some files from your disk, so as create more space.

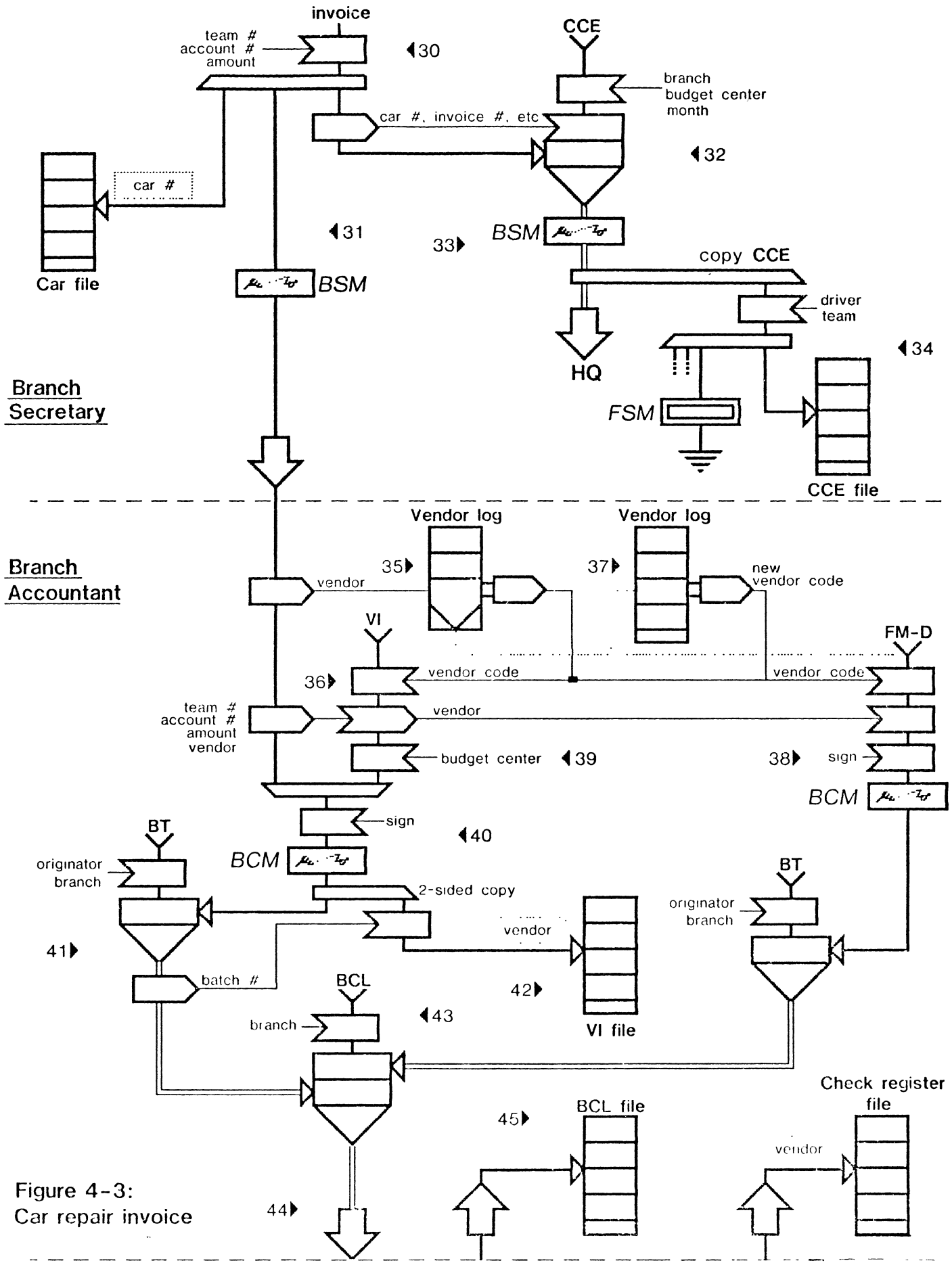
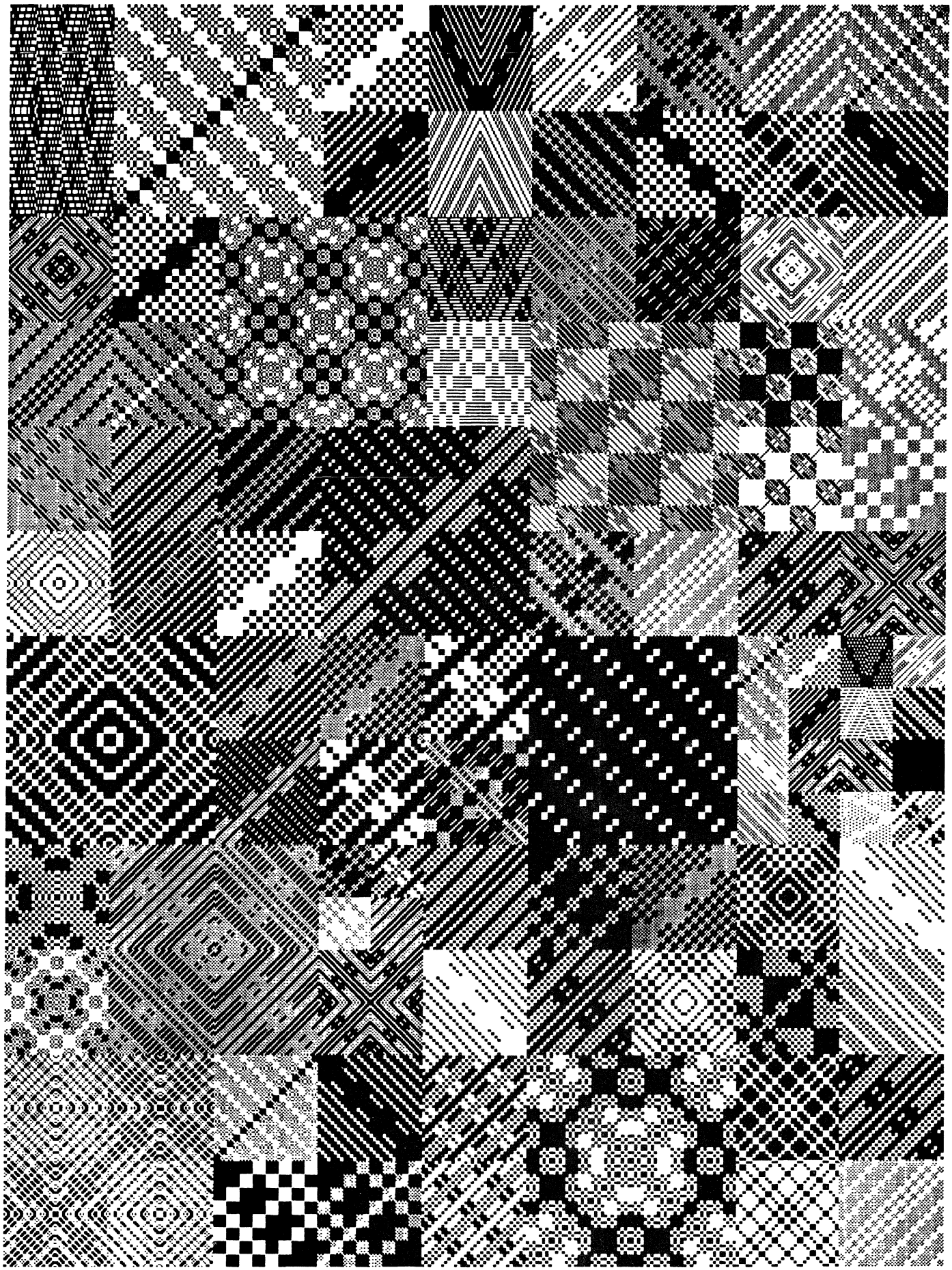


Figure 4-3:
Car repair invoice



DRAW

by PATRICK C. BAUDELAIRE

Draw Manual

Table of Contents

Introduction

DRAW is an interactive illustrator program for creating pictures composed of lines, curves and text captions. All the illustrations in this document have been created with DRAW. Appendix C shows a few interesting examples.

Illustrations are drawn on the Alto screen with the help of the mouse (the reader not familiar with the use of this device should consult Appendix A which explains standard keyboard and mouse terminology).

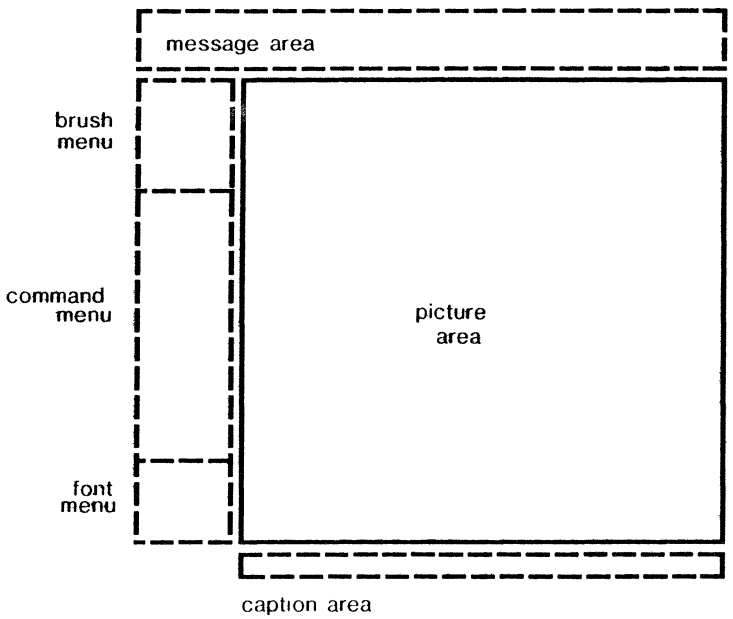
Three types of actions are usually necessary for creating and altering pictures:

- Selecting an operation or *command* to be executed by DRAW. Most commands are selected by clicking in a **command menu area** of the screen. Some commands may also be typed on the keyboard.

- Defining *points* in the picture wich govern the drawing of a line, of a curve, or the modification of the picture. These points are specified with the help of the mouse in the **picture area** of the screen.

- Selecting the *brush* to be used for drawing a line or a curve in the picture, or selecting the *font* to be used for writing a caption. Brush and font are also selected by clicking in the **brush menu area** or the **font menu area** of the screen.

As just mentioned, the DRAW screen is divided into a number of areas:



The **picture area** shows the current state of the illustration being composed. The **menu areas** contain symbols which serve to identify commands, brushes and fonts that may be selected. The **caption area** is used to show typed captions destined to be added to the illustration. Finally, the **message area** is used by DRAW to display informational, error or prompting messages to the user.

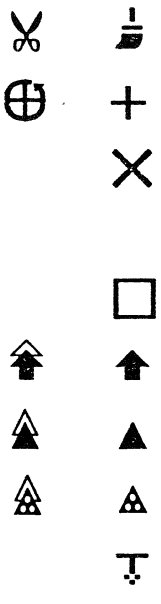
1. Using the menu

The menu is organized in three parts:

Brush selection

The "paint brush" used for drawing lines and curves may be changed by pointing in this area. The size and the shape of the brush may be changed independently. There are four different brush sizes (on the left side of the menu) and four different brush shapes (on the right side of the menu). Brush selection may be done at any time, regardless of the current selected command. This is done by clicking the chosen brush size and chosen brush shape.

The current brush is shown here.



Command selection

All commands involving display interaction are in the central area of the menu. Each is represented by a small ideographic symbol. A command is selected by clicking the corresponding symbol. DRAW indicates which command is selected by changing the shape of the cursor to match the command symbol.

Font selection

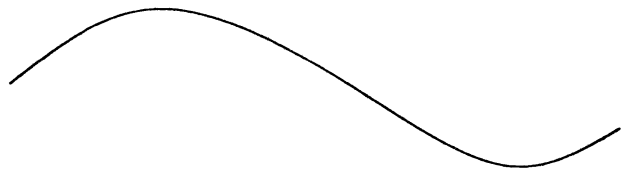
One of four text fonts (numbered from 0 to 3) may be selected by clicking the corresponding font number. The current font is shown on a black background. Font selection may be done at any time, regardless of the current selected command.

When pointing at the menu, you may click any one of the three mouse buttons.

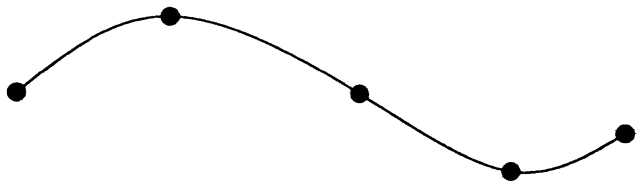
A few additional operations, which do not involve display interaction, are invoked by control keys. They never affect the current selected command.

2. Drawing a curve

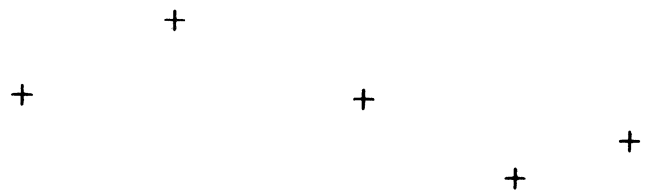
A curve is defined by a number of special points, called *knots*, distributed along its trajectory. The density of knots is generally a function of the curvature of the curve. If the curve is not closed, its end points will be the first and last knots. For instance, a curve looking like this:



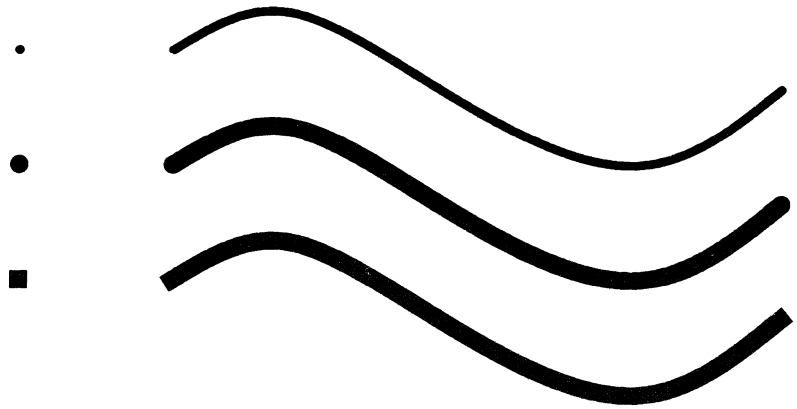
could be defined by these 5 knots (shown as big dots):



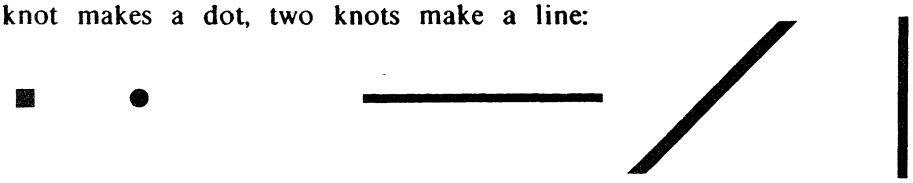
In order to draw a similar curve, first select the command +. Then using the RED button, click down 5 knots at these approximate positions:



Then terminate the drawing command by hitting the ESC key (at the top left corner of the keyboard), or by selecting a new command. The appearance of the curve depends on which brush is currently selected, for instance:

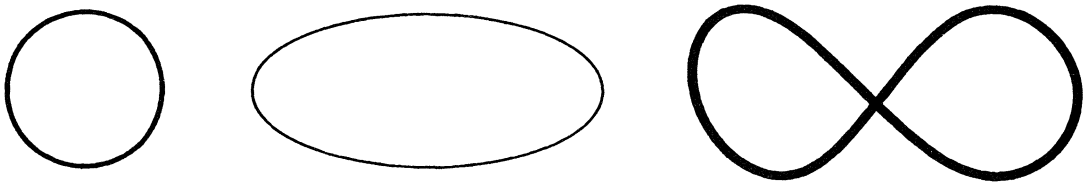


One single knot makes a dot, two knots make a line:



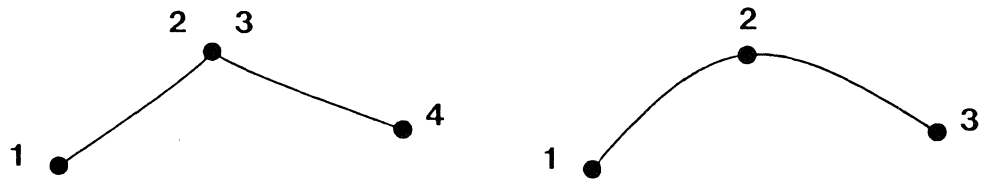
In order to make a smooth *closed* curve, select the menu command ⊕, and then specify the knots by clicking RED:

Of course, you need to define at least three knots. Here are some examples:



When defining knots, you may edit your input: delete the last knot with CTRL A, start over with DEL.

