UNIX à la Mac

An introductory look at A/UX. What's so different about the Mac's alternative lifestyle?

by Matt Zeidenberg

Much has been made of the coming of UNIX, an operating system developed at Bell Laboratories in 1970. Despite manufacturers such as Sun, Hewlett-Packard, and Apple, who have managed to add a user interface to an otherwise faceless system, operating in a standard UNIX environment has never been for the faint of heart. Although Apple's version, A/UX, is based on AT&T's System V version of UNIX, Apple has added major extensions for network operation and support for Macintosh applications. Even so, operating A/UX on a Mac is in many ways just like running UNIX System V on any other microcomputer, minicomputer, or workstation. The day may come when operating A/UX will be as easy as running the Macintosh operating system, but until then a basic knowledge of A/UX and how it differs from Mac OS will satisfy the curious.

Multiprocessing and Security

First and foremost, A/UX is a multiuser, timesharing operating system. When you sit down in front of a UNIX system, it prompts you for your name (or a user code that has been set up for you) and then asks for your secret password. If the password is valid, it logs you on—that is, allows you to use the system—and places you in your directory. Every user has a personal directory of files. Thus, unlike most stand-alone personal computer operating systems (including the Mac's), A/UX provides security, protecting the system and individual files on the system from unauthorized access, and individual users from each other.

UNIX, in addition to being a multiuser system, is a multiprocessing system (see "Multiuser A/UX"). Under UNIX, each user may run several programs simultaneously. For instance, you can edit a file, compile a program, and run a third program, all at once. Apple's MultiFinder runs only one application at a time, even though several may appear on the desktop.

Command Language versus Graphics Interface

While the Macintosh OS is a graphics-based system, A/UX is command oriented. Instead of selecting menu items, you type commands, a procedure that has both disadvantages and advantages. The main drawback is that you have to remember the commands and their syntax.

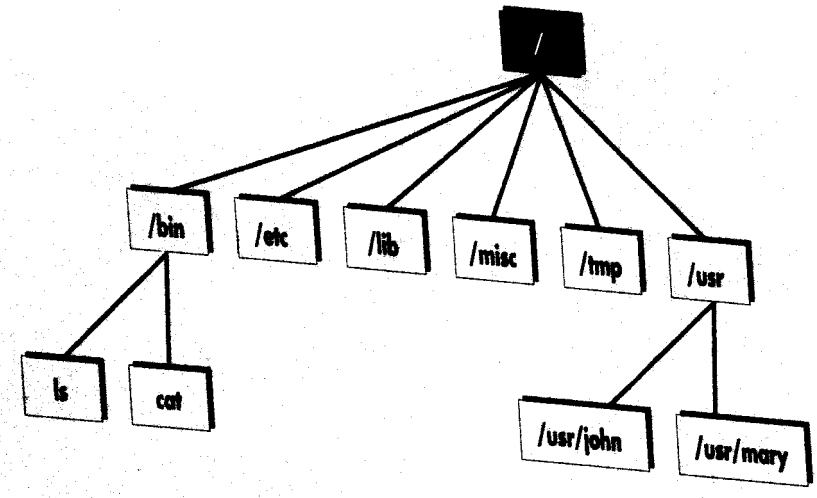
The main advantage of a command language is its flexibility. With a command, you can easily accomplish many tasks that you cannot do using a graphics interface. For instance, in A/UX, to remove all files that begin with the letter *b,* you type the command **rm b***. (The asterisk character is called a *wild card;* it matches anything.) Using the Macintosh Finder, by contrast, you must select each file individually.

Of course, you need not make a choice between a graphics interface and a command language; graphics interfaces that have been built on top of UNIX—such as X, Andrew, and NeWS (see "What's in a Window?" *Macworld,* November 1987)—give you both. Menus, icons, and dialog boxes are ideal for simple commands, while a command language enables you to issue complex commands. Both X and NeWS interfaces are available for A/UX.

The File System

A/UX is very similar to the Macintosh OS in the structure of its file system. Both UNIX and the Mac OS use a hierarchical file system (HFS) (see "Tree Views"). Under A/UX, folders are called *directories.* As in the Mac OS, A/UX directories can contain other directories. With the Macintosh OS, the top-level directory is the particular disk or volume one is using; under A/UX the top-level directory is always called the root (as in the root of a tree—albeit an upside down tree).

Matt Zeidenberg, a graduate student in computer science at the University of Wisconsin, Madison, has written on artificial intelligence, computer graphics, and networking.



