Journal of Microcomputer Applications (1992) **15**, 209–227

TUTORIAL

UNIX for microcomputers: an alternative to better management of microcomputer resources and higher productivity

Hossein Saiedian and Stanley Wileman

Department of Computer Science, University of Nebraska, Omaha, NE 68182, USA

From laptops to supercomputers, UNIX[1] is leading the way to better management of resources and higher productivity, particularly in multiuser systems and open multivendor computing environments. The UNIX philosophy which is responsible, in part, for this success is captured in the phrase 'small is beautiful', which refers to the concept of providing a set of small, well-defined utilities which can be easily combined to develop application programs and solve computation problems. In this paper, we provide a brief introduction to the UNIX operating system and present a short history of operating systems for the PCs and describe the central role that operating systems play in providing a user-friendly environment for the users. We then compare the UNIX and DOS operating systems and argue that UNIX in many ways is superior to DOS. The advantages of UNIX over DOS are enumerated. The potential disadvantages of UNIX are discussed. The road ahead for personal computing and UNIX-based microcomputers is briefly discussed. The paper concludes by suggesting that microcomputer users should strongly consider using UNIX as an alternative to DOS.

1. Introduction

Although early versions of the UNIX operating system were developed nearly two decades ago, many microcomputer users have never heard of, used, or learned anything about the software that provided inspiration for much of what MS-DOS[2] and OS/2 offer today. While understandable, it is unfortunate, since UNIX provides greatly needed stability for microcomputer users and developers in a time when new microprocessors, peripherals, operating systems, and applications abound. As technological improvements result in a plethora of more sophisticated hardware and application software, our ability to use effectively these systems is dependent on a consistent, well-defined interface between the user, the application programs, and the hardware.

An operating system is the software 'base' that makes a computer system usable. Its purpose is to efficiently organize and control the resources (memory, disks, printers, terminals, etc.). It provides the mechanisms by which complicated peripheral devices like floppy and hard disks, primary and extended memory, and graphic displays can be easily utilized by application programs. These mechanisms are provided by system calls, or functions that are invoked by programs to utilize resources, and utility programs which use these system calls to perform common tasks. For example, most systems provide calls to read or write a single record from a disk file, and a utility program that uses those system calls to copy all the records in one disk file to another. Operating systems are

[1] UNIX is a trademark of AT&T Bell Laboratories.
[2] MS-DOS is registered trademark of Microsoft Corporation.

0745–7138/92/030209+19 $03.00/0

tailored to the particular set of hardware in the computer system, and as a result, there are many different operating systems available. Most of these operating systems, however, are designed for a single family of computer systems. MS-DOS and OS/2, for example, and are available only for systems based on the INTEL 80 × 86 family, while DEC's VMS operating system can only be used on the VAX[3] family of computers.

UNIX is unique among operating systems in that it is available for almost every computer system in use, from the largest supercomputers (where it is dominant) to each of the current microcomputer systems, including IBMs, Apples, Commodores, and others. The same set of system calls and utility programs are provided in each case, making it easy and profitable for application developers and users to use UNIX. The UNIX philosophy which is responsible, in part, for this success is captured in the phrase 'small is beautiful', which refers to the concept of providing a set of small, well-defined tools which can be easily combined to solve problems.

2. History of PCs and PC operating systems

The first generation of microcomputer systems to gain any commercial popularity was based on the 8-bit INTEL 8080 microprocessor. These machines, including the MITS Altair and the IMSAI, were successful because they were based on a consistent definition of the interface between the microprocessor and the peripherals, the S-100 bus. This allowed many third-party providers to produce disk drive controllers, modems, serial and parallel interfaces, and other specialized products, each of which could be attached to the same microcomputer system.

The other important factor influencing the success of these early microcomputers was the development of the CP/M[4] operating system (Control Program for Microcomputers). CP/M was popular because, like the S-100 bus, it provided a consistent set of functions for utilizing the computer system's resources. Prior to CP/M, developers and users resorted to *ad hoc* mechanisms for accessing and managing the system resources. With CP/M, software developers could produce applications that employed CRTs, printers, primary memory (RAM), and disk files using a well-defined set of system calls. A standard set of utility programs for file management (copying, deleting, and so forth) was provided with CP/M. CP/M, and systems that followed it, included features that were derived from earlier operating systems. CP/M's PIP (Peripheral Interchange Program) utility for copying files was named after and provided functions similar to a system for the Digital Equipment PDP-11.

The next generation of microcomputers, based on the 16-bit INTEL 8088/8086 microprocessor, prompted additional operating system development. CP/M was adapted to these machines, resulting in CP/M-86, but it never achieved the same level of success as the operating system developed by IBM and Microsoft for the 'PC', as it became known. Alternately known as MS-DOS and PC-DOS, the operating system was based in a large part on UNIX; the set of system calls provided by MS-DOS is similar to the set provided by UNIX (with certain notable exceptions, mentioned later). Other factors may have contributed to CP/M's failure in the PC market, but the well-designed set of system calls provided by MS-DOS certainly played a major role.

[3] VAX is a trademark of Digital Equipment Corporation.
[4] CP/M is registered trademark of Digital Research, Incorporated.