To obtain a smooth regular curve, it is better to distribute the knots *evenly* along the curve. It is not usually necessary to specify many knots unless you want a particularly sinuous curve. For sharp bends specify more knots. In the extreme, two or more overlapping knots will make a cusp.



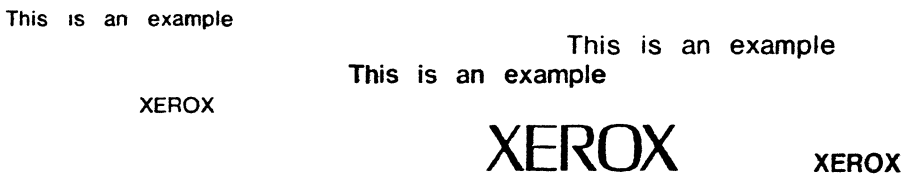
3. Text

3.1 Text input

Text captions are added to an illustration by typing on the keyboard. The caption being typed is displayed in the **caption area** located at the bottom right corner of the display area. Delete characters with the BS key. To start over hit the RETURN key and type again.

To put the text caption into the picture, invoke the text command by either hitting the RETURN key or clicking the caption area itself. As a result, a short section of the text caption becomes the cursor and the text buffer is *closed*: the caption is displayed on a black background. This means that further keyboard input will start a *new* caption.

You may now repeatedly deposit copies of the text caption in the picture area by clicking the RED mouse button. If part of the caption lies outside the picture area, the text is not displayed but is shown as a striped rectangle of the same size. The *very last* deposited copy may be erased with either CTRL A or DEL.



The text font may be changed at any time, using the lower area of the menu. The caption area and cursor are updated accordingly, but text already deposited in the illustration is not affected.

3.2 Text positioning

The truncated portion of the caption which appears as a cursor is used for easier positioning of the caption in the illustration. In order to allow either centering or alignment of a caption, the portion displayed may be either the *beginning*, the *middle* or the *end* of the caption. The effect is accomplished by setting the appropriate **text positioning mode** with CTRL T, as illustated below. The options are: *center*, *left*, *right*, *top* and *bottom*. The initial mode is *center*.

left *center* *right*
bottom
top

3.3 Menu command

If the central section of the caption is blank, or if the characters are too large, it may not be practical to display a portion of the caption as the cursor. Then it is probably preferable to use the T-like pointing symbol . This is done by selecting the text command in the menu area (*rather* than by typing RETURN or by clicking the caption area). Except for the cursor, the effect and operation of this command are as described earlier: point at some position in the illustration and click RED.

The position of the caption relative to the clicked point is a function of the text positioning mode. For example, if the mode is *left*, the clicked position will become the left edge of the caption:

Left positioned caption

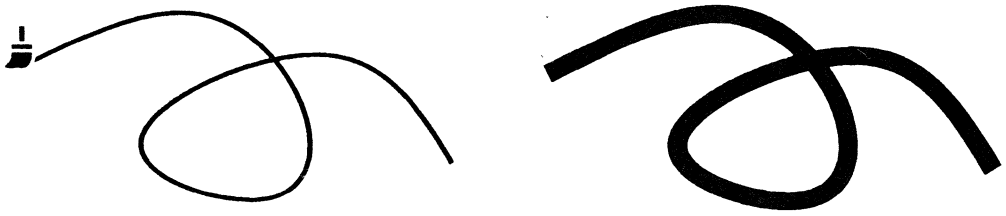
Similarly for other positioning modes:

Centered caption

Right positioned caption

4. Redrawing and rewriting

The shape of curves cannot be changed. However you can redraw a curve using a different brush. Select the menu command with a brush-like symbol . Redraw a curve, using the current brush, by pointing at the curve and clicking RED:



Remember that the current brush may be changed either before or after selecting the command .

With the same command, you may also rewrite a text caption in a different font:

XEROX XEROX

The menu command with the scissor-like ideogram ✂ has a similar function. It is used to make dashed curves and dashed lines. It has no effect on text.

It is also used for the inverse operation.

5. Moving, copying, transforming

You may copy and move lines, curves and text. Lines and curves, but not text, may also be transformed in various ways: rotated, scaled up or down, stretched, etc. You must first select the objects to which these operations are to be applied.

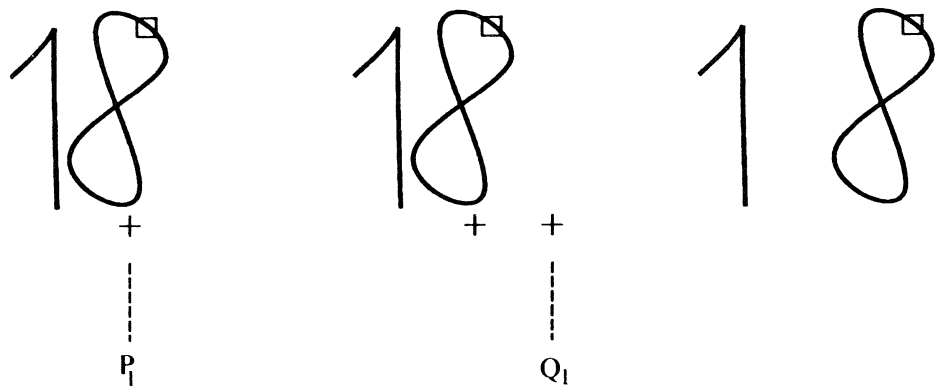
5.1 Selecting

First get from the menu the selection command ☐. All previously selected objects, if any, are unselected. Using RED, click all the objects to be selected. Selected text is shown on a black background. Selected lines and curves exhibit a little square symbol.

If you want to select *everything* in the picture, use CTRL E.

5.2 Translating

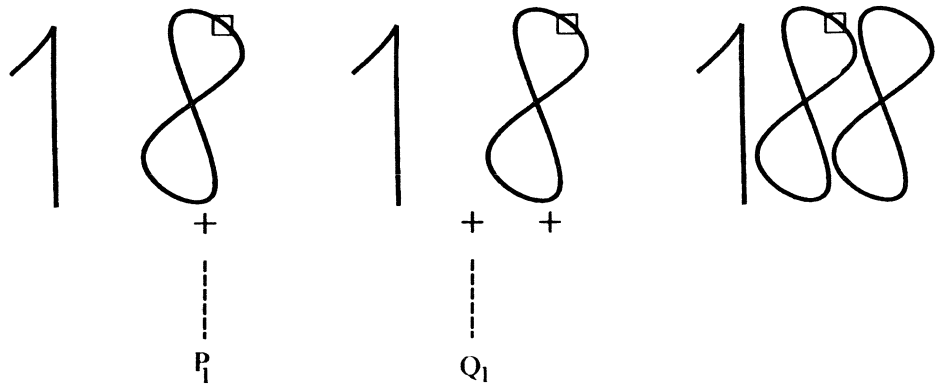
Selected objects may be translated with the command ⬆. Using RED, click *twice* with the tip of the arrow symbol ⬆ to specify a source point P_1 and a target point Q_1. The translated objects remain selected.



The translation is defined uniquely by the length and orientation of the vector P_1Q_1. Therefore the two points may be arbitrarily positioned relative to the selected objects.

5.3 Copying

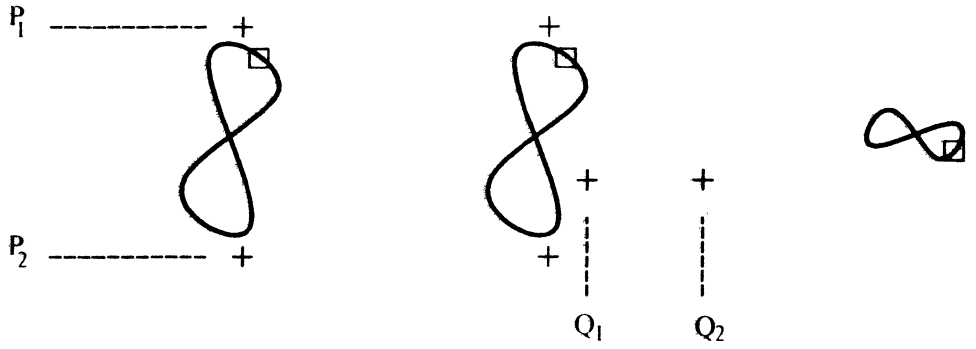
The command ⬆ is for making translated *copies* of the selected objects. It operates in the same fashion:



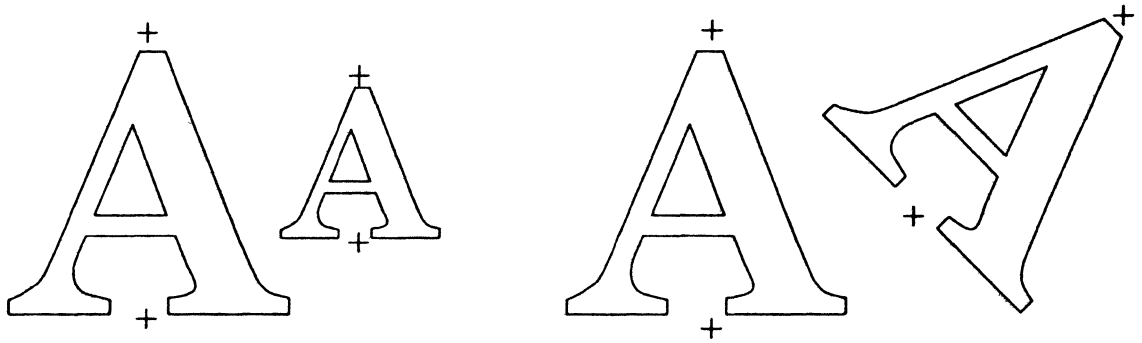
5.4 Transforming: rotation and scaling

The command ▲ is used to apply a *similarity* transformation, i.e. an arbitrary combination of rotation, scaling and translation.

The transformation is specified by *four* points (say P_1, P_2, Q_1 and Q_2). First define some dimension P_1P_2 of the original object, and then define the corresponding dimension Q_1Q_2 of the transformed object. The transformed objects remain selected. Text is treated in a special way: it is simply *repositioned*, by transforming the *left-top* corner of the text caption.



To be more precise, the transformation is defined uniquely by the mapping of the source vector P_1P_2 into the target vector Q_1Q_2. The four points may be arbitrarily positioned. Notice that there will be no rotation if P_1P_2 and Q_1Q_2 are parallel, and no scaling if P_1P_2 and Q_1Q_2 are of equal length. Section *9.1* gives details for specifying such constrained transformations.

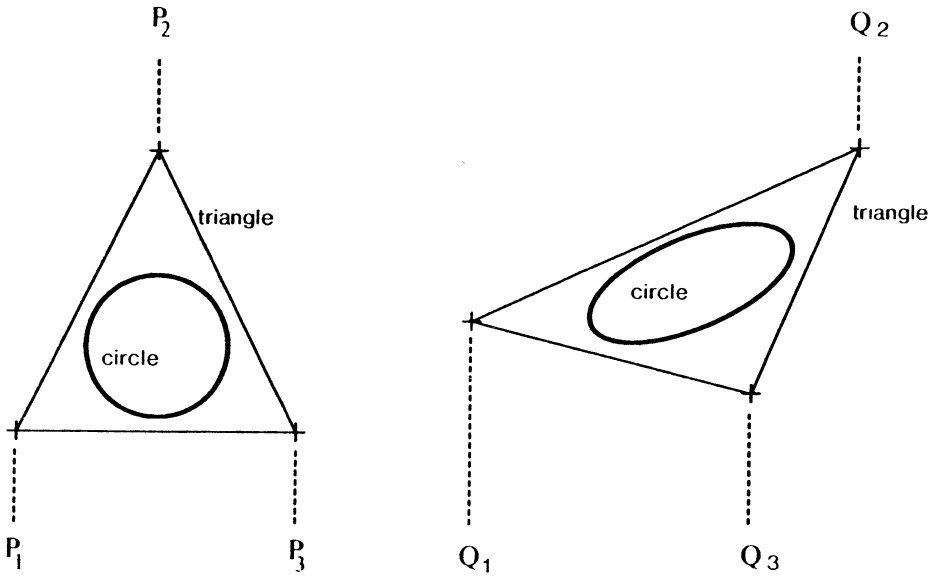


The command ▲ is used in a similar fashion to make transformed *copies* of selected objects.