Tree Views

The top-level directory, called the root, is denoted by a single slash (/). The root contains several subdirectories. The subdirectory /bin contains the programs that are executed by most commands, such as ls and cat. When you type a command at the shell, the shell usually looks first in /bin for a program to execute. Various other files are stored in the subdirectories /etc, /lib, /misc, and /tmp (these may have their own subdirectories, not shown). User files are stored in subdirectories of /usr. The directory system is similar to the Mac's HFS, but in A/UX there are large areas of the directory structure that are used by the system, whereas in the Mac HFS, system files are confined to the System Folder.

Virtual Memory

A/UX, unlike the Mac OS or MS-DOS, has an advanced feature called *virtual memory,* which allows applications to behave as if they had much more memory than is actually installed. For instance, an application that requires 2 megabytes of memory can run on a system with only 1MB of RAM. When the program starts up, A/UX loads the first portion of the program from the disk into memory. Later, when more of the program is needed, A/UX loads the required sections into memory, swapping out unnecessary parts to make room. The application designer need know nothing about virtual memory. A program can be written as if it were running in 2MB. Program and data space is limited only by available space on the hard disk.

This is a somewhat oversimplified description of the way A/UX operates. Actually, virtual memory techniques almost always bring data in and out of memory in chunks of one to several disk blocks, for efficiency's sake. These chunks are called *pages.* For each page that is brought into memory, another page must be overwritten. If any changes have been made to the page that is being overwritten, it must first be copied to the disk.

Virtual memory is complicated by the fact that UNIX is a multiprocessing system; many programs, as well as the operating system itself, share memory. The operating system must take care that users don't step on each other's toes and that each has enough memory to run.

Virtual memory requires hardware support in order to run efficiently. This is because every time a program references a memory address, the hardware must determine whether the data associated with that address is in memory. If it isn't, a *page fault* occurs, and the operating system must load in the page containing the required data. This process, called *address translation,* occurs so frequently that it needs hardware support. The Macintosh II provides this support via the paged memory management unit (PMMU), which is available as an option. A/UX requires both the PMMU and increased memory for operation. The 68030 microprocessor has a built-in PMMU; future 68030 versions of the Mac II will therefore not require additional PMMU hardware.

Comparing the Finder and the Shell

Action	Mac Finder	A/UX Shell
Running a program	Double-click on application icon	Type program name
Deleting a document	Drag icon to trash	Type rm
Renaming a document	Use cut-and-paste editor on document name beneath icon	Type mv
Examining contents of a folder (directory)	Double-click on folder to open window	Type ls
Duplicating a folder or document	Choose Duplicate from menu, or drag icon to destination	Type cp

C Shells versus the Finder

A/UX has a program called the *shell,* or *command interpreter,* that interprets typed commands and passes them to the operating system for execution. Because UNIX was written by and for computer experts, its commands tend to be cryptic. For instance, we have *cp* for copy and *mv* for move; UNIX users don't like to type.

The shell is so named because it surrounds the operating system. The user interacts with the shell rather than with the operating system. Similarly, on the Mac, the user interacts with the Finder rather than with the Mac OS. There are several versions of the UNIX shell, the most popular being the C shell, the Bourne shell, and the Korn shell. The following discussion applies to the C shell.

The shell does more than interpret individual command lines. It is actually a programming language that lets you specify loops, conditional branching, and so forth. Thus if you have a particular se-

quence of shell commands that you use frequently, you can put them all into a file and execute the file as a shell program.

Comparing Commands

Let's compare the most basic user actions in the Mac OS and A/UX. The operations are only roughly analogous, though, since UNIX normally lacks a graphics interface. For instance, to see the contents of a directory (folder) on UNIX, you need to type a command—on the Mac you need only open the folder.

■ Double-clicking on the Mac either starts an application or displays the contents of a folder. Under A/UX, to start an application, you must type its name. So, for instance, to start the text editor *vi,* you type **vi**.

■ On the Mac, when you are using the Finder, typically you have an active window that represents the contents of a folder. Under A/UX, you have a default directory, which is like the active window in the Finder. When you log on to A/UX, the default directory is set to an area in the file system called the *home directory.* You can create subdirectories of a home directory, subsubdirectories of subdirectories, and so forth.

By typing **ls** (for list) you can see the contents of the default directory. This is roughly analogous to opening a folder by double-clicking on it. Type **ls -l** to display an extended or long version of the directory (see "Directory Displayed"). The *-l* in *ls -l* is a flag; it tells the ls command to do something special. In this case the flag makes A/UX print out more information about each file: its owner and its protection status (that is, who is allowed to read, alter, or execute it).