With more powerful microprocessors, larger primary memories, larger and cheaper disks, and more sophisticated video displays, current microcomputer systems are again prompting operating system development. The Microsoft offering, OS/2, is intended to provide a sophisticated graphical user interface, similar to that provided by the operating system for Apple Computer's Macintosh line, and at the same time provide support for concurrent (simultaneous) execution of multiple user applications. There is less dominance in this arena by OS/2, however, since many users are unconvinced that a major change in the way in how they use their microcomputers will yield any added benefits. Indeed, many 'add-ons' for MS-DOS already provide graphical interfaces and apparent concurrent execution of programs.

UNIX, too, is becoming a significant operating system for the current group of microcomputer systems. It is available for most medium and large hardware configurations, and is provided by numerous vendors. Additionally, many UNIX-like utilities are available for MS-DOS that allow users to experience some of the power of the UNIX tool set. These programs use the MS-DOS system calls but provide the same user interface found in UNIX.

3. The UNIX operating system

The UNIX operating system was designed and implemented at AT&T laboratories by Ken Thompson in 1969.[5] The first version of UNIX was designed for the DEC PDP-7 system and was written entirely in assembly language. When UNIX was developed in the late 1960s, many computer systems still ran jobs one at a time in batch mode. Because these computer systems served only one user at a time, they did not take full advantage of the power and speed of computers. Furthermore, these systems made it difficult to share data and programs, and they did not promote co-operation among programmers working on the same project.

UNIX system provides three major improvements over batch systems:

Multiuser. The UNIX system allow simultaneous access by more than one user.
Interactive. The UNIX system allows a person to communicate interactively and directly with a computer via a terminal.
Shareability. The UNIX system makes it easy for users to share data and programs.

The UNIX operating system was rewritten in 1973, this time in the language C,[6] with the help of Dennis Ritchie, also a researcher at the Bell Labs. Since C is a high-level programming language, it helped make UNIX system software more portable and understandable. Also, the fact that the UNIX operating system was developed primarily by two researchers (Ken Thompson and Dennis Ritchie) has contributed to the coherence and uniformity of UNIX design.

Universities and colleges have also played a major role in furthering the popularity and enhancement of UNIX. When the UNIX operating system became widely available in 1975, Bell Labs offered it to universities at no charge or at minimal cost. The universities, in turn, used it in the computer science departments, ensuring that the computer science students became familiar with it. Because UNIX is an advanced operating system, the students became acquainted with a sophisticated program

[5] The name UNIX was coined in 1970 by Brian Kernighan, a researcher at Bell Labs.
[6] Both MS-DOS and OS/2 are also written primarily in language C.

development environment, and as these students entered industry, they expected to work with a similar advanced environment. Thus the road for UNIX to enter industry was paved.

In addition to introducing its students to UNIX, the universities made valuable contributions to the development and enhancement of the system. For example, the computer science department of the University of California at Berkeley made significant additions and changes to UNIX to make it more popular. These changes were so significant that today one of the two most prominent versions of UNIX is called Berkeley Standard Distribution (BSD) of the UNIX system. The other major version is AT&T's UNIX System V. It is this heritage, development in a research environment and enhancement in a university, that has made the UNIX operating system powerful and friendly, and one of the most notable successes in the field of operating systems [12]. UNIX has now become the preferred operating system for many software developers, particularly developers of advanced, graphic-intensive CASE systems. It is estimated that UNIX systems will soon be shipped with 20 per cent of all new computers built each year [3].

Two trends in the computer industry have created requirements that set the stage for the growth of UNIX systems [14]: advancement in hardware technology created the need for an operating system that could take advantage of available computer hardware, and decreasing cost of hardware—hardware manufacturers could no longer afford to develop and support proprietary operating systems.

The UNIX system meets both of the above needs. That is, it takes full advantage of available hardware and it is 'generic'. Furthermore, since UNIX is now written in a high-level and machine-independent language (i.e., C), it can be easily ported to different machines. Once a UNIX operating system is ported to another machine, a large library of utility programs and functions becomes available on the new machine. This portability has been a major contributing factor to the success of the UNIX operating system.

UNIX offers yet another advantage to software developers. Since it was developed by highly skilled programmers to support their own projects, it provides a rich software development environment. That is, it makes it easy to develop and combine programs, thus encouraging a modular, tool-oriented, building-block approach to program design [10]. UNIX has become so important in the computer industry that in recent years there has been widespread desire for standardization of the system. According to Deitel [3], over one million UNIX systems have been shipped and UNIX may eventually become the world's most widely used operating system.

4. UNIX vs. DOS

Many people who use PCs think of MS-DOS as the only available operating system for the PCs. Those who need some kind of multitasking capability and who have a 80386-based machine might use concurrent DOS. But concurrent DOS, at best, provides multiple copies of DOS that can run concurrently.

UNIX is not an operating system limited to minicomputers. It is available on many machines, from supercomputers to personal computers. In fact, Deitel states that UNIX has become the personal computer operating system of choice among 'power users' and is likely to become the personal computer operating system of choice for millions of

others. Many UNIX systems, especially those based on 80386 processors, provide users with MS-DOS compatibility.

It should be noted that MS-DOS has adopted many of the traits of the UNIX system. For example, the entire file structure and many popular commands of DOS, such as those for making a directory (MKDIR), for switching between directories (CD), for displaying a file (MORE), for piping, and so forth, are borrowed from UNIX where they were originally made popular.