5.5 Transforming: stretching, slanting, symmetry, etc

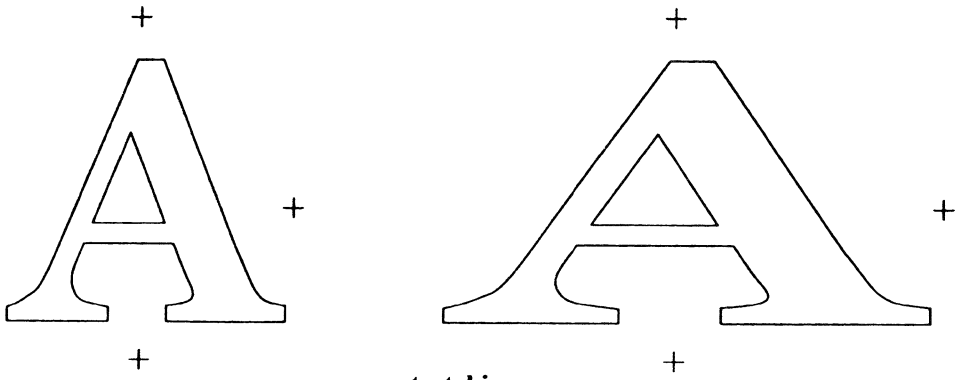
An other pair of commands permit more complex transformations. The command executes a general *affine* transformation, which is an arbitary combination of translation, rotation, scaling, stretching, slanting, symmetry and more. To be precise: an affine transformation is a linear transformation which preserves parallelism; a linear transformation preserves colinearity of points; non-affine linear transformations include perspective projections.

The transformation is specified by *six* points (say, in the order of input: P_1, P_2, P_3, Q_1, Q_2 and Q_3). It is defined by the mapping of the source triangle $P_1P_2P_3$ into the target triangle $Q_1Q_2Q_3$, as illustrated here:

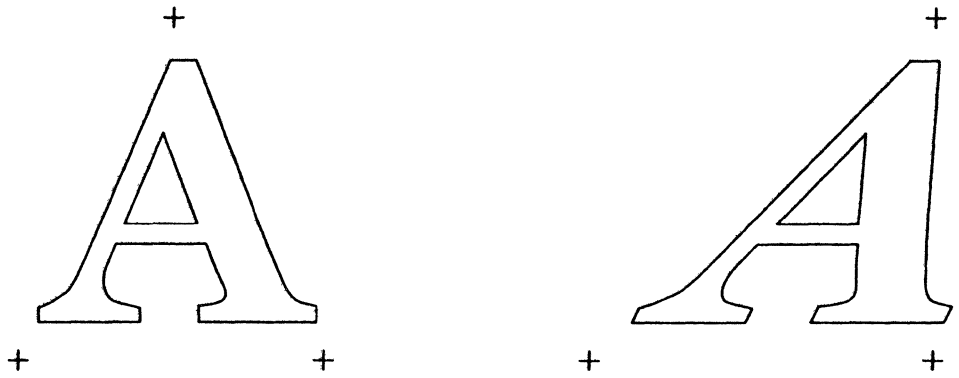


As shown above, text is treated in a simple way: it is merely repositioned. Like the 4-point transformation (▲ and ▲), the 6-point transformation comes in two versions: ⧋ (transform) and ⧋ (copy and transform).

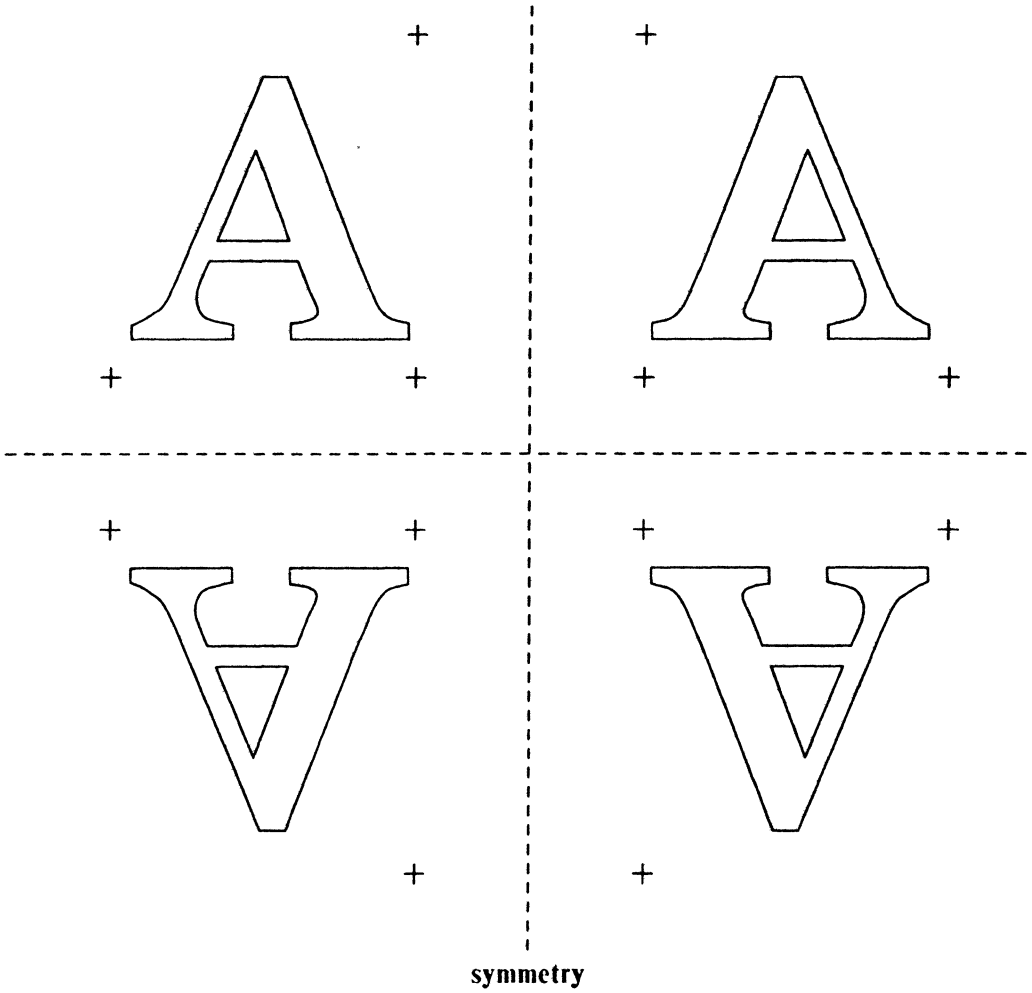
For best results, $P_1P_2P_3$ and $Q_1Q_2Q_3$ should be real triangles, i.e the three vertices should not be colinear. Particular effects are obtained by carefully choosing the shapes and the sizes of the two triangles (see section *9.1*). Here are some examples:



stretching



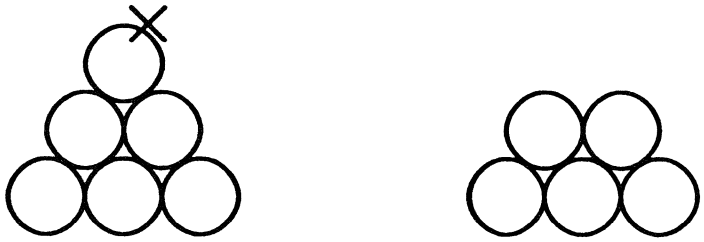
slanting



symmetry

6. Deleting

To delete lines, curves and text, select the **menu command** ✕. Then, using RED, click at the objects to be deleted.



The keyboard command CTRL D will delete selected objects. Therefore, CTRL E followed by CTRL D will delete the whole picture.

7. Refreshing

After deleting an object, the screen may not accurately represent the illustration, and will need to be *refreshed.* The inaccuracies result whenever one of several overlapping objects is deleted: a portion of an other object may also be erased. These effects do not affect the illustration itself, but only its current representation on the display. A new refreshed picture may always be obtained by hitting the TAB key. This is a relatively fast operation which, like any other keyboard command, may be executed at any time.

8. Undoing

There is a limited "undo" command for operations which are destructive or simply not easily invertible: *delete, translate* and *transform.* The command is CTRL U. It will undo the effects of the following commands:

CTRL D ✕ ⬆ ▲ ⧋

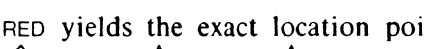
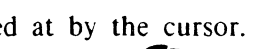
You may undo up to several levels of *delete, translate* and *transform* operations. The depth is a runtime variable which depends on availability of free storage.

9. Use of the mouse buttons

In the previous sections, only the use of the RED button has been described. Although all mouse buttons are equivalent in the menu area, they have various interpretations when used in the picture area. The mouse buttons are used in one of *two modes*, depending on the command.

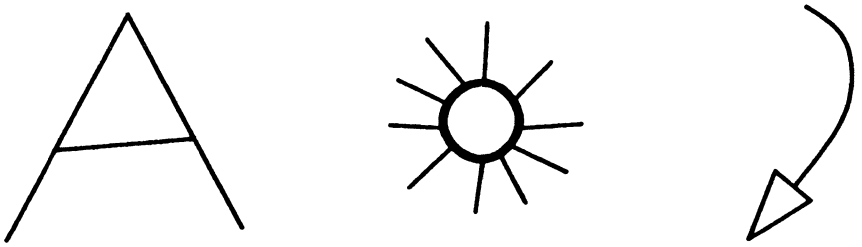
9.1 Defining a point

With the following commands:  , the mouse is used for **defining a point** in the picture area, namely a knot of a curve, the position of a text caption, or the geometric parameters of a translation or a transformation. The point is defined by **clicking** one of the mouse buttons. Each button has a different function:

RED yields the exact location pointed at by the cursor. Click with the tip of  , the center of  , the bottom of  .

YELLOW yields a point *only* if the cursor is in the vicinity of an object. Then it will be the nearest point on the nearest curve, or the center of the nearest caption. The main applications are:

connecting curves: end knots of a curve or a line may be positioned on some other line or curve

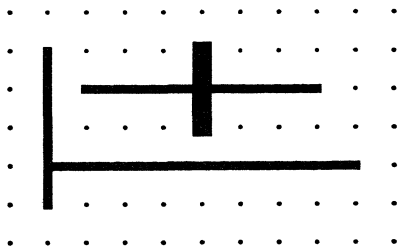


positioning arrow heads (Appendix B): using text positioning mode *center*, arrow head characters may be positioned at the end point of a curve or a line

constrained translation: in order to do a translation precisely in a given direction (e.g. horizontally or vertically), construct a temporary line along this direction, and specify the translation points *on this line*

BLUE is used to position a point on a *grid*. The grid which may be displayed and erased with the keyboard command CTRL G. Useful applications are:

vertical and horizontal lines



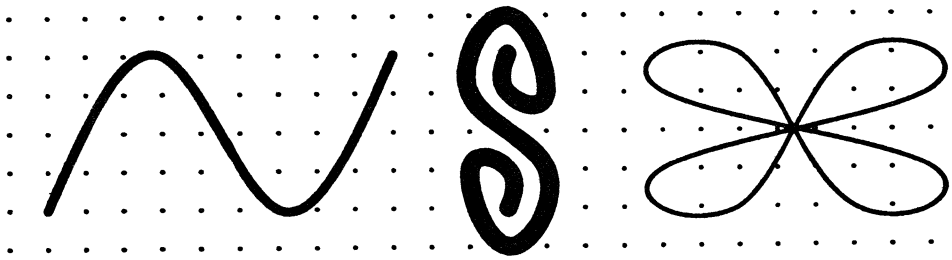
text alignment

left justified text

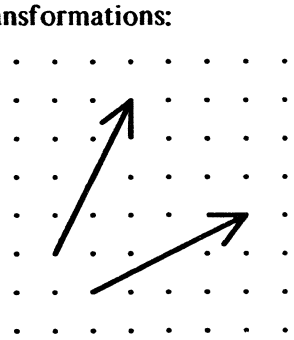
centered text

right justified text

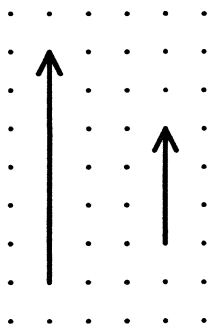
symmetric curves



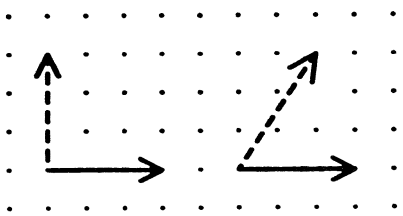
particular transformations:



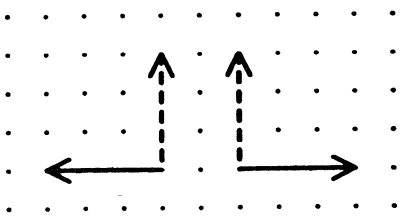
rotation



scaling



slanting

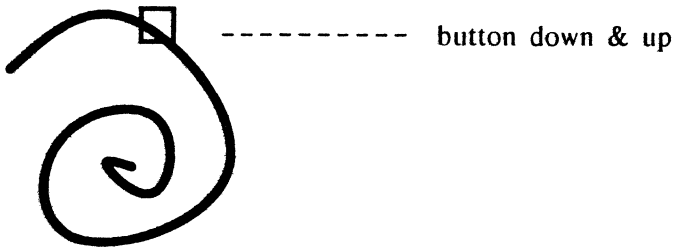


symmetry

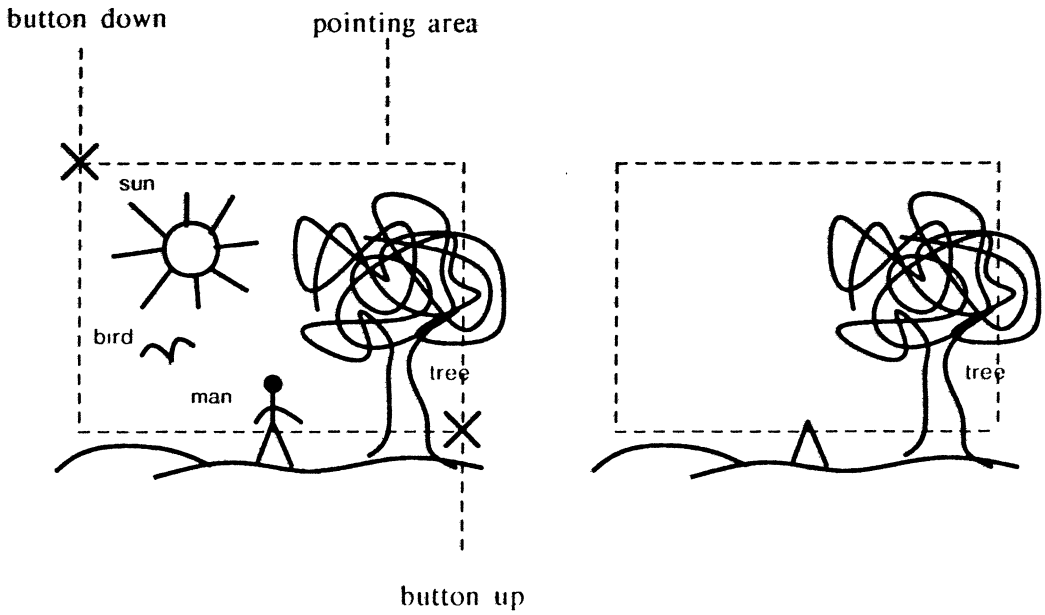
9.2 Pointing

With the commands  , the mouse is used for **pointing** at lines, curves and text captions. All buttons are equivalent. They may be used in two different ways, for designating either *one single object* or *several objects* enclosed inside some rectangular area.