In order to change the default directory, you use the cd (change directory) command. For instance, suppose you are currently in your home directory. The path name of the home directory might be /usr0/matt. The term *path name* derives from the fact that the name shows the path from the top level (root) directory to the user's directory. Imagine that your home directory contains a subdirectory called *macworld,* you can change to this subdirectory by typing **cd macworld**. Now the default directory path name is /usr0/matt/macworld. Now typing **ls** lists the files in this directory.

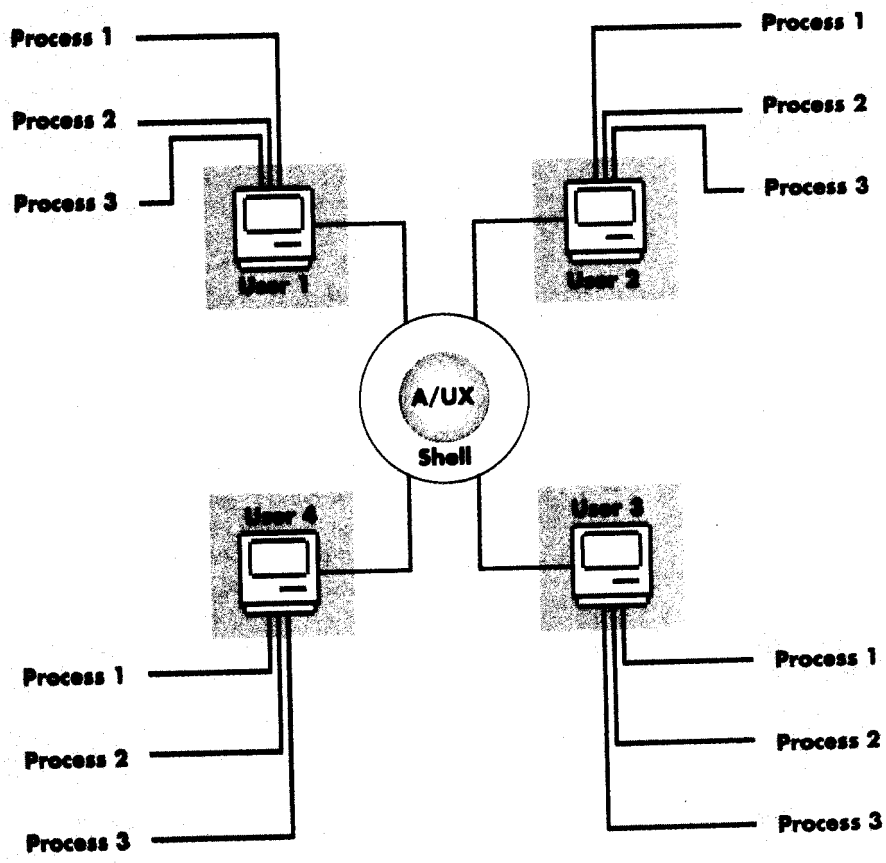
Under A/UX, there is a way to get quickly to any directory. For instance, if you want to get to a directory called *foo,* but you don't know exactly where it is, type **cd ~foo**. The tilde (~) acts as a wild card for any prefix in the path name.

■ To create a folder in the Finder, you select the command New Folder from the File menu. You then name the folder by editing the name beneath its icon. Under A/UX, to create a new directory called *foo* in the current default directory, simply type **mkdir foo**. You can then open it by typing **cd foo**.

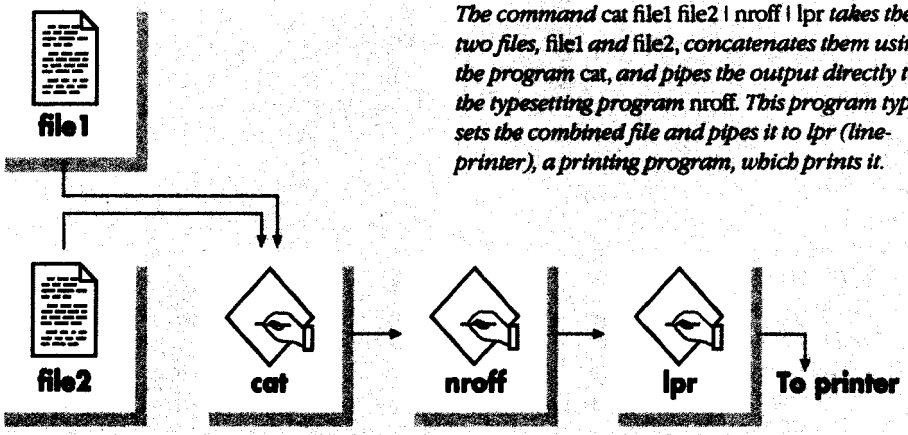
■ In order to delete files and folders with the Finder, you select them and drag them to the trash. Under A/UX, to delete a file called *bar,* type **rm bar** (rm being short for remove). To delete a directory called *foo,* type **rmdir foo**. Using wild cards makes this command more versatile; to delete all files ending in *.for,* for example, type **rm *.for**.