One of the most widely known UNIX-based operating systems for PCs is called XENIX.[7] Developed by Microsoft in the early 1980s, it offers a full implementation of UNIX with commercial features for the PC users.

The UNIX operating system, in general, and XENIX in particular are in many ways superior to MS-DOS as described below:

Multiusers. UNIX is *inherently* a multiuser system; that is, it can support more than one user at a time. Since UNIX allows more than one user to utilize the system resources almost simultaneously, the costly resources can be utilized more efficiently, thus minimizing the cost per user.

Multitasking. UNIX also *inherently* supports multitasking. That is, it allows a user to run more than one job at a time. The multitasking/multiuser capabilities of UNIX make it possible for several users to simultaneously use the same system. With multiple terminals connected, either through a serial port, a modem, or a network, each user has the same access to the system's resources as the user who is using the attached keyboard and video display. This multiuser facility is augmented by commands for identifying each user before they begin work (the login/logout process), and by facilities for identifying the owner of each file so secure information remains secure. Controlled file sharing is provided in all implementations of UNIX, and most commercial versions provide record and file locking.

General purpose. UNIX has programs that give users access to the system, allow users to work with files, control program execution, get information, make computations, and communicate with others.

Software development tools. The UNIX system provides an outstanding environment for program development. The 80386 versions of UNIX (for example, XENIX) are complete UNIX systems. They include a family of well over one hundred utility programs, often referred to as commands. These commands allow users to create, display, print, copy, sort, search, format, rename and delete files. Other utilities provide on-line documentation of UNIX itself. Tools such as the C compiler, shells (explained below), and the configuration management tools that come with UNIX make program development much easier. The language C has become very popular because of UNIX. It is a modern system language and allows programmers to write modular programs and manipulate data from bits to files as necessary when writing system programs.

Document preparation. UNIX provides powerful programs that allow users to format texts to be printed and prepared for typesetting. The document preparation software is called *troff*. In addition, it provides *grap* for making graphs, *pic* for drawing, *tbl* for

[7] XENIX is a trademark of Microsoft Corporation.

formatting tables, and *eqn* for formatting equations.[8] It also has several programs that can check spelling and grammar structure of a file. These programs include *spell*, *style*, and *diction*.

The shell. The shell is a command interpreter that acts as an interface between users and the operating system. There are three popular shells in use today: the Bourne Shell [1], standard on System V, the C Shell [5], also known as Berkeley UNIX, and the Korn Shell [6]. Because different users can use different shells at the same time on one system, a system can appear different to different users. The choice of shell demonstrates one of the powers of UNIX: the ability to provide customized user interfaces [14]. Shell commands can be arranged in a file for later execution. This flexibility allows users to perform complex functions with relative ease, often with rather short commands. The UNIX shell is a full programming language and allows powerful interactive programming.

Security. UNIX allows users to protect their files from access by other users. It also allows users to share selected files with certain users. Only the owner of a file controls which users have permission to access a file and how they can access it. Such a feature is not available under MS-DOS; once a user turns the computer on, he has access to the entire file system.

Of course, UNIX has other features that makes it superior over MS-DOS including: features for networking and distributed file sharing. A simple yet powerful file/directory system that supports the organization of files into an arbitrarily deep hierarchy. Every directory can contain files and/or more directories. A directory entry for a file (called a *link*) can be placed in more than one directory, thus allowing a file to appear in many directories at once. Features that allow one program to pass its output directly to another program, making it unnecessary to use extra storage space. This feature, known as pipe, is an example of interprocess communication. The piping feature of UNIX is much more powerful than in DOS as explained later. A simple, consistent, and flexible notation for filename generation. Names can be used absolutely, or relative to any directory in the file system. A facility called *regular expressions* that defines pattern-matching strings. This language can be used with many of the UNIX utilities for searching and pattern-matching purposes. A computer-based message system (or mail system) for users' communications. A structural high-level language (language C) for system programming as well as application development. Features that allow a user to redirect the result of a program from one peripheral device or file to another. This feature is known as device-independent I/O. Each device, from interactive terminals to main memory, is treated like a file, allowing uniform file and device input and output. Some of these features are described in the next section.

5. Other unique features of UNIX

While UNIX provides all capabilities available in MS-DOS, it also goes far beyond these. Many of these capabilities are a result of the ability of UNIX to permit

[8] It should be mentioned that, because of the increasing popularity of UNIX, the list of application programs, including word processing applications, available to run under UNIX is growing every day. There are word processing software such as Microsoft Word as well as spread sheet programs, database management products, and compilers for Pascal, BASIC, FORTRAN, COBOL, etc., available for UNIX systems.

concurrent, or apparent simultaneous execution of several programs. The ability to edit or print a file while a lengthy program is executing is an obvious application of this feature, but it is typical to find several programs always running. These so-called daemon programs can respond to unanticipated events as well as providing for the execution of commands at specified times.

Networking/Distributed File Systems. Connecting microcomputer systems using MS-DOS into networks was clearly an afterthought. Many vendors now supply software extensions to MS-DOS to permit networking, and thus many different standards exist. UNIX, on the other hand, provided networking capabilities from its earliest days, even before the development of modern local area networks.