Pointing at *one single object* is simply done by **clicking**, that is to say pointing in the vicinity of the object and quickly depressing and releasing the button.



Designating *several objects* may be done by **bounding** a rectangular area: point at one of its corners, depress a mouse button, move the cursor to the opposite corner and then release the button. This operation designates all the objects which are *totally enclosed* within the rectangular area. The trajectory of the cursor is irrelevant. There is no visual feedback during the bounding operation, but this does not seem to be a major hindrance.



10. Saving and retrieving pictures

DRAW uses its own format for picture files. The recommended file name extension is DRAW.

Two keyboard commands are provided for reading and writing picture files:

CTRL W, followed by a file name, terminated by ESC or RETURN, writes the *whole* picture onto the file. So:

CTRL W PICTURE.DRAW RETURN

CTRL R, followed by a file name, terminated by ESC or RETURN, reads in a picture from the file and *adds* it to the current picture.

Thus a file "notebook" of elementary pictures may be constituted and utilized for composing complex pictures from simpler elements.

The DRAW file format is also accepted by the spline editor FRED (see documentation on <GR-DOCS>FRED.EARS). Therefore FRED may also be used to create pictures to be incorporated in DRAW illustrations. However FRED ignores text and curve brushes, and need only be used for making complicated curves requiring careful editing.

11. Printing

DRAW pictures may also be printed and merged into documents. For that purpose, DRAW outputs one single page PRESS file (the recommended file name extention is PRESS). The command is CTRL P. For instance type:

CTRL P PICTURE.PRESS RETURN

12. Fonts

Fonts to be used with DRAW *must* satisfy standard naming conventions (e.g. HELVETICA12B). Furthermore, for better printing results you should use coordinated Alto/EARS fonts.

DRAW currently uses a default font set which is read in when the program is started:

HELVETICA12 (standard size)
HELVETICA12B (bold)
HELVETICA8 (small size)
ARROWS10 (see Appendix B)

You may change fonts with the command CTRL F, followed by a font number (0 to 3), followed by a font name. For instance, change font 3 with:

CTRL F 3 LOGO24.AL RETURN

You may also release a font and reclaim the corresponding storage space, by omitting the file name:

CTRL F 3 RETURN

13. On-line documentation

DRAW comes with a 13 page on-line manual (an abstracted version of the present documentation). The manual is activated and disactivated with CTRL ?. Consecutive pages are obtained by hitting the LF key.

When viewing the manual, the current picture is automatically saved and restored. The first page of the manual is a packed summary of all the commands. Therefore you may take a quick look-- so to speak-- at this summary at any time, without losing your picture. A printable version of the summary may be found on <ALTODOCS>DRAW-SUMMARY.EARS.

14. How to run DRAW

Get the file package <ALTO>DRAW.DM and unpack it with the Alto program LOAD. It contains the following files:

DRAW.RUN: this is the illustrator,

MANUAL1.DRAW to MANUAL13.DRAW: 13 pages of on-line manual,

default fonts (see above),

CIRCLE.DRAW: contains a carefully done 8-knot closed curve which impersonates a circle very well (look at it in sections *2* and *5.5*).

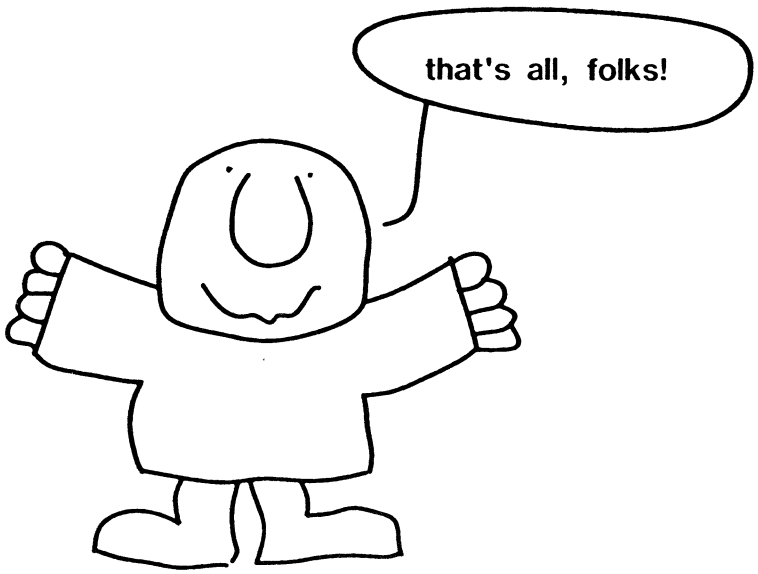
If you only need the program DRAW.RUN, the current version may simply be obtained from the directory <ALTO>.

To run the program, type:

DRAW RETURN

Because of a current shortage of memory storage, it may be good practice to release the fonts you will not be using (see section *12*).

Terminate with CTRL Q.



15. Summary of commands

15.1 Menu commands

The most commonly used menu commands may also be invoked from the keyboard. This allows faster interaction for the experienced user.

menu command	equivalent keyboard command	action
	ESC	draw curves & lines
		draw closed curves & lines
		redraw curves & lines, rewrite text
		dash/undash curves & lines
	CTRL X	delete
	CTRL S	select
	CTRL Z	translate
	CTRL C	copy & translate
		transform (4-point)
		copy & transform (4-point)
		transform (6-point)
		copy & transform (6-point)
		position text
text buffer	RETURN	position text

15.2 Keyboard commands

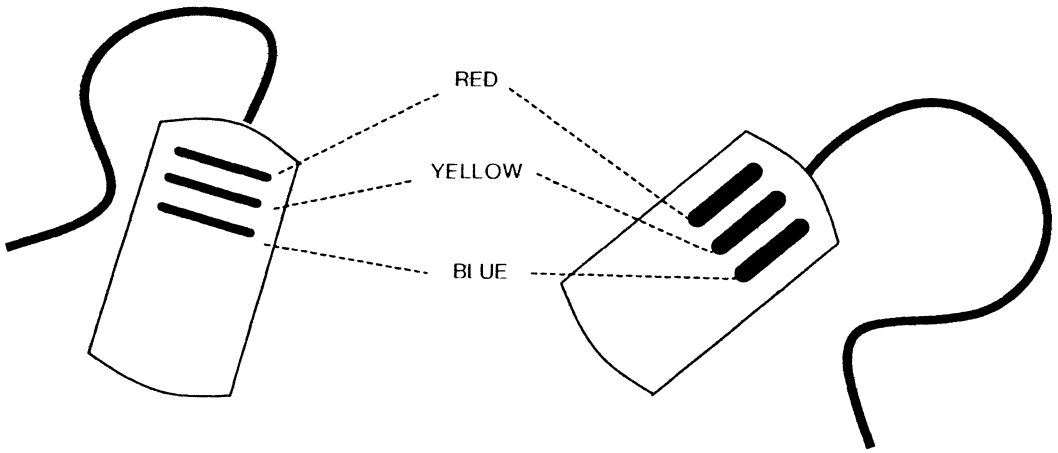
CTRL A	delete last mouse input (applies to: )
DEL	start over (applies to: )
BS	delete characters in caption area
CTRL T	change text positioning mode
TAB	refresh the picture
CTRL G	display/erase the grid
CTRL D	delete current selection
CTRL E	select everything
CTRL U	undo (applies to: CTRL D )
CTRL W	write a picture file (DRAW format)
CTRL R	read a picture file (DRAW format)
CTRL P	write a PRESS file
CTRL F *n*	change font *n*
CTRL ?	enable/disable on-line manual
LF	get next manual page
CTRL Q	quit

Appendix A

Mouse and keyboard terminology

The mouse

The three mouse buttons on a mouse are labeled RED, YELLOW and BLUE. The physical arrangement of the buttons depends on the model:



The most common interaction with the mouse is the **click**: the mouse is first moved over the table until the cursor, which follows the mouse movement, is positioned at the desired spot on the screen; then a button is depressed and raised *in one motion.* Commands described in this manual usually call for clicking a particular mouse button, for instance: "point at a screen position and click RED".

The mouse may also be used to **bound** a rectangular area which the user wishes to define: first position the mouse at one of the corners of the rectangular area and depress a mouse button, then move the cursor to the opposite corner and release the depressed mouse button. The trajectory of the cursor between the two corners of the rectangle thus defined is unimportant.

The keyboard

In this manual, keyboard keys are given in underlined small size capitals, e.g. Q. The special dark colored keys are identified by their keyboard label:

ESC, TAB, CTRL, SHIFT, LF, DEL, BS, RETURN

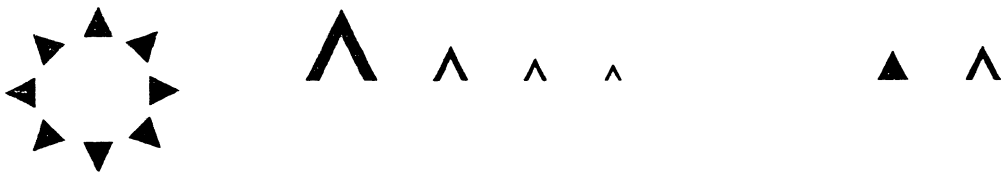
The CTRL and SHIFT keys are used similarly to the shift key on a typewriter. For instance, CTRL A means: depress the CTRL key; depress and release the A key; then release the CTRL key.

Appendix B

Arrow head font

There is a special font of arrow heads for use with various illustrator programs (MARKUP, DRAW, SIL). The Alto and EARS versions are: <ALTOFONTS>ARROWS10.AL and <FONTS>ARROWS10.EP.

The arrow heads come in 8 directions, 4 sizes and 2 styles:



The keyboard mapping is as follows:

- the arrow heads occupy the left 8 keys of each row;
- *direction* = counterclockwise, from left to right:

- *size* = row:

top row

bottom row

- *style* = lower case & upper case:

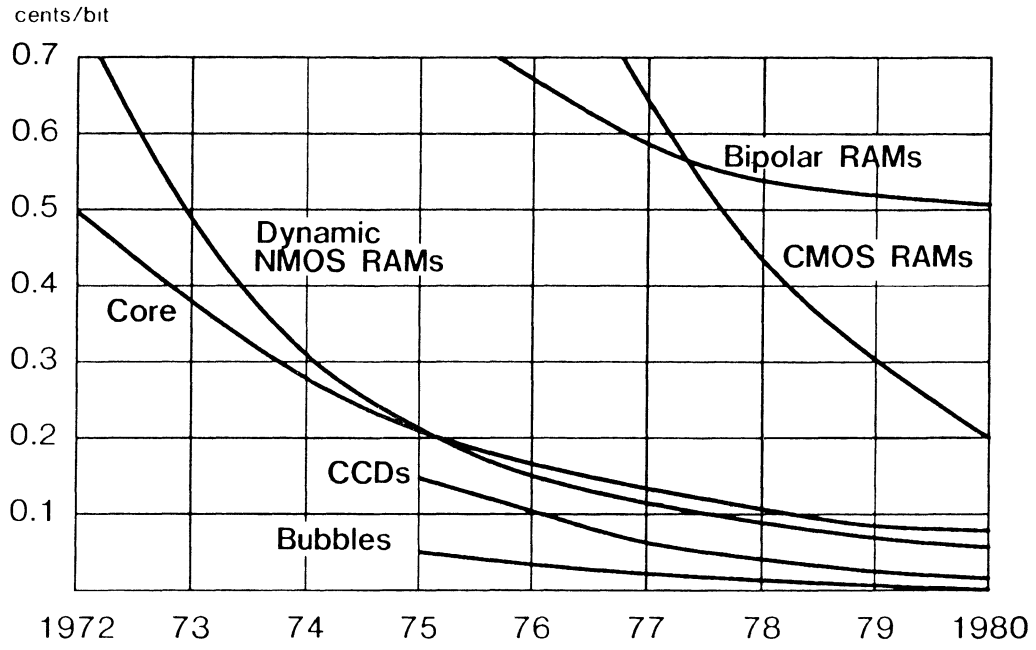
"SHIFT ▶" is

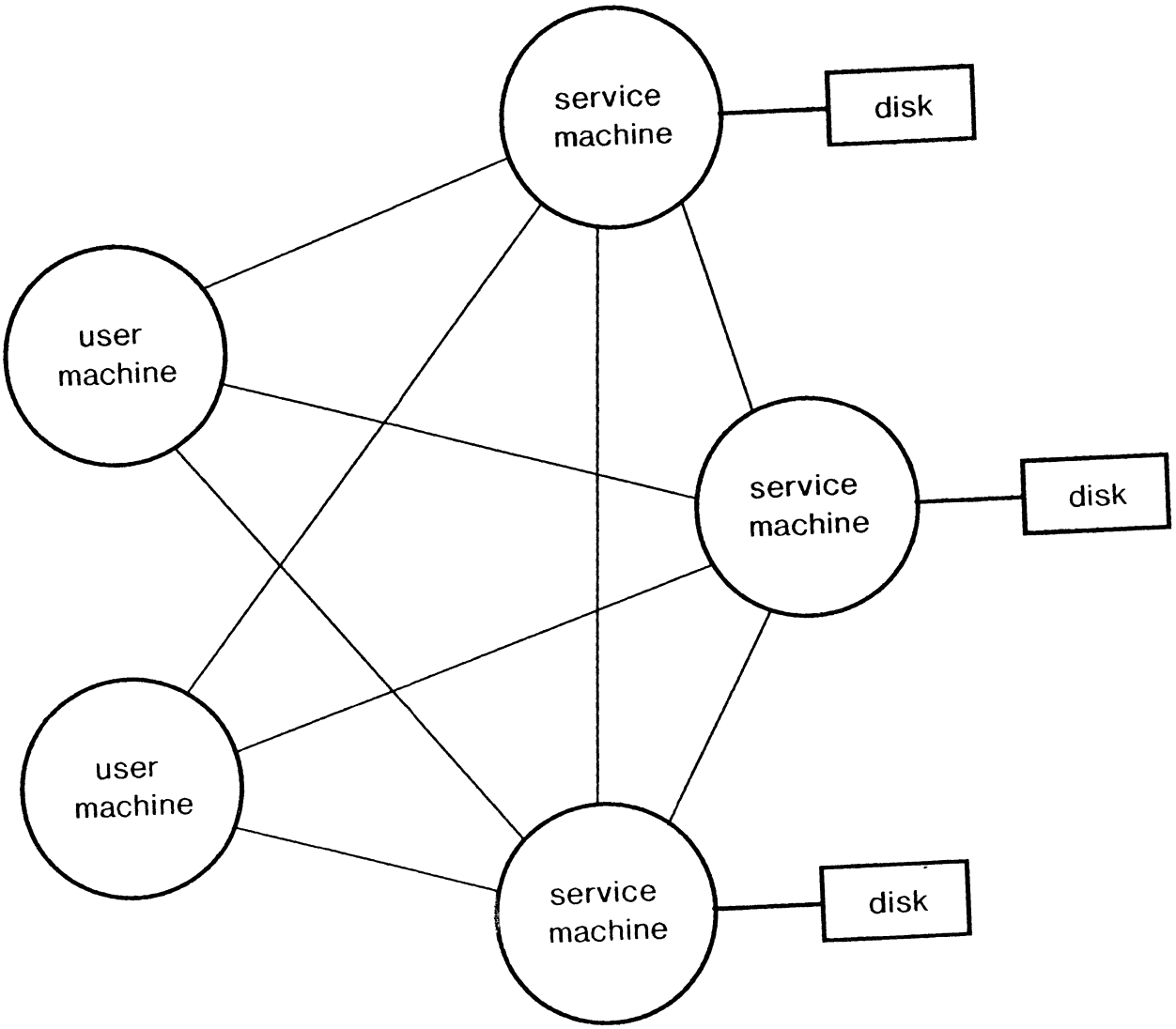
- reference key:

"v" is ▼ (however, "<" is ∧ ...)