■ With the Finder, renaming a file is a simple matter; you edit the name of the file beneath its icon. Under A/UX, if you have a file called *fromage,* and you want to rename it *cheese,* you must type **mv fromage cheese**. The *mv* stands for move; this command also moves a file from one directory to another.

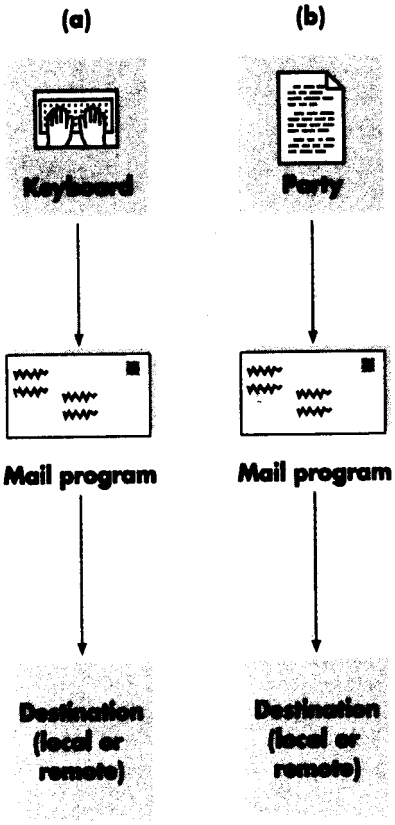
■ To duplicate a file under the Finder, use the Duplicate command in the File menu. To copy a file from one disk to another, you drag the file's icon from the first disk to the second. Under A/UX, you use the cp (short for copy) command. For instance, if you are in your home directory, and it contains subdirectories called *foo* and *bar,* and you want to duplicate a file called *data* that is in subdirectory *bar,* you type **cp foo/data bar/data**.



Multiuser A/UX
A/UX is a multiuser, multiprocess system. Users can have as many processes as they wish. Here four users have three processes apiece. Each user communicates with A/UX via a special program called the shell.



Connecting Pipes
The command cat file1 file2 | nroff | lpr *takes the two files,* file1 *and* file2, *concatenates them using the program* cat, *and pipes the output directly to the typesetting program* nroff. *This program typesets the combined file and pipes it to* lpr *(line-printer), a printing program, which prints it.*



Redirecting Input
(a) Normally, in using a UNIX mail program, you type messages interactively from the terminal. Alternately, as shown in (b), you can create a message file using a UNIX editor such as vi. Then when you type **mail john < party** *the message contained in the file* party *is sent to John. The less than symbol (<) indicates that the mail program takes its input from the file instead of the terminal.*

Move Commands Compared

■ On the Mac, it is easy to duplicate a file, but not so easy to simply move it. If you drag a file's icon to a new folder or disk, a copy of the file remains in the original location; you must then remove the unwanted copy. A/UX lets you move a file in one step using the mv command. For instance, to move the file *data* in the example above instead of duplicating it, you simply type **mv foo/data bar/data**. This moves the file *data* from directory *foo* to directory *bar.*

With A/UX, the full name of a file is the full path name of the directory it is in, followed by the file name. For instance, a file containing the text for this article might be titled */usr0/matt/macworld/article.* Viewed in this light, you can see that the mv command is actually a sort of rename command, since it changes the file's full name. (This is why the mv command also renames files, as we saw earlier).

Examining Files

With the Mac operating system, in order to examine a file, you have to enter an editor, such as MacWrite. Under A/UX to see the contents of the file *foo,* you type **cat foo**. The cat command concatenates, or outputs, the file to the standard output device, the screen.

The more command is a slight variation on the cat command: more displays a file on the terminal one screenful at a time. After each screenful, you simply hit the space bar to see more; when you're finished, you press **q** (for quit). The more command prevents files from flying by so fast that you can't even see them.

A/UX-Specific Command Features

The A/UX command language, or shell, has a number of advanced features that have no equivalent under the Macintosh OS.