The UUCP (UNIX to UNIX Copy Program) facilities provide, as an integral part of the UNIX system, programs for file exchange (uucp), electronic mail (mail and rmail), and remote command execution (uux) on any UNIX system in a worldwide network. The exact number of systems in this network is never known, since new systems are continually being added.

UUCP works as a store and forward networking facility. Requests for file transfers and remote command execution are queued until a connection between two communicating systems is established. Then all work involving these machines is carried out. Since many machines are geographically remote from each other, this queuing of work permits delaying file transfers until inexpensive connections between the machines can be made using, for example, night long-distance telephone connections through high-speed modems. Those machines that are physically close to each other can be directly connected without modems, and uucp requests can be carried out immediately.

If transfers involve a machine that cannot be reached directly (for whatever reason), a machine that can be reached can serve as an intermediary. For example, if system A can communicate with system B, and system B can communicate with system C, then A can still exchange files and mail with C, using B as a store and forward network node.

With the development of high-speed local area networks and wide area networks, additional networking facilities were added to UNIX and are now reasonably standard. The popular TCP/IP network protocols were developed for UNIX, and are now used by thousands of interconnected UNIX systems. Telnet and ftp are the most frequently used programs utilizing TCP/IP. Telnet provides remote login facilities to any system on the network, while ftp ('file transfer protocol') allows file transfers to be easily effected. With each of these programs, the user identifies the remote machine by name (or by network number), and then provides a valid user name and password on the remote machine. (Note, once again, that security is important in UNIX.) Then the terminal session begins or requests to send or receive files can be given. The same mail command used with UUCP networking can also be used with TCP/IP, with the system deciding which of the two networking schemes to use in delivery.

Using these same networks, distributed file systems are also available to the UNIX user through systems like NFS (Network File System) and NIS (Network Information System). (See [15] for more information on NFS and NIS.) These facilities permit files on remote machines to be treated as if they are local, thus obviating the need for file copying. This can reduce secondary storage (disk) requirements, since only one copy of shared files need be stored. The management of shared databases is made simpler, and organizations can maintain timely, accurate versions of important files, documents, and programs. This facility can also be used to provide support for diskless workstations;

these machines contain bootstrapping programs in read-only memory that then request their executable code and data files directly from another machine equipped with disks.

Most networking facilities employ the client/server concept. With telnet and ftp, the system originating the connection functions as a client while the target system executes the server software. The same client/server concept is used with NFS and NIS. With NFS, filesystems to be distributed (exported) physically exist on the server system; these filesystems are then remotely mounted on one or more client systems. Client systems, too, may have physical filesystems of their own which can be made available for export. Thus a system can be both a client and a server. A common system configuration has one large system exporting major filesystems to several smaller systems, these smaller systems export their small local filesystems to the larger system. This arrangement eases the maintenance of backups, since all filesystems can be accessed from the single large system.

NIS, also called Yellow Pages (YP), was developed by Sun Microsystems to allow centralized administration of common configuration files. For example, the file /etc/passwd identifies users and their passwords. A NIS server can distribute this file to each client, where local modifications can be automatically merged with the master copy maintained on the server. Thus a central user database can be maintained that includes all users, while individual systems can restrict access to a subset of the larger group. In a network with tens, or even hundreds, of systems, NIS can make the task of modifying the user database manageable. Without it, the systems administrator must make local copies of the database on each machine, a task which might not be completed before the next change is required.

While UNIX systems traditionally use a coherent set of networking tools, MS-DOS systems use a variety of competing, incompatible networking facilities. Two of these, Novell NetWare and Microsoft's LAN Manager, are used on a significant number of machines. This not only causes difficulty integrating groups of PCs using different networking systems, but also makes the integration of these PCs and UNIX systems troublesome. Recently, however, Novell has entered into partnership with Hewlett-Packard and Unix Systems Laboratories (USL); the goal is the integration of NetWare into the UNIX networking suite. Microsoft LAN Manager is also available for some UNIX systems, and will likely become more prominent when the Windows NT operating system is available, since that system is designed to comply with existing UNIX standards.

Interprocess Communications. The ability to connect the output of one program to the input of another via a pipe is a facility likely familiar to MS-DOS users. For example, the command

```
prep some.doc | sort | uniq | comm word-list
```

might be used to perform a spelling check on file some.doc by putting each word on a line by itself (via prep), sorting them into alphabetic order (via sort), eliminating duplicates (via uniq) and finding all those that are not in a common word list called word-list. In MS-DOS, each of these programs would run separately, and disk files would be read and written to save the output of each program and provide it to the next. In UNIX, however, all four of these programs would run concurrently, with the transfer of data between programs taking place entirely in memory. This results in significantly

reduced elapsed execution time for such pipelines. Moreover, most of the UNIX commands can be used in this fashion. Thus complicated functions can be pieced together as pipelines of common commands. Also, common functions, such as output formatting, need not be built into numerous commands since output of almost any program may be passed through appropriate *filters* via pipes. Furthermore, as mentioned earlier, a file of shell commands may be executed like any other command. Shell programming thus may be used to combine ordinary programs conveniently for quite sophisticated applications without the necessity of programming in conventional languages.[9]

Filename Generation. MS-DOS provides two metacharacters, * and ?, which can be used to reference a group of similarly named files. For example, *.doc could be used to reference all files with the extension doc. In UNIX, additional metacharacters are available that provide enhanced selection capabilities. For example, the construction ?[A-C]*[678].doc might be used to reference only those files that begin with a character, followed by letters A, B, or C, followed by zero or more characters, followed by digit 6, 7, or 8, followed by .doc.[10] These metacharacters can be used in any combinations for filename generation. For example, all of the following are valid: *ABC, A*BC, ABC*, A*B*C*, AB.C*, AB.*C, and A*B*.*C*. (The MS-DOS * metacharacter can only be used as the beginning of a filename or an extension.)