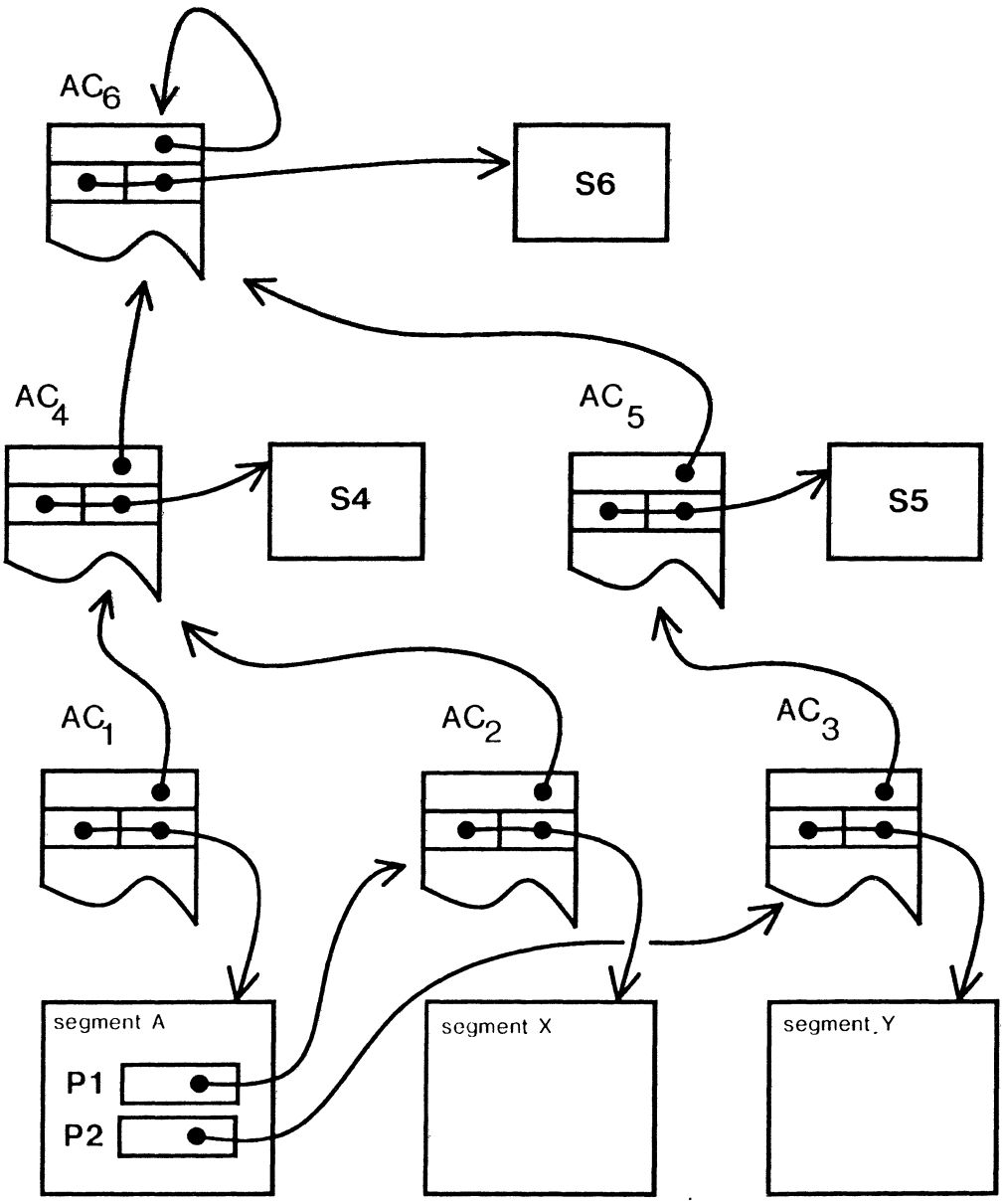
Appendix C

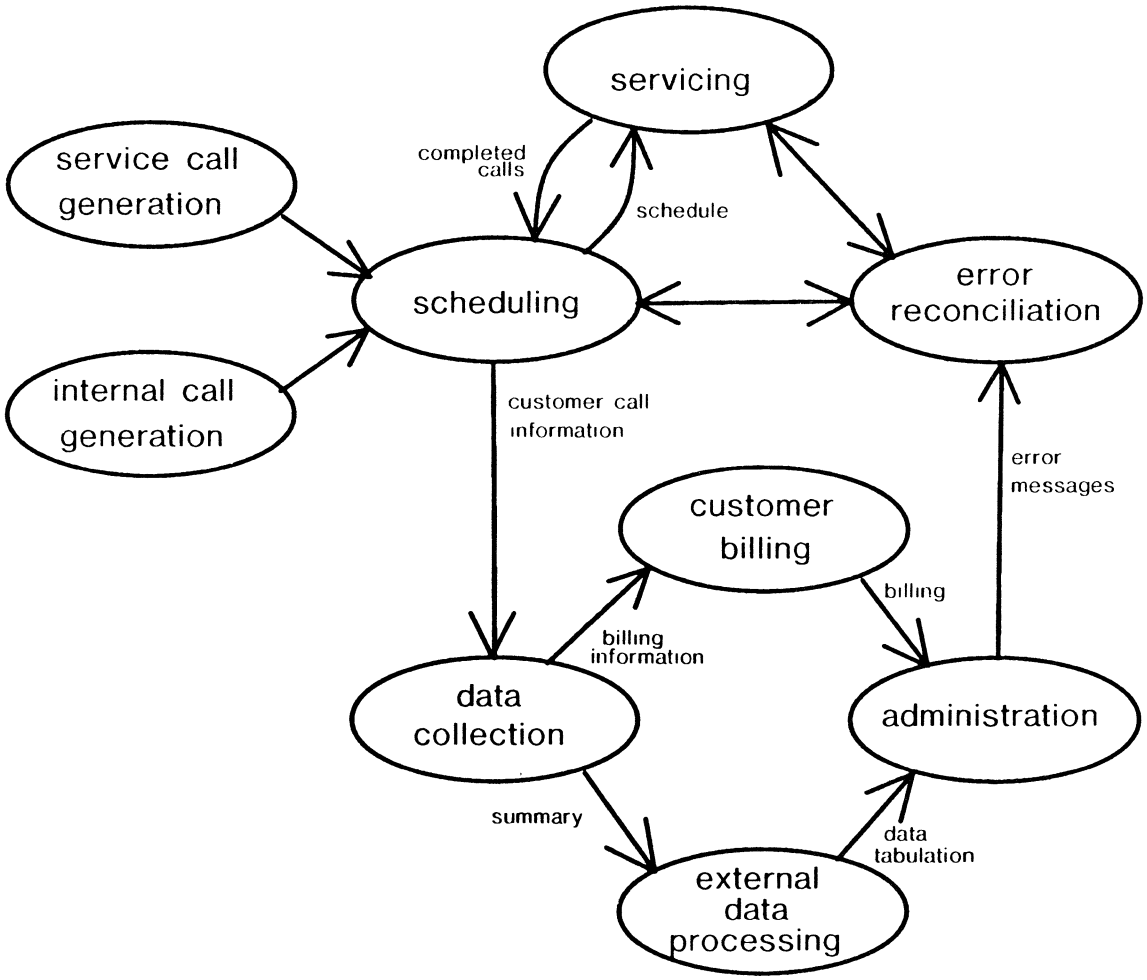
Examples



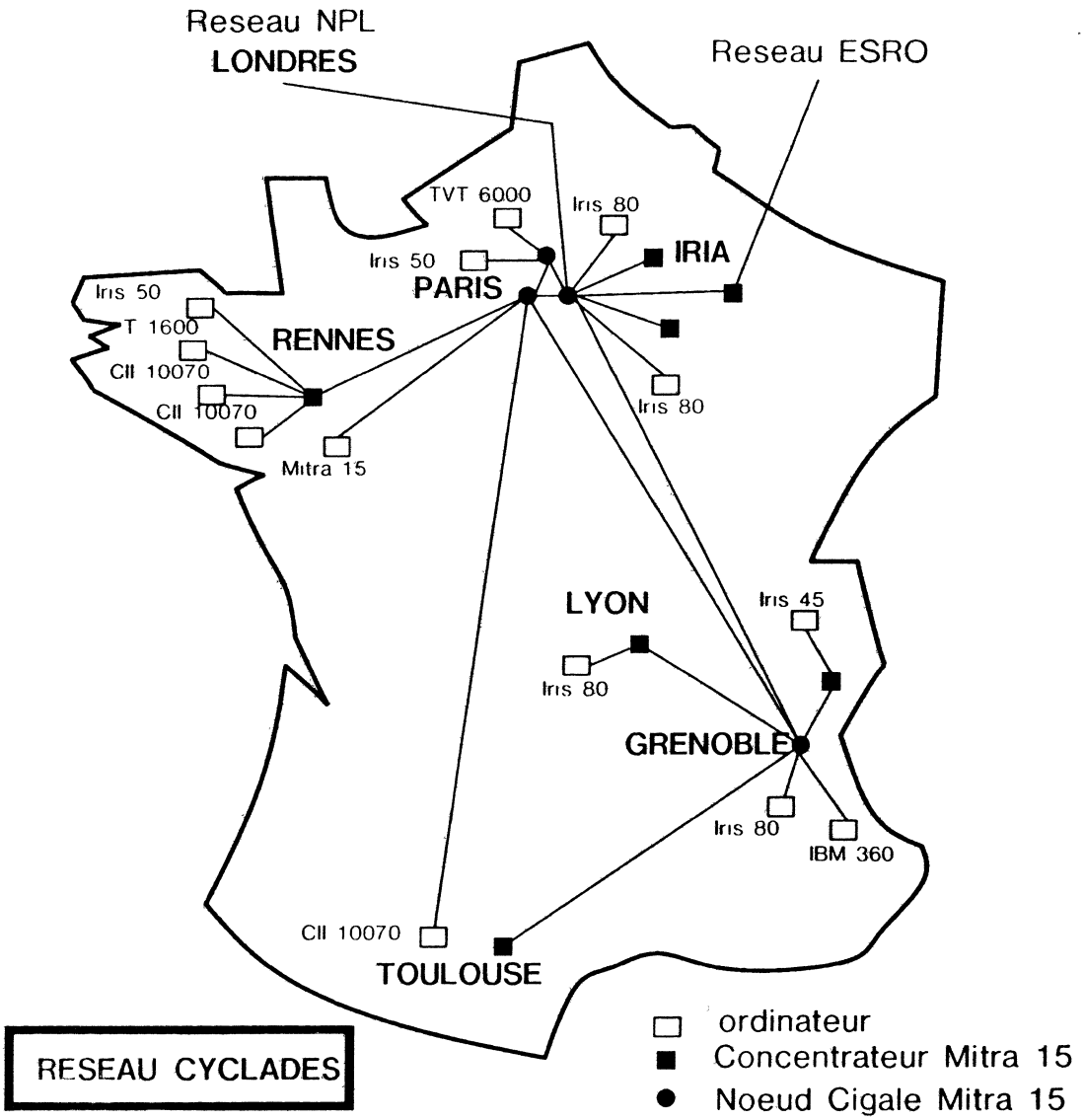


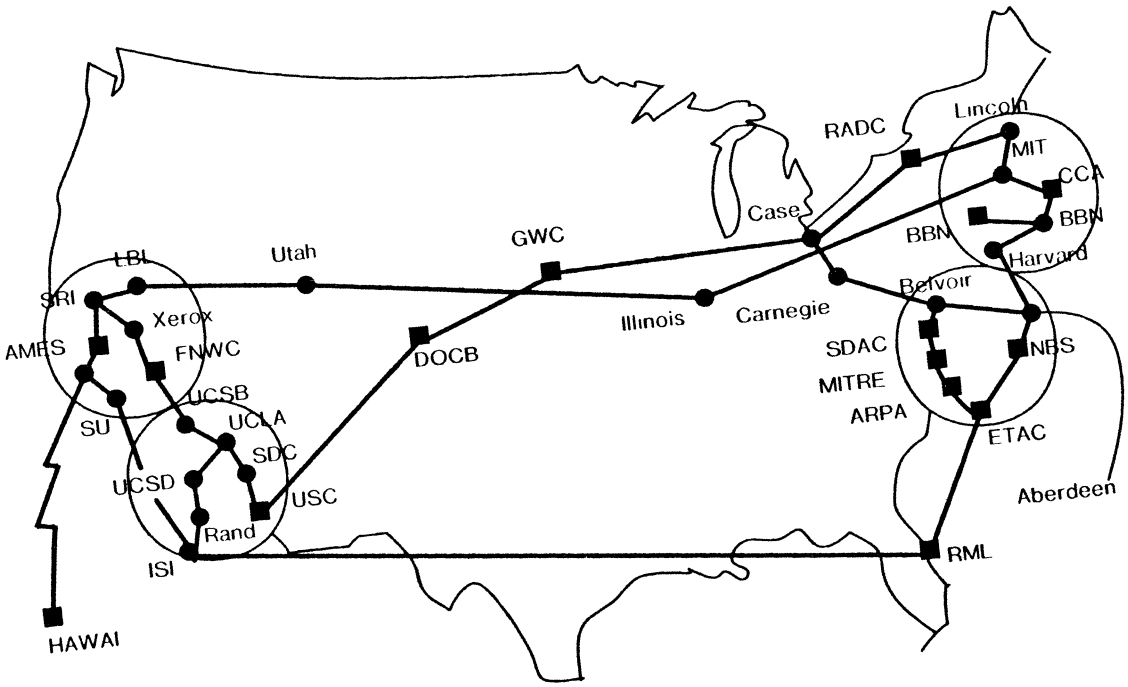
SYSTEM CONFIGURATION


AC6
S6
AC4
S4
AC5
S5
AC1
AC2
AC3
segment A
P1
P2
segment X
segment Y



INFORMATION FLOW DIAGRAM


Reseau NPL
LONDRES
Reseau ESRO
TVT 6000
Iris 80
Iris 50
IRIA
PARIS
Iris 50
T 1600
RENNES
CII 10070
CII 10070
Mitra 15
Iris 80
LYON
Iris 45
Iris 80
GRENOBLE
Iris 80
IBM 360
CII 10070
TOULOUSE
RESEAU CYCLADES
ordinateur
Concentrateur Mitra 15
Noeud Cigale Mitra 15



ARPA NETWORK
(may 1973)

XEROX

Telecommunications Guide & Directory

Appendix D

Recent changes

This section describes some features that have been added to DRAW since the DRAW manual was last revised. Some of them are experimental and may not remain in future versions of DRAW.