As mentioned earlier, A/UX protects files against unauthorized access. The only file-protection mechanisms available on the Mac involve either making files invisible or encrypting them. Of course you can store floppy disks in a safe, but assuming you want to protect files stored on a hard disk, you must rely on the file protection provided by the operating system.

Each file under A/UX has three levels of permissions: permission to read the file, to write (change) the file, and to execute (run) the file. As the owner of a file you can grant these permissions to three categories of users: yourself, your group, and others. Your group consists of people you designate by using the set group command. *Others* means everyone on the system, other than the user and the group members.

To set permissions on a file, you use the chmod (change mode) command. For instance, if you want members of your group to be able to read and execute the file *foo,* type **chmod g+rx foo** (*g* stands for group, *r* stands for read, and *x* for execute).

You control access to the contents of your directory by setting the execution permission level of the directory. If you want everyone to be able to look at your subdirectory *macworld,* simply type **chmod o+x macworld**, when you're in the parent directory.

Redirection

A/UX assumes that all interactive processes take their input and output from the terminal. The terminal is viewed as two files, *stdin* and *stdout,* standard input and output.

UNIX allows you to get input and output from files other than stdin and stdout (see "Redirecting Input"). For instance, suppose you want to use the UNIX typesetting program *nroff.* You have an input file consisting of typesetting commands and text, called *article.in.* You want to create a file called *article.out* consisting of the typeset text. To do this, type the command **nroff < article.in > article.out**. The less than sign (<) tells nroff to take its input from the following argument (in this case, article.in); the greater than sign (>) does the same for output. This is called redirection of input and output.

The Pipe

The *pipe* in UNIX is a way of directing the output from one program to the input of another (see "Connecting Pipes"). For instance, if you want to print the contents of the current directory, you type **ls | pr**. The vertical bar is the pipe; it indicates that the output of the ls command should be used as the input to the pr (print) command.

You can string more than one pipe together; for instance, the command *cat file1 file2* | *nroff* | *mail joe* concatenates the two files *file1* and *file2*, uses the result as input to the typesetting program *nroff*, and sends nroff's output as mail to user *Joe*.

The closest thing to a pipe in the Mac OS is the Clipboard. You can think of a pipe as a sort of automatic Clipboard—a way of instructing one application to deliver its output to another, instead of your having to move the information using cut, copy, and paste commands.

Background Processes

What Mac user hasn't had the frustrating experience of waiting for a program to finish running? With A/UX, this is not a problem.

For instance, suppose you want to do other things while you are waiting for a long document to be typeset. You simply type **nroff < article.in > article.out &**. The shell returns the number of the nroff background process that you started, while allowing you to continue working and typing other shell commands. When the background process is complete, the process number is displayed. You can run as many background processes as you like, but since they all share one central processing unit, they compete for time.

A/UX on the Network

One reason for using A/UX is the wealth of software for UNIX in the network environment. In its initial release, A/UX doesn't support AppleShare, Apple's own network file-access system, although support for AppleShare is planned. In the meantime, A/UX uses Sun Microsystems' Network File System (NFS). NFS was developed under UNIX but can provide network file services to machines running heterogeneous operating systems.

The basic advantage of NFS is that it nests remote machines' files in a user's own local, hierarchical file system. Suppose you have a top-level directory called *usr* with subdirectories *bin, lib,* and *doc* on your machine, which is named *client*.

Now suppose you have a second machine, named *server*. On this machine, there are some documents in the directory called */usr/text* that you want to access. You can place this entire directory in your local file system by using the command

mount server:/usr/text /usr/doc/text

The mount server command makes all the files in usr/text on server available locally on client in /usr/doc/text.

Whenever you access a file in this new directory, NFS actually sends a request to the server machine to read or write it. The file is not actually copied to the client machine. If you use a file often in a particular program or application, NFS may make a local copy for the duration of your work session, for efficiency's sake.

Here's how it works: the NFS running on the local machine makes a remote procedure call (RPC) to the NFS on the remote machine over the network. The server process then ships back the data or file, and the client process proceeds. The whole thing looks, from the programmer's point of view, like an ordinary procedure call, as in C or Pascal.

The nice thing about NFS is that it makes the network transparent to the user. The remote files are simply embedded in the local file system, and users have no idea whether they are working with local or remote files. A file that looks like it's on your disk may actually be on a disk across the hall or across the country. □

See *Where to Buy* for contact information.

Directory Displayed

The long version of the list command (ls -l) prints information about the file, as shown here. Links show the number of times that a file appears in directories.