UNIX file-naming rules are considerably more flexible than those of MS-DOS. Filenames may contain almost any characters the user desires, and the longest filename can include 14 characters (255 in some versions). Furthermore, a filename may be suffixed with an extension to indicate the nature of its content (e.g., .c in foo.c would indicate that this file contains C source code). However, extensions are not in any way central to the file system and are not mandatory. Thus all of the following are acceptable filenames: foo, foo.ext, foo.ext1.ext2, and .foo.ext. (The latter file, which begins with a period, is referred to as a *dot file*.) Additionally, a single file can be given multiple names (actually called links) which permit complicated names like /usr/smith/bin/text/editor to be referenced by simpler names, like edit. The links also facilitate another flexibility for the users: a file may be referenced in many different directories under the same or different names without having to duplicate it. For example, the following command makes a link to an existing file, addr-file, in the directory database:

```
ln -s /database/addr/file /software/addresses
```

Now there are two filenames: addresses in the software directory (the new name, or a link) and addr-file, in the database directory (the original file). They can be accessed and/

[9] For example, the above sequence of commands may be modified to

```
prep $1 | sort | uniq | comm word-list
```

and placed in a file called speller. The user has now just created a new command called speller for spell checking! (The notation $1 represents an input parameter. In this context, it will be replaced by the file name supplied when user types speller file-name.)

[10] In other words, question mark represents a single character, square brackets are used to define a class of characters that matches any single character within the brackets, and asterisk represents zero or more occurrences of any character.

or modified in normal ways. However, there is only one physical file; the other one (i.e., addresses) is a link. The links reduce unnecessary duplication and decrease inconsistency.

Regular Expressions. Another unique feature of the UNIX system is its uniformity. For example, it provides a powerful feature called *regular expressions* for pattern-matching and searching. This feature has several metacharacters that when combined with letters, digits, and special characters, can define many complex and different patterns that match strings of characters. Once the regular expressions are learned, they can be used with all utilities that may require pattern-matching or searching, for example the vi or grep utilities or shell files.

Mailing System. UNIX systems provide a computer-based message system (also known as an electronic mail system) that allows two or more users to communicate electronically. The mailing system provides facilities for the users to create, send, read, save, and delete messages. The development of a mailing system has significantly improved communication between computer users and has proven to be an invaluable communication tool to users in various organizations. The advantages of a mailing system can be summarized as follows: they allow asynchronous communication in which the expected recipient(s) of a message need not be present at the time of message transmission. Users are released from geographic restrictions. Sender's time and receiver's are optimally used. Reliable and speedy transport for messages is provided.

Since microcomputers have become heavily used for office automation in recent years, the mailing facility becomes specially important since offices are communication-intensive environments and since computer-based message facilities are considered integral parts of any office information system [4].

Other Features. The MS-DOS batch file facility is useful for making 'canned procedures' of commands, but its limited facilities for conditional execution, expression evaluation, and variable manipulation have been frequently lamented. Indeed, many add-on packages are available to enhance the limited facilities provided by MS-DOS. Unfortunately, there is no standardization among these add-ons. UNIX systems typically give each user a choice of multiple shells, or command line interpreters, that permit easy construction of sophisticated command procedures. Indeed, most of these shells provide most, if not all, of the features found in common programming languages, and it is not unusual to find commonly used commands implemented as shell procedures.

A vast array of applications, too numerous to mention here, exists for UNIX. UNIX was originally developed to aid document preparation, so it is not unusual to find that UNIX includes many sophisticated text preparation tools. Even the most primitive UNIX editor, *ed*, far surpasses the limited capabilities of the much-maligned EDLIN. Spelling checkers, text formatters, and typesetting utilities are commonplace.

In addition to the above, UNIX provides many other utilities that simplify tasks that if not totally impossible, will be very clumsy under DOS, for example *grep* that locates all occurrences of a string in a set of files, *wc* that performs a line, word, and byte count on a file, *history* that shows a history of commands entered, and so forth.

Graphics facilities have been added to UNIX as the hardware in this area has developed. Many of the alien landscapes in recent *Star Trek* movies were produced using UNIX-based systems. Graphics hardware has also been employed to provide graphical user interfaces, most notably the popular X-Windows facility found on many work-

stations. The X-Window has greatly enhanced user-friendly interfaces of UNIX-based systems.

The X-Window system, called X for short, is a network-based graphics windowing system which was developed jointly by MIT and Digital Equipment Corporation. Several versions of X-Window have been released; the most recent version is X-Window Version 11 (X11). Like other windowing systems, X-Window divides the screen into multiple input and output areas called *windows*. Using a terminal emulator, windows can act as 'virtual terminals', running applications. X-Window takes user input from a *pointer*. The pointer allows the user to control an application without using the keyboard. This method of using application programs is often easier (and friendlier) to learn than traditional keyboard control because it is more intuitive. Of course, the X-Window also allows keyboard input. (Only one window at a time can receive keyboard input.) For more information on the X-Window system, please see [9].