Freehand drawing

Lines and curves can be drawn freehand, as an alternative to specifying line end-points or knots. To go into freehand drawing mode, pick up with the cursor the brush symbol shown in the menu inside a small square. Then draw on the screen in the usual way: press a mouse button, move the mouse around, and release the mouse button. The trajectory of the brush-cursor is painted on the screen as you move along. But when you have finished, the trajectory is erased and *replaced* by a smooth curve fitted through knots that have been automatically selected along the trajectory. Some smoothing of the trajectory is done in order to eliminate irregularities (due mainly to the mouse). The curve now behaves like any curve.

DRAW tries to help you in two ways:

If the trajectory that you draw seems to have little curvature, it will be replaced by a *straight line* rather than a curve.

If you use the grid button (BLUE), DRAW will force the trajectory (and thus the resulting line or curve) to *start* and *end* on a grid point.

These two features should help in drawing horizontal and vertical lines on the grid mesh: boxes, rectangles, diagrams, etc.

Color

DRAW permits you to introduce color into pictures that are to be printed on a color printer. Curves, lines, and text can be in black or in one of six basic colors: yellow, magenta, cyan, red, green, and violet (actually a dark blue).

Eight commands in the menu pertain to color. Seven of them are tags representing available colors, each identified by a single letter (B, Y, M, C, R, G, V) inside a small circle. The color commands are used in two ways. Simply clicking a color tag selects the current color in the same way the current brush or current font are selected; all objects that you draw subsequently will be in that color. You can also pick a color tag from the menu and then click RED over each object you want to assign that color.

If you click the COLOR OFF menu command, it will change to COLOR ON, and every object in the picture will have a tag attached to it showing its color.

Command line parameters

You can specify a number of parameters in the command line that you use to invoke DRAW. Each parameter consists of a decimal number, followed by "/", followed by a letter. Default values are used when the parameters are omitted or if the values you supply seem unreasonable.

Parameter	*Specifies*
n/S	The maximum number of *splines* (curves and lines) that a picture can contain. The default is 200.
n/K	The maximum number of *knots* a curve can have. The default is 100.
n/T	The maximum number of *text* captions that a picture can contain. The default is 100.
n/G	The *grid* spacing, in Alto screen units. The default is 16.
n/D	The length, in screen units, of line segments in *dashed* curves and lines. The default is 16.
n/O	The length, in screen units, of *omitted* line segments in dashed curves and lines. The default is 8.

Example:

```
>Draw 250/S 50/K 8/G^CR
```

The user profile

Your user profile (file User.cm) may contain a DRAW section supplying settings for DRAW. First, you may include up to four *font definitions* that determine the fonts available in the menu for text captions. A font definition has the form

FONT:*n font-name*

where *n* is the font number (0 to 3). If you do put any font definitions in your user profile, then *only* those fonts are used by DRAW; if you *don't* include any font definitions, DRAW uses a standard set consisting of Helvetica12, Helvetica12B, Helvetica8, and Arrows10.

Second, you can define the actual thickness of the four sizes of brushes DRAW uses to paint lines and curves. A thickness definition has the form

LINEWIDTH:*n width*

where *n* is a number identifying the brush (0 to 3 from top to bottom in the menu) and *width* is half the thickness of the line in micas (one mica is .001 cm or 1/2540 inch). The default thicknesses for brushes 0 to 3 are 16, 32, 64, and 128 micas.

An example of a reasonable user profile section for DRAW might be:

```
[DRAW]
FONT:0 Helvetica12
FONT:1 Helvetica18
LINEWIDTH:3 200
```

FTP

by DAVID R. BOGGS
EDWARD A. TAFT

FTP Reference Manual

Table of Contents

1. Introduction

FTP is a Pup-based File Transfer Program for moving files to and from an Alto file system. The program has three main parts:

1. An FTP Server, which listens for file transfer requests from other hosts,
2. An FTP User, which initiates file transfers under control of either the keyboard or the command line, and
3. A User Telnet for logging into a remote host using the Pup Telnet protocol.

1.1 Concepts and terminology

Tranferring a file from one machine (or *host*) to another over a network requires the active cooperation of programs on both machines. In a typical scenario for file transfer, a human user (or a program acting on his behalf) invokes a program called an *FTP User* and directs it to establish contact with an *FTP Server* program on another machine. Once contact has been established, the FTP User initiates requests and supplies parameters for the actual transfer of files, which the User and Server proceed to carry out cooperatively. The FTP User and FTP Server roles differ in that the FTP User interacts with the human user (usually through some sort of keyboard interpreter) and takes the initiative in user/server interactions, whereas the FTP Server plays a comparatively passive role.

The question of which machine is the FTP User and which is the FTP Server is entirely independent of the direction of file transfer. The two basic file transfer operations are called *Retrieve* and *Store*; the Retrieve operation causes a file to move from Server to User, whereas Store causes a file to move from User to Server.

The Alto FTP subsystem contains both an FTP User and an FTP Server, running as independent processes. Therefore, to transfer files between a pair of Altos, one should start up the FTP subsystem on both machines, then issue commands to the FTP User process on one machine directing it to establish contact with the FTP Server process in the other machine. Subsequent file transfers are controlled entirely from the FTP User end, with no human intervention required at the Server machine.

Transferring files to or from a Maxc system or an IFS involves establishing contact with FTP Server processes that run all the time on those machines. Hence, one may simply invoke the Alto FTP subsystem and direct its FTP User process to connect to the machine.

In the descriptions that follow, the terms *local* and *remote* are relative to the machine on which the FTP User program is active. That is, we speak of typing commands to our local FTP User program and directing it to establish contact with an FTP Server on some remote machine. A Retrieve command then copies a file from the remote file system to the local file system, whereas a Store command copies a file from the local file system to the remote file system.

Furthermore, we refer to *local* and *remote filenames*. These must conform to the conventions used by the local and remote host computers, which may be dissimilar (for example, Alto versus Maxc). The Alto FTP knows nothing about Maxc filename conventions or vice versa.

The Alto FTP subsystem also includes a third process, called a *User Telnet*, which simulates a terminal in a manner exactly analogous to the Chat subsystem (though lacking some of its finer features). By this means, you may log in to a file server machine to perform operations not directly available via the basic file transfer mechanisms.

2. Calling the FTP subsystem

A number of options are available when running FTP. The program decides which parts of itself to enable and where user commands will come from by inspecting the command line. The general form of the command line to invoke FTP looks like:

>Ftp/*global-switches* *host-name* *command-list* CR

All parts of the command line except the subsystem name FTP are optional and may be omitted.

2.1 Global switches

Global switches, explained below, select some global program options such as using the Trident disk instead of the Diablo. The first token after the *global-switches*, if present, is assumed to be a *host-name* (a discussion of which appears later in the description of the Open command). The User FTP will attempt to connect to the FTP Server on that host. After connecting to the server, if a *command-list* is present, an interpreter is started which feeds these commands to the User FTP. When the command list is exhausted, FTP returns to the Alto Executive. If no command list is present, an interactive keyboard command interpreter is started.

Each global switch has a default value which is used if the switch is not explicitly set. To set a switch to *false* precede it with a minus sign (thus FTP/−S means "no Server"), to set a switch to *true* just mention the switch.

Switch	*Default*	*Action*
/S	true	[Server] starts the FTP Server. The Server is not started if the User is enabled and is being controlled from the command line.
/U	true	[User] starts the FTP User. As explained above, the interactive command interpreter or the command line interpreter will be started depending on the contents of the command line.
/C	true	[Chat] starts the Telnet. The Telnet is not started if the User is enabled and is being controlled from the command line, or if the system disk is a Trident.
/T	false	[Trident] sets the system disk to be a Trident drive. The default is 0, but can be changed by following the /T with a unit number between 0 and 7 (thus FTP/T5 means use Trident unit 5). User and Server commands apply to files on this disk but command line input is still taken from Com.cm on the Diablo drive.
/L	false	[Log] causes all output to the User FTP window to also go to the file FTP.log on DP0, overwriting the previous contents. Log is true if the User is enabled and is being controlled from the command line.
/A	false	[AppendLog] enables the log but appends to FTP.log rather than overwriting it.
/E	true	[Error] causes FTP to ask you if you want to continue when a non-fatal error happens during execution of a command line. FTP/−E will cause FTP to recover automatically from non-fatal errors without consulting you.

/R	true	[Ram] allows FTP to use some microcode which speeds things up slightly. If your Alto has no ram, this switch is ignored.
/B	false	[Boot] creates FTP.Boot for distribution to boot servers.
/D	false	[Debug] starts FTP in debug mode.

The rest of the global switches are explained below under "Server Options".

2.2 FTP user log

FTP can keep a log (typescript) file for the FTP User window. The file name is FTP.log. It is always enabled when FTP is being controlled from the command line; otherwise it is controlled by the /L and /A global switches. Some keyboard commands do not treat the user window as a simple teletype, so the typescript for these commands will not be exactly what you saw, alas.

2.3 Using a Trident disk

Starting FTP with the /T global switch causes FTP to access local files on a Trident disk (assuming one is connected) rather than on the standard Diablo disk. Accessing a file on a Trident requires more code and more free storage than accessing a file on the Diablo. Since FTP is very short on space, only a User or a Server FTP is started when the /T switch is set. The default is to start a User FTP, but specifying no user (FTP/T−U) or specifying a server (FTP/TS) will start a Server FTP instead.

2.4 Server options

Server options are controlled by global switches and by subcommands of the Server keyboard command. There are currently four options:

Switch	*Default*	*Action*
none		If no server option is specified, retrieve requests (disk to net) are allowed. Store requests (net to disk) are allowed unless the store would overwrite an already existing file. Delete and Rename are not permitted.
/P	false	[Protected] Retrieve requests are allowed. Store, Delete, and Rename are not permitted.
/O	false	[Overwrite] Retrieve requests are allowed. Store requests may overwrite files. Delete and Rename are permitted.
/K	false	[Kill] FTP will return to the Alto Exec when the server connection is closed. A simple form of remote job entry can be performed if the user FTP stores some Executive commands into Rem.cm.

2.5 The FTP display

The top inch or so of the display contains a title line and an error window. The title line displays the release date of that version of FTP, the current date and time, the machine's internetwork address, and the number of free pages on the disk. The error window displays certain error

messages if they arrive from the network (errors are discussed in more detail below). A window is created below the title line for each part of FTP that is enabled during a session (server, user, and telnet).

If the FTP Server is enabled, it opens a window and identifies itself. If a User FTP subsequently connects to this Server, the User's network address will be displayed. The Server will log the commands it carries out on behalf of the remote User in this window. The Server is not enabled when FTP is being controlled from the command line.

The FTP User opens the next window down and identifies itself. The command herald is an asterisk.

The User Telnet opens the bottommost window, identifies itself, and waits for a host name to be entered. The Telnet is not enabled when FTP is being controlled from the command line.

3. Keyboard command syntax

FTP's interactive command interpreter presents a user interface somewhat similar to that of the Alto Executive. The standard editing characters, command recognition features, and help facility (via "?") are available. When FTP is waiting for keyboard input, a blinking cursor will appear at the next character position.

3.1 Directing keyboard input to the User and Telnet windows

The bottom two unmarked keys control which window gets characters from the keyboard. Striking the unmarked key to the right of the right-hand SHIFT key directs keyboard input to the Telnet window. Striking the unmarked key to the right of the RETURN key directs keyboard input to the FTP User window. A blinking cursor will appear in the window that owns the keyboard if the window is awaiting typein.

3.2 Keyboard commands

*Open *host-name*CR

Opens a connection to the FTP Server in the specified host. FTP permits only one user FTP connection at a time. In most cases the word "Open" may be omitted: i.e., a well formed *host-name* is a legal command and implies a request to Open a connection. FTP will try for one minute to connect to the specified host. If you made a mistake typing *host-name* and wish to abort the connection attempt, hit the middle unmarked key (to the right of RETURN).

Ordinarily, *host-name* should be the name of the machine you wish to connect to (e.g., Maxc). Most Altos and Novas have names which have been registered in Name Lookup Servers. So long as a name lookup server is available, FTP is able to obtain the information necessary to translate a known host name to an inter-network address.

If the name of the server machine is not known, you may specify an inter-network address in place of the *host-name*. The general form of an inter-network address is:

network # host # socket

where each of the three fields is an octal number. The *network* number designates the network to which the Server host is connected (which may be different from the one to which the User host is connected); this (along with the "#" that follows it) may be omitted if the Server and User are

known to be connected to the same network. The *host* number designates the Server host's address on that network. The *socket* number designates the actual Server process on that host; ordinarily it should be omitted, since the default is the regular FTP server socket. Hence, to connect to the FTP server running in Alto host number 123 on the directly-connected Ethernet, you should say Open 123#CR (the trailing "#" is required).

*CloseCR

Closes the currently open User FTP connection.

*Login *user-name password*CR

Supplies any login parameters required by the remote server before it will permit file transfers. FTP will use the user name and password in the Operating System, if they are there, and logging into FTP will pass them back to the OS in the same manner as the Alto Executive's Login command.

When you issue the Login command, FTP will first display the existing user name known to the OS. If you now type a space, FTP will prompt you for a password, whereas if you want to provide a different user name, you should first type that name (which will obliterate the previous one) followed by a space. The command may be terminated by carriage return after entering the user name to omit entering the password.

The parameters are not checked immediately for legality, but rather are sent to the server for checking when the first file transfer command is issued. If a file transfer command is refused by the server because the name or password is incorrect, FTP will prompt you as if you had issued the Login command and then retry the transfer request. Striking DEL in this context will abort the transfer command.

A user name and password must be supplied when transferring files to and from a Maxc system or an IFS. The Alto FTP Server requires a user-password to be supplied if the server machine's disk is password-protected and if the password in the server machine's OS does not match the password on the disk. Thus if the OS was booted and FTP invoked because a Request-for-Connection was received (which bypasses password checking), FTP will refuse access to files unless a password is supplied. However if the OS was booted normally, FTP assumes that the disk owner (who knew the password) will control access by using the server option switches. The user-name is ignored.

*Connect *directory-name password*CR

Requests the FTP server to *connect* you to the specified directory on the remote system, i.e., to give you owner-like access to it. The password may be omitted by typing RETURN after the directory name. As with Login, these parameters are not verified until the next transfer command is issued. At present, the Connect command is meaningful only when transferring files to or from a Maxc system or an IFS; the Alto FTP server currently ignores connect requests. If the multiple directory feature of the Alto Operating System ever comes into widespread use, this may be changed.

*Directory *directory-name*CR

Causes *directory-name* to be used as the default remote directory in data transfer commands (essentially it causes *directory-name* to be attached to all remote filenames that do not explicitly mention a directory). Specifying a default directory in no way modifies your access privileges, whereas connecting gives you owner access (and usually requires a password). Explicitly mentioning a directory in a file name overrides the default directory, which overrides the connected directory, which overrides the login directory. Punctuation separating *directory-name* from other parts of a remote filename should not be included. For example you might type Directory Alto not Directory <Alto>.

*Retrieve *remote-filename*CR

Initiates transfer of the specified remote file to the local host. The syntax of *remote-filename* must conform to the remote host's file system name conventions. Before transferring a file, FTP will suggest a *local-filename* (generally the same as the *remote-filename* without directory or version), and will tell you whether or not the file already exists on your local disk. At this point you may make one of three choices:

1. Type RETURN to cause the data to be transferred to *local-filename.*
2. Type DEL to indicate that the file is not to be transferred.
3. Type any desired *local-filename* followed by RETURN. The previous *local-filename* will disappear, the new filename will replace it, and FTP will tell you whether a file exists with that name. This filename must conform to Alto conventions. You now have the same three choices.

If *remote-filename* designates multiple files (the remote host permits "*" or some equivalent in file names), each file will be transferred separately and FTP will ask you to make one of the above three choices for each file. At present, only Maxc and IFS support this capability. That is, you may supply "*"s in the *remote-filename* when retrieving files from a Maxc or IFS, but not when retrieving files from another Alto.

*Store *local-filename*CR

Initiates transfer of the specified local file to the remote host. Alto filename conventions apply to the *local-filename*; "*" expansion is not supported. FTP will suggest a *remote-filename*, to which you should respond in a manner similar to that described for Retrieve except that if you supply a different filename, it must conform to the remote file system's conventions. The default *remote-filename* is one with the same name and extension as the local file; the remote server defaults other fields as necessary. If the remote host is a Maxc system or an IFS, then the directory is that most recently supplied in Login, Connect, or Directory commands and the version is the next higher.

*Dump *remote-filename*CR

Bundles together a group of files from the local file system into a *dump-format* file (see the Alto Executive documentation for the dump-file format and more on dump-files in general) and stores the result as *remote-filename.* FTP will ask you for the names of local files to include in the dump-file. Terminate the dump by typing just RETURN when FTP asks for another filename. By convention, files in dump-format have extension .dm.

*Load *remote-filename*CR

Performs the inverse operation of Dump, unbundling a dump-format file from the remote file system and storing the constituent files in the local file system. For each file in the dump-file, FTP will suggest a local file name and tell you whether a file by that name exists on your disk. You should respond in the manner described for Retrieve.

*List *remote-file-designator*CR

Lists all files in the remote file system which correspond to *remote-file-designator.* The *remote-file-designator* must conform to file naming conventions on the remote host, and may designate multiple files if "*" expansion or some equivalent is supported there.

If the *remote-file-designator* is terminated by a comma rather than RETURN, FTP prints a prompt of "**" at the left margin and prepares to accept one or more subcommands. These subcommands request printout of additional information about each file. To terminate subcommand input, type RETURN in response to the subcommand prompt. The subcommands are:

Type	Print file type and byte size.
Length	Print length of file in bytes.
Creation	Print date of creation.
Write	Print date of last write.
Read	Print date of last read.
Times	Print times as well as dates.
Author	Print author (creator) of file.
Verbose	Same as Type Write Read Author.
Everything	Print all information about the file.

This information is only as reliable as the Server that provides it, and not all Servers provide all of these file properties. Altos derive much of this information from hints, so do not be alarmed if it is sometimes wrong.

*Delete *remote-filename*CR

Deletes *remote-filename* from the remote file system. The syntax of the *remote-filename* must conform to the remote host's file system name conventions. After determining that the remote file exists, FTP asks you to confirm your intention to delete it. If the *remote-filename* designates multiple files (the remote host permits "*" or some equivalent in file names), FTP asks you to confirm the deletion of each file.

*Rename *old-filename new-filename*CR

Renames *old-filename* in the remote file system to be *new-filename.* The syntax of the two filenames must conform to the remote host's file system name conventions, and each filename must specify exactly one file.

*QuitCR

Returns control to the Alto Executive, closing all open connections.

*Type *data-type*CR

Forces the data to be interpreted according to the specified *data-type,* which may be Text or Binary. Initially the type is Unspecified, meaning that the source process should, if possible, decide on the appropriate type based on local information.

*Byte-size *decimal-number*CR

Applicable only to files of type Binary, Byte-size specifies the logical byte size of the data to be transferred. The default is 8.

*EOL *convention*CR

Applicable only to files of type Text, this command specifies the End-of-line convention to be used for transferring text files. The values for *convention* are CR, CRLF, and Transparent. The default is CR.

*User ...

Allows you to toggle switches which control operation of the FTP User. There is currently only one, Debug, which controls display of protocol interactions. Warning: this printout (and the corresponding one in the Server command below) sometimes includes passwords.

*Server_...

Allows you to toggle switches that control operation of the FTP Server. The switches are Protected, Overwrite, Kill, and Debug, corresponding to the global switches /P, /O, /K, and /D.

*Telnet_...

Allows you to toggle switches that control operation of the Telnet. There is currently only one, Close, which closes the Telnet connection if one is open, and clears the Telnet window.

4. Command line syntax

The User FTP can also be controlled from the command line. As explained previously, the first token after the subsystem name and *global-switches* must be a legal *host-name*; if the User FTP can't connect to the FTP Server on that host it will abort and return control to the Alto Executive. If a *command-list* follows the host name, the command line interpreter is invoked instead of the interactive keyboard interpreter. This permits the full capabilities of the Alto Executive (filename recognition, "*" expansion, command files, etc.) to be used in constructing commands for FTP.

Each command is of the form:

keyword/switch-list argument ... argument

To get a special character (any one of *#';) past the Alto Executive, you must precede it by a single quote. To get a "/" into an FTP argument, the "/" must be proceeded by two single quotes (the second one tells FTP to treat the "/" as an ordinary character in the argument, and the first one gets the second one past the Alto Executive).

Unambiguous abbreviations of commands are legal. However, when constructing command files, you should always spell commands in full, since the uniqueness of abbreviations in the present version of FTP is not guaranteed in future versions.

A command is distinguished from arguments to the previous command by having a switch on it, so every command must have at least one switch. The switch /C has no special meaning and should be used on commands where no other switches are needed or desired.

4.1 Command line errors

Command line errors fall into three groups: syntax errors, file errors, and connection errors. FTP can recover from some of these, though it leaves the decision about whether to try up to you.

Syntax errors such as unrecognized commands or the wrong number of arguments to a command cause FTP's command interpreter to get out of sync with the command file. FTP can recover from syntax errors by simply ignoring text until it encounters another command (i.e., another token with a switch).

File errors such as trying to retrieve a file that does not exist are relatively harmless. FTP recovers from file errors by skipping the offending file.

Connection errors such as executing a store command when there is no open connection could cause FTP to crash. FTP can't recover from connection errors.

When FTP detects an error, it displays an error message in the User window. If the error is fatal, FTP waits for you to type any character and then aborts, causing the Alto Executive to flush the rest of the command line, including any commands to invoke other subsystems after FTP. If FTP can recover from the error, it asks you to confirm whether you wish to continue. If you confirm, it plunges on, otherwise it aborts. The confirmation request can be bypassed by invoking FTP with the global error switch false (FTP/-E ...) in which case it will plunge on after all non-fatal errors. If you aren't around when an error happens and you have told FTP to get confirmation before continuing after an error, the remote Server will probably time out and close the connection. If you then return and tell FTP to continue, it will get a fatal connection error and abort.

4.2 Command line commands

Open/C *host-name*

See the description in "Keyboard commands". The first token after the subsystem name and global switches is assumed to be a host name and no Open verb is required (in fact, if you supply it, FTP will try to make a connection to the host named Open, which is almost certainly not what you want).

Close/C

Closes the currently open User FTP connection.

Login/C *user-name password*

See description in "Keyboard commands". The *password* may be omitted.

Login/Q *user-name*

Causes FTP to prompt you for the password. This form of Login should be used in command files since including passwords in command files is bad practice.

Connect/C *directory-name password*

See description in "Keyboard commands". The *password* may be omitted.

Connect/Q *directory-name*

Causes FTP to prompt you for the password needed to connect to the specified *directory-name.* This form of Connect should be used in command files since including passwords in command files is bad practice.

Directory/C *directory-name*

See description in "Keyboard commands".

Retrieve/C *remote-filename ... remote-filename*

Retrieves each *remote-filename,* constructing a local filename from the name body of the actual remote filename as received from the Server. FTP will overwrite an existing file unless the /N (No overwrite) switch is appended to the Retrieve command keyword.

If the remote host allows "*" (or some equivalent) in a filename, a single *remote-filename* may result in the retrieval of several files. (Note that one must quote the "*" to get it past the Alto Executive's command scanner.) As noted previously, this capability is implemented only by the Maxc and IFS FTP Servers at present.

Retrieve/S *remote-filename local-filename*

Retrieves *remote-filename* and names it *local-filename* in the local file system. This version of Retrieve must have exactly two arguments. FTP will overwrite an existing file unless the /N (No overwrite) switch is also appended to the Retrieve command keyword. The *remote-filename* should not cause the server to send multiple files.

Retrieve/U *remote-filename ... remote-filename*

Retrieves *remote-filename* if its creation date is later than the creation date of the corresponding local file. A file will be retrieved only if a file with the same name already exists on the local disk. This option may be combined with Retrieve/S to rename the file as it is transferred.

Retrieve/V *remote-filename ... remote-filename*

Requests confirmation from the keyboard before writing a local file. This option is useful in combination with the Update option since creation date is not a fool-proof criterion for updating a file.

Store/C *local-filename ... local-filename*

Stores each *local-filename* on the remote host, constructing a remote filename from the name body of the *local-filename*. A *local-filename* may contain "*", since it will be expanded by the Alto Executive into the actual list of filenames before the FTP subsystem is invoked.

Store/S *local-filename remote-filename*

Stores *local-filename* on the remote host as *remote-filename*. The *remote-filename* must conform to the file name conventions of the remote host. This version of Store must have exactly two arguments.

Dump/C *remote-filename local-filename ... local-filename*

See the description in "Keyboard Commands".

Load/C *remote-filename*

See the description in "Keyboard commands". If the /V switch is appended to the Load command keyword, FTP will request confimation before writing each file. Type RETURN to write the file, DEL to skip it. FTP will overwrite an existing file unless the /N (No overwrite) switch is appended to the Load command keyword.

Delete/C *remote-filename ... remote-filename*

See the description in "Keyboard commands". If the /V switch is appended to the Delete command keyword, FTP will request confirmation before deleting each file. Type RETURN to delete the file, and DEL if you don't want to delete it.

Compare/C *remote-filename ... remote-filename*

Compares the contents of each *remote-filename* with the file by the same name in the local file system. It tells you how long the files are if they are identical or the byte position of the first mismatch if they are not. (No corresponding command is available in the Keyboard command interpreter for implementation reasons: there is not enough room for it in Alto memory.)

Compare/S *remote-filename local-filename*

Compares *remote-filename* with *local-filename.* The *remote-filename* must conform to the file name conventions of the remote host. This version of Compare must have exactly two arguments.

Rename/C *old-filename new-filename*

See the description in "Keyboard commands".

Type/C *data-type*

See the description in "Keyboard commands".

Byte-size/C *decimal-number*

See the description in "Keyboard commands".

EOL/C *convention*

See the description in "Keyboard commands".

Debug/C

See the description of the Debug subcommand of the User command in "Keyboard commands".

4.3 Command line examples

To transfer files FTP.run and FTP.syms from the Alto called Michelson to the Alto called Morley, one might start up FTP on Michelson (to act as an FTP Server), then walk over to Morley and type:

```
>Ftp Michelson Retrieve/c Ftp.run Ftp.syms^CR
```

Alternatively, one could start an FTP server on Morley (invoking it by FTP/O to permit files to be overwritten on Morley's disk), then issue the following command to Michelson:

```
>Ftp Morley Store/c Ftp.run Ftp.syms^CR
```

The latter approach is recommended for transferring large groups of files such as *.run (since expansion of the "*" will be performed by the Alto Executive).

To retrieve User.cm from the FTP server running on Alto serial number 123 (name unknown, but it is on the local Ethernet):

```
>Ftp 123'# Retrieve User.cm^CR
```

Note that the "#" must be preceded by a single quote when included in a command line, since otherwise the Alto Executive does funny things with it. (Quotes are not necessary when typing to FTP's interactive keyboard interpreter).

To start FTP, have the FTP User connect to Ivy, and then accept further commands from the keyboard:

>Ftp Ivy^CR

To retrieve <System>Pup-Network.txt from Maxc and store it on the Alto as PupDirectory.bravo, and store PupRTP.bcpl, Pup1b.bcpl, and PupBSPStreams.bcpl on <DRB> with their names unchanged:

>Ftp Maxc Connect/c drb mypassword Retrieve/s <System>Pup-Network.txt
PupDirectory.bravo Store/c PupRTP.bcpl Pup1b.bcpl PupBSPStreams.bcpl^CR

To retrieve the latest copy of all .Run files from the <Alto> directory on Maxc, overwriting copies on the local disk (the single quote is necessary to prevent the Alto Executive from expanding the "*"):

>Ftp Maxc Ret/c <alto>'*.Run^CR

To update the local disk with new copies of all <Alto> files whose names are contained in file UpdateFiles.cm, requesting confirmation before each retrieval:

>Ftp Maxc Dir/c Alto Ret/u/v @UpdateFiles.cm@^CR

To store all files with extension .Bcpl from the local disk to your login directory on Iris (the Alto Executive will expand *.bcpl before invoking FTP):

>Ftp Iris St/c *.Bcpl^CR

To retrieve <System>Host-name/descriptor-file.txt;43 (two single quotes are necessary to get the "/" past the Alto Executive and the FTP command scanner, and one quote is necessary to get the ";" past the Alto Executive):

>Ftp Maxc Ret/c <System>Host-name"/descriptor-file.txt';43^CR

5. File Property Defaulting

Without explicit information from the file system, it is often difficult to determine whether a file is Binary or Text, if Binary, what its byte-size is, and if Text, what End-of-line convention is used. The User and Server FTPs use some simple heuristics to determine the correct manner in which to transfer a file. The heuristics generally do the right thing in the face of incomplete information, and can be overridden by explicit commands from a human user who knows better.