The X-Window system provides a standard platform for graphic-based applications; unlike other windowing systems (e.g., Microsoft Windows), it will be available on everything from personal computers to supercomputers. X-Window is a networked windowing system, which turns every node in a network into a display server for any other node.

The X-Window system also uses the same client/server concept employed by networking applications. In fact, X is a networking protocol, or set of rules governing communication between systems. The server in an X installation is the machine with the display, while the client is the machine on which the graphical application executes. While these may be different machines, the two are frequently the same.

6. Potential drawbacks of UNIX

Many of the apparent disadvantages of UNIX are due to lack of familiarity with the system. A common complaint of newcomers to UNIX is that the standard utilities have cryptic names. For example, in MS-DOS you COPY a file, but in UNIX you cp it; RENAME a file in MS-DOS, but mv (or 'move') it in UNIX.[11]

We do not believe that the above concern is a genuine one for several reasons. Firstly, users get used to new commands once they use them repeatedly. Secondly, one can use the *alias* mechanism available in UNIX. Aliases enable one to rename and redefine commands so they are more userfriendly and thus avoid some of the peculiarities of command line syntax. For example, alias COPY cp and alias RENAME mv, would rename the commands for copying and renaming to COPY and RENAME, respectively. These and other aliases can be placed in one of the *dot files* that is automatically executed when a user logs in. The alias mechanism allows arbitrary commands, including parameters, to be given user-selected names. Thirdly, with the invention of graphical UNIX interfaces, there is hardly a need to memorize command names at all.

Another common complaint about UNIX concerns the use of floppy disks. In MS-DOS, no special commands are needed to announce the user's intention to insert or remove a disk. In UNIX, however, the entire collection of files (including those on

[11] Of course we must note that you cannot easily rename a directory in MS-DOS, but mv works fine on directories in UNIX.

floppies) that can be accessed is treated as a single entity, and thus specific commands (mount and unmount) are required to integrate floppy disks into the directory hierarchy. Once again, use of these commands become second-nature with experience, and removable disk usage is rarer with inexpensive large hard disks and cartridge tape facilities for backups.

Those unfamiliar with multiuser systems find the necessity of logging in and out somewhat cumbersome. A typical MS-DOS user will turn on the computer, work, and turn off the computer. With UNIX, beginning work must be preceded by logging in, and the procedures for powering up or down a UNIX system are somewhat more involved than those for MS-DOS. This is due principally to concerns for maintaining the integrity of hard disks, and to allow multiple users time to complete their work before the computer is stopped. Security concerns also require users to log off after a session with the computer; failure to do so will allow anyone with physical access to the user's computer (or terminal) to access potentially secure information.

The most viable argument against switching from MS-DOS to UNIX concerns the investment in MS-DOS application software, files, and training. This problem has been addressed by many UNIX vendors, anxious to make sales in the lucrative microcomputer market, by providing MS-DOS as a UNIX application. This protects application and training investment in MS-DOS and still gives the capability to migrate to UNIX.

Other 'negative' aspects of UNIX include its more involved installation (although it needs to be done only once) as well as its cost. However, one has to remind oneself that the end result is worth the effort!

7. UNIX systems for PCs

Although UNIX was very popular in academic and research environments, it lacked some features required for the business environment, where PCs have become very popular. Some software developers thus have offered their own commercial versions of UNIX to be used with the PCs. Among these are XENIX (developed by Microsoft), SCO UNIX (developed by Santa Cruz Operation), and 386/IX (available from Interactive Systems).[12] Of course, AT&T offers its own implementation of UNIX System V for 80386-based machines as well as its own UNIX PCs. In addition to the above products, there are several other UNIX implementations for the PCs which are not as commercially popular as the previously mentioned products. These implementations include Minix, Coherent, and Cromix. Of course, both IBM and Apple, the industry's long-time PC competitors, have also their own versions of UNIX operating systems. IBM's version, called AIX, runs on OS/2 and RS/6000 systems while Apple's MAC II can run A/UX, Apple's version of UNIX.

XENIX. XENIX was developed by Microsoft in the early 1980s to meet the needs for a commerical version of UNIX. In many ways, XENIX's capabilities resemble those one would expect to find on a much larger computer system. The resemblance is not accidental, according to Woodcock and Halvorson [18]: 'XENIX is an operating system with an intriguing history—one in which the mainframe world, in the form of an operating system called UNIX, and the microcomputer world, through the efforts of Microsoft Corporation, were joined.'

[12] SCO also markets its own XENIX called SCO XENIX.

XENIX includes some general improvements as well as new features to use in a business environment. The general improvements of XENIX include its ability to take advantage of unique features of major microprocessors (e.g., Motorola 6800, INTEL 80 × 86, Z8000) and thus avail itself of memory segmentation, dynamic relocatibility of code, separation of data and instruction, input/output by memory-mapping, and so forth. XENIX enchancements include record and file locking. That is, a program can gain exclusive access to a given record, a group of records, or an entire file, thus preserving the consistency and integrity of shared files. XENIX supports most of the commands of UNIX System V as well as new commands such as config, settime, shutdown, etc., which are basically used for disk management. Furthermore, XENIX gives options of floppy booting. It allows DOS and XENIX to reside on the disk simultaneously and switching between the two would be a matter of re-booting. Microsoft also licenses XENIX to various manufacturers, including IBM, which release it on their own.

Additional information on features of XENIX can be found in Woodcock and Halvorson's book on XENIX [18]. This is considered as one of the better books on XENIX. Other books include: Jean Yates *et al.*, *The Business Guide to the XENIX System*, 1984, Addison-Wesley, Reading, Mass. Myril Clement, *UNIX and XENIX System V Programming Tool Kit*, 1986, Tab Books, Blue Ridge Summit, PA. Doug Topham: *UNIX and XENIX: A Step by Step Guide*, 1984, Brady Communications, Bowie, MD.

SCO UNIX. The SCO UNIX is one of the better implementations of UNIX for the PCs. It includes options such as NFS for setting up a network file system, TCP/IP for Ethernet support, and a windowing system. There are two versions of SCO UNIX—one that supports two users (the basic UNIX system) and one that supports an 'unlimited' number of users.

SCO UNIX includes a series of menus used for installation. These menus greatly simplify the installation and configuration of the system. SCO UNIX also offers a flexible development environment, especially if programming in C is required. The standard C compiler is developed by Microsoft; however, one can get an AT&T C compiler. Under SCO UNIX one has the option to get a cross-development capability to develop programs for DOS or OS/2 with 80 × 86 processors. SCO UNIX also offers DOS support, that is, one can boot either DOS or UNIX from the same disk. (Once the system is configured, the user can specify a prompt asking what operating system to boot. The default is UNIX unless the user specifies DOS.)[13] A relatively good book on SCO UNIX is Cuthbertson's [2].

386/IX. The 386/IX UNIX is developed by Interactive Systems. It includes many of the features of the SCO UNIX except that it costs less.[14]

The SCO UNIX and 386/IX both support C and Bourne shells; they both provide a DOS interface to let users share a disk between UNIX and DOS, and they both allow users to copy files between the UNIX partition and the /C: partition (referenced by DOS). Furthermore, both SCO UNIX and 386/IX provide support for *virtual terminals*.

[13] Additional information about SCO UNIX can be obtained from Santa Cruz Operation: Santa Cruz Operation, 400 Encinan St., P.O. Box 1900, Santa Cruz, CA 95061, USA (408) 425-7222.

[14] Additional information about the 386/IX can be obtained from Interactive Systems: 241 Colorado Ave., Santa Monica CA 90404, USA (213) 453-8649.

That is, a single terminal can be used as several *logical* terminals. For example, once the system has been booted and the primary monitor is displayed, one can switch to another virtual terminal by entering key strokes Alt-Fn under SCO UNIX and Alt-SysReq-Fn under 386/IX, where Fn is one of the functions on the PC keyboard. Now you can log on as another user! Another keystroke sequence lets you cycle through all the virtual terminals. Both of these systems allow users to use the PC monitor as the system console, a user terminal, or both. Other terminals can be connected so two or more people can use the system at the same time.

SunOS 386i. SunOS is a version of the UNIX system provided for the Sun workstations including the Sun 386i. SunOS is based on UNIX System V and BSD UNIX. It includes, among other things, a network file system (NSF), and a mouse and graphical (window-based) interface. SunOS supports both C-shell, Bourne-shell and more than 400 commands or utility programs.

The Sun Microsystem 386i contains the 80386 microprocessor, a hard disk, a 3·5-inch floppy disk drive, and connectors for Ethernet and various peripheral devices. The Microsystem 386i supports multitasking, multiusers system-like UNIX, and provides an 8088 emulation mode that enables one to run IBM-PC compatible programs. Since 386i supports a windowing interface, users may have many SunOS windows and many MS-DOS windows executing at one time; they can switch between windows, and may cut and paste data between windows, even between SunOS windows and MS-DOS windows [7].[15]

8. UNIX books, periodicals, and user groups

The following is a list of good books on UNIX:

1. *Life with UNIX* by D. Libes and S. Ressler
   Prentice-Hall, 1989, 350 pages, ISBN 0–13–536657–7
   Major sections include: UNIX in Time, UNIX Information, Inside UNIX, and Outside UNIX.
2. *The UNIX Programming Environment* by B. W. Kernighan and R. Pike
   Prentice-Hall, 1984, 357 pages, ISBN 0–13–937699–2
   The above book is perhaps one of the classic books on UNIX for programmers.
3. *Introducing the UNIX System* by H. McGilton and R. Morgan
   McGraw-Hill, 1983, 556 pages, ISBN 0–07–045001–3
   This is an introductory book for most parts. It has good sections on editors and text formatting as well as system management.
4. *Exploring the UNIX System* by S. G. Kochan and P. H. Wood
   Hyden Book Company, 1984
   Also a good introductory book on UNIX with chapters on file system, Bourne shell, UNIX editors, security, and administration.

Of course, Sobell's book [14] is also an excellent practical UNIX book. Other books on specific aspects of UNIX, for example, networking, text editing, document preparation, and administration, have also been published. There are also many useful papers in

[15] Information about SunOS can be obtained from Sun Microsystems: Sun Microsystems, 2550 Garcia Ave., Mountain View, CA 94043.

Bell Systems Technical Journal Vol. 57(6) (July–August 1978) and in the *AT&T Laboratory Technical Journal* Vol. 63(8) (October 1984). Furthermore, there are many UNIX periodicals including Journal of the USENIX Association, University of California Press, 2120 Berkeley Way, Berkeley, CA 94720, USA and UNIX World, Subscription Department, P.O. Box 1929, Marion, OH 43305, USA, which cover many aspects of UNIX systems. There are also numerous UNIX user groups that distribute public domain software, and other useful information on UNIX as well as variations of UNIX such as XENIX. Two such user groups include the following: UniForum, The International Association of UNIX Systems Users, 2901 Tasman Drive, #201, Santa Clara, CA 95054, USA, Tel.: 408-986-8840, Fax: 408-986-1645 and USENIX Association, P.O. Box 2299, Berkeley, CA 94710, USA.

9. UNIX philosophy

We have elaborated on what is sometimes referred to as the UNIX 'philosophy' in previous sections while we were comparing UNIX features with those of DOS. The following statements taken from [11] define this 'philosophy' in such a way that is more explicit and more oriented toward the levels of the operating system that the ordinary user sees: make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new features. Expect the output of every program to become the input of another, as yet unknown, program. Design and build software, even operating systems, to be tried early, ideally within weeks. Do not hesitate to throw away the clumsy parts and rebuild them. Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out.

10. Personal computing and UNIX: the road ahead

In this section we briefly discuss three relatively new topics that are related to both PCs and UNIX operating systems. These topics include PowerPC and its UNIX-based PowerOpen operating system, the NeXT system and its UNIX-based Mach operating system, and trends in distributed operating systems based on the UNIX. In the latter subsection, we will also talk about the Chorus[16] distributed operating systems.

10.1 *Co-operation between Apple and IBM*

In July of 1991, IBM and Apple, the industry's two big personal computer competitors, announced a joint venture to develop an entirely new type of personal computer with a new UNIX-based operating system. This new venture could be the turning point in the history of personal computers.

IBM and Apple announced that they have decided to develop the PowerPC, a system that will exemplify new hardware and software standards for microcomputers. The prospect of co-operation between Apple and IBM could be the most important event of the 1990s.

The PowerPC for Apple and IBM will be an offspring of IBM's RS/6000 systems—a series of high-end UNIX-based workstation systems. The PowerPC system will use a

[16] Chorus is a registered trademark of Chorus systems.

single-chip version of the RS/6000 architecture. (The chip is developed by Motorola.) IBM and Apple will incorporate PowerPC chips into their products. The advantage for the users will be that all systems based on this chip will be able to run the same software.

The operating system for PowerPC is called PowerOpen, a new version of the UNIX based on IBM's AIX and Apple's A/UX. (AIX and A/UX are IBM's and Apple's versions of UNIX operating systems.) PowerOpen will conform to OFS/1 (the UNIX standard for Open System Foundation). As a result, it will support multitasking, multiusers, and other UNIX features. While AIX (or A/UX) applications and new applications written specifically for PowerOpen will run in native code, the users are still able to run current programs. Furthermore, the new programs written for the PowerOpen operating system will have option of Macintosh interface or the OSF UNIX-standard graphical user interface (GUI).

Perhaps it may not be until 1993 before applications running on the PowerOpen platform enter the market; nevertheless, it would be interesting to speculate on the impact they will have on the personal computing world and the challenges they provide for personal computer applications developed for Microsoft MS-DOS.

10.2 *The NeXT computer system and its UNIX-based OS*

Another major development in the world of personal computing has been the emergence of the NeXT computer system, a system that could transform personal computing as thoroughly as the Macintosh has. The NeXT system has many new features which cannot be covered in this paper. An interested reader may refer to [17] for more detail.

One intriguing feature of the NeXT system is a standard device called *voice annotation*. With a click of the NeXT mouse, a user can pick up a microphone to record comments about the document he or she is working on. Once the user is done, the system marks the document to alert future users that a comment has been recorded. Future users can play back the recording with another simple click of the mouse. Another high-tech feature of the NeXT system is its incorporation of CD-ROM laser-optical disk technology which has made it possible to store the equivalent of hundreds of floppy disks on a single CD-ROM. This feature of the NeXT system implies that optical disk technology is likely to become widely used in the future.

The operating system of NeXT is a UNIX-based system called Mach. Mach is actually source-compatible with BSD UNIX 4.3. That is, programs that run on 68000-based computers using BSD 4.3 UNIX will also run on the NeXT system. The NeXT comes with a graphical user interface (GUI) that allows the users to manipulate images on the screen with the mouse. The shell and terminal applications allow power-users to use standard UNIX shell when desired for direct access to UNIX commands and utilities.

NeXT appears to have learned from the experience of the rest of the industry by developing a GUI, UNIX-based operating system. It started out with a functionality towards which the PC and workstation industry is still striving. NeXT will also benefit from UNIX (by avoiding the danger of being left out) once and if the industry decides to produce some *de facto* standard for UNIX-based operating systems for PCs and workstations. Furthermore, according to Deitel [3], many believe that