The FTP protocol specifies a standard representation for a file while in transit over a network. If the file is of type Binary, each logical byte is packed right-justified in an integral number of 8-bit bytes. The byte-size is sent as a property along with the file. If the file is of type Text, each character is sent right-justified in an 8-bit byte. An EOL convention may be sent as a file property. The default is that RETURN marks the end of a line.

5.1 File types

FTP determines the type of a local file by reading it and looking for bytes with the high-order bit on. If any byte in the file has a high-order bit on, the file is assumed to be Type Binary, otherwise it is assumed to be Type Text. FTP will generate a warning, but allow you to send what it thinks to be a text file as type Binary, since no information is lost. It will refuse to send a binary file as type text.

Moral: Don't specify a Type unless you know what you are doing. The heuristic will not lose information.

5.2 Byte-size

If a file is type Binary, the byte-size is assumed to be 8 unless otherwise specified. The FTP User and Server will both accept binary files of any byte-size and write them as 8 bit bytes on the disk. No transformation is done on the data as it is written to the disk: it is stored in network default format. Since there is no place to save the byte-size property, it is lost.

Similarly, requests for Binary files will be honored with any byte size, and whatever is on the disk will be sent to the net without transformation. Since Alto files have no byte-size information, the byte-size property will be defaulted to 8 unless otherwise specified, in which case whatever was otherwise specified will be returned as the byte-size.

> *Moral*: Don't specify a Byte-size unless you know what you are doing. Alto-Alto transfers can't go wrong. Alto-Maxc transfers with weird byte-sizes will not work unless the byte-size specified in the Alto to Maxc direction is the same as the byte-size in which the file was stored on the Alto. If it isn't, the Alto will not give any error indication, but the result will be garbage.

5.3 End-of-line conventions

FTPs are expected to be able to convert text files between their own internal EOL convention and the network standard. Conveniently enough, the Alto file system's internal representation of a text file is the same as the network standard. The Alto FTP does not do any transformations on text files. It will refuse to store a text file coming in from the net whose EOL convention is CRLF.

As an escape to bypass conversion and checking, EOL convention Transparent tells both ends *not* to convert to network standard, but rather send a file as-is. This is included for Lisp files which contain internal character pointers that are messed up by removing LFs.

> *Moral*: Don't specify an EOL convention unless you know what you are doing. If your text file is a Lisp source file, specify EOL convention Transparent.

5.4 File dates

The Alto file system keeps three dates with each file: Creation, Read, and Write. FTP treats the read and write dates as properties describing the local copy of a file: when the file was last read and written in the local file system. FTP treats the creation date as a property of the file contents: when the file contents were originally created, not when the local copy was created. Thus when FTP makes a file on the local disk, the creation date is set to the creation date supplied by the remote FTP, the write date is set to "now" and the read date is set to "never read".

6. Abort and error messages

Error and Abort packets are displayed in the system window. Abort packets are fatal; Error packets are not necessarily so.

The most common Abort message is "Timeout. Good bye", generated when a server process has not received any commands for a long time (typically 5 minutes).

The most common Error message is "Port input queue overflowed" indicating a momentary shortage of input buffers at the remote host. Receiving an Error Pup does not imply that the file in transit has been damaged. Loss of or damage to a file will be indicated by an explicit message in the User FTP window. The next iteration of Pup will probably rename Error Pups to be Information Pups.

7. Telnet

FTP provides a simple User Telnet as a convenience for logging into a remote host (e.g., Maxc) to poke around without having to leave the FTP subsystem and start Chat. It lacks most of the creature comforts Chat provides, such as automatic attaching to detached jobs, automatic logging in, etc. The Telnet is not enabled when the User FTP is being controlled from the command line. When the Telnet does not have an open connection, it waits for you to type a host name with the syntax explained above for the Open command, and then attempts to connect to the specified host. If you wish to abort the connection attempt, hit the bottom unmarked key (opposite the right-hand SHIFT key). You can get a larger Telnet window by not starting a server (type FTP/−S to the Executive).

NEPTUNE

by K. KNOX

Neptune Reference Manual

Table of Contents

1. Introduction

Neptune is a file utility program for managing file directories on Alto disks. Using Neptune, you can examine directories, delete files, copy files from one disk to another, and display their contents and attributes. Neptune can be used on either single-disk or multiple-disk Altos, including systems with Trident disk drives.

2. Start-up procedure

There are no command line options, so to execute the program, simply type

>Neptune^{CR}

to the Alto Executive. After displaying the available directories, the program waits for you to issue commands.

Neptune's normal start-up procedure depends on whether your Alto has one or two disk drives and on what disks you have loaded into the drives:

If you have only one disk drive, or you have two but you haven't loaded a disk into the second drive (DP1), Neptune displays the directory for the standard Alto disk (DP0) on the left side of the screen and a blank window on the right.

If you have two disk drives with individual disks loaded into each one, Neptune displays the directory for DP0 on the left and for DP1 on the right. In this case, if the disk in DP1 is password-protected, Neptune will require you to type that disk's password before you may proceed further.

If you have two disk drives with the two halves of a double-disk file system loaded into them, Neptune will treat them as a single disk of twice the normal size and will display their combined directory in the left window only.

3. The screen

The display consists of five major windows which are outlined with black lines. Directory information is displayed inside, or just outside, these windows.

At the very top of the screen is the *system window* where general information and error messages are displayed. A "ready" message will be visible in this window when Neptune is waiting for you to enter commands. To the left and right of the system window are the words **Start**, **Quit**, **Clear**, and **Type** enclosed in gray boxes; these are commands that you can invoke by clicking them with the mouse, as will be described in section. 4.

Underneath the system window, the screen is divided into two parts, a left and a right side. Each side contains information about one directory.

The upper window on each side is a status window. Just above this window appear the *owner name* and *disk name* for the disk shown on that side. Inside the status window is displayed such information as the number of free pages on the the disk, the total number of files, and the number of files for which you have selected various operations. Directly underneath the window are the *disk drive*, the *directory name*, and the *filter* used to view the directory.

The major portion of each side consists of a directory window where the names of the files are displayed in alphabetical order. You can scroll the directory to view other parts of it.

4. Commands

Neptune is controlled partially by the mouse and partially by the keyboard. You can do three things using the mouse:

scroll the directory windows, much as in Bravo;

select individual files that you want to manipulate;

invoke commands.

The keyboard is used only for text entry and for confirmation of certain commands invoked by the mouse.

4.1 Scrolling

A directory can be scrolled very much like a document in Bravo. Just to the left of each directory window there is an invisible *scroll bar.* When the cursor is over this bar, the cursor takes on the shape of a vertical double-headed arrow. Pressing and releasing RED inside the scroll bar moves the text opposite the cursor to the top of the window. BLUE scrolls in the opposite direction, moving the text at the top of the window down to the cursor position. While you are holding down RED or BLUE, the cursor changes to an upward or downward-pointing arrow, respectively.

YELLOW positions the directory in the window in proportion to the cursor's position along the left edge of the window. The top and bottom of the scroll bar correspond to the beginning and end of the directory. When you hold down YELLOW, the cursor turns into a triangle, and another, stationary triangle appears to show where the directory is presently positioned.

The beginning and end of the directory are indicated by

~~~ BEGINNING ~~~

~~~~~ END ~~~~~

4.2 Selecting files and operations

Neptune's most useful operations on individual files are *copy* and *delete.* You select *which files* you wish to manipulate by pointing at them, and you specify *which operation* you want to perform on each one by clicking a particular mouse button. The operations are not performed immediately but are deferred until you invoke the **Start** command, which tells Neptune to perform them all.

To select a file name, move the cursor over the name and press a mouse button. The button does not have to be released over the name for the file to be selected. This means that you may select a number of adjacent files by holding down the mouse button and sweeping the cursor over the group of names.

There are three possible operations, corresponding to the three mouse buttons. Selecting a name with a mouse button has one of the following meanings, depending on which button you press:

RED	Make a copy of this file in the other directory.
YELLOW	Copy this file under a different name.
BLUE	Delete the file from this directory.

While a name is selected it is displayed white-on-black rather than black-on-white. The operation that is to be performed is indicated by a special symbol in the narrow column along the inner edge of the directory window. This *operations column* is separated from the directory names by a gray line.

If you select a file name with RED, an arrow will appear in the operations column, pointing toward the other directory, indicating that the file is to be copied into the other directory. If you select the file with YELLOW rather than RED, Neptune will immediately request that you type a name (terminated by CR or ESC) which is to be the new name of the file in the other directory. This name will be displayed just below the selected file name, with the words "change to" in the operations column.

If you select a file name with BLUE, a large X will appear next to it in the operations column, indicating that the file is to be deleted.

If you select a file name that is already selected, the file name will be deselected and the operation on that file cancelled.

When you are satisfied with the selections, start execution of the operations by positioning the cursor over **Start** (in the upper-left corner) and clicking RED. Neptune will now begin performing the operations you have selected, displaying messages in the system window to tell you what it is doing. When it is done, it will update the directory windows and say "ready" in the system window. You may select a maximum of 100 files before issuing the **Start** command. Operations are performed in the order that they were selected. Neptune may turn off part of the display while it is performing the operations. This is so that it can use the memory occupied by the display bit map as buffers to speed up disk transfers.

4.3 Changing the filter

Initially, all the file names in a directory are displayed in that directory's window. You can change this by specifying a file name pattern or *filter* to describe some subset of the file names you wish Neptune to display. The filter currently being used to display a directory is shown underneath its status window, along with the disk drive and directory name; this appears something like

DP0: <SysDir> *.*

In this case, DP0 is the disk drive, SysDir is the directory name, and *.* is the filter. When you start up Neptune, the initial filter for each window is *.*, which matches all files.

To change a filter, position the cursor over the existing one and click RED. A small blinking square will appear to the left of the filter. Next, type the new filter, terminated with CR, ESC, or TAB. All the file names that match this filter will now be shown in the corresponding directory window. You may include "*" in the filter to match any sequence of characters, just as in the Alto Executive. If you strike DEL while typing the filter, the default filter *.* will be used.

When you change a filter, any file operations you have selected but not yet executed in the affected window are cancelled. This avoids any possible confusion about file names that are selected but are not visible in the directory window because they don't match the new filter.

4.4 Displaying a file

Neptune can display file attributes and contents in much the same fashion as the Executive's FileStat and Type commands. To display a file, position the cursor over **Type** (near the upper-right corner of the screen) and click RED.

At this point, Neptune will temporarily deselect any file names that you may have selected previously, and a message will appear in the system window requesting you to select a file name.

Position the cursor over the name of the file you wish to display (in either directory window) and click RED.

As soon as you do this, Neptune will replace the two directory windows with a single window in which the first page of the file is shown. Other information about the file, including its length, serial number, creation date, etc., is displayed in the system window.

You cannot scroll the file window, but you can view successive pages of the file, one page at a time. To move on to the next page, type a space. When you reach the end of the file or you strike DEL, Neptune will restore its normal display (including any selections you made prior to invoking **Type**), and revert to the "ready" state.

4.5 Other commands

The **Start** and **Type** commands have already been described. In the upper-right corner of the screen are two other commands, **Quit** and **Clear.**

To leave Neptune, click RED over **Quit**. Normally, Neptune will give control back to the Alto Executive immediately. However, if you have selected any file operations and have not yet performed them, Neptune will request that you confirm by typing CR or ESC. In this case, you may cancel the **Quit** command by striking DEL.

You can cancel all selections in both directory windows by clicking RED over **Clear**. You must then confirm this command by typing CR or ESC, or cancel it by typing DEL.

5. Disks and directories

Neptune has facilities for dealing with disks other than the standard Alto disk, and with directories other than the standard one on each disk. Most users won't have occasion to use these features. If you find that you have accidently entered a mode in which you are asked to choose a new disk drive or directory, strike DEL to return to the "ready" state with the previous selections intact.

Displayed between the status and directory windows, on each side, are the current *disk drive*, the *directory name*, and the *filter* used to display that directory. When you start up Neptune, the left-hand window's disk drive is DP0, its directory is SysDir, and its filter is *.*. Whenever Neptune is in the "ready" state, you can change any of these by pointing at it and clicking RED. This section describes how to change the disk drive and the directory; the method for changing the filter has already been described (section 4.3).

A disk drive has a name consisting of a drive type and drive number. The two Diablo drives (the type used in standard Altos) are named DP0 and DP1. On Altos that have Trident disk drives attached, the eight possible Trident drives are called TP0 through TP7. If a Trident model T-300 drive is attached, there may be up to three independent file systems stored on a disk pack mounted in that drive. You identify the one you want by use of ficticious drive names; for Trident drive 3, they are called TP3, TP403, and TP1003. See the Trident documentation in the Alto Subsystems manual for more information about T-300 disks.

On any given disk there are one or more directories containing the names of files stored on that disk. The directories are themselves files and have names. The standard directory on any disk, and the one you always refer to if you don't say anything else, is called SysDir.

In Neptune, the current disk drive and directory name for a window are not entirely independent of each other. You may change the directory name without specifying a new disk drive, but changing the disk drive may require that you then select a directory name.

5.1 Switching to a different disk

To select a new disk drive for the left or right side of the screen, position the cursor over the current disk drive name for that side and click RED. Neptune will clear out the directory window and display in its place the names of all possible disk drives, i.e., all drives that are connected to the Alto and that have disk packs loaded and ready. Choose the desired disk drive by pointing at its name and clicking RED.

If the disk you selected is password-protected (i.e., a password was specified when the Operating System was installed on that disk), Neptune will require you to enter the password at this time.

After you have chosen one of the disk drives, Neptune will look for directories on the disk loaded in that drive. Ordinarily the only directory present will be SysDir, and Neptune will revert to its "ready" state after displaying the directory in the normal fashion. On the other hand, if the disk has alternate directories available, Neptune will now request that you select one of them, as discussed below in section 5.2.

Directly following the list of available disk drives is the phrase "No Disk". Selecting this rather than a disk drive name leaves that side of the screen undefined, i.e., not referring to any disk drive. A side with no disk is indicated by the phrase "No Disk" as its drive name and a gray outline around its directory window.

5.2 Switching to a different directory

Before we discuss how to change the current directory name, a warning about the use of alternate directories is in order. The only fully-supported directory in the Alto file system is the one called SysDir. The Operating System does give limited support to other directories, but not all Alto programs will recognize them. More information about alternate directories is given in the Alto Operating System manual.

To change the directory name for a window, point at the current one with the cursor and click RED. Neptune will then look for directory files on the current disk. All the directories that it finds will be displayed in the directory window. Choose the desired directory by pointing at its name and clicking RED. Neptune will now display the file names in that directory and revert to its "ready" state.

If you wish to create a new directory rather than selecting an existing one, then instead of clicking RED over an existing name simply type the new directory name followed by CR or ESC. There must not already be a file by that name. After you have done this, the directory window will of course be empty. You can now put files into this newly-created directory by copying them from the other directory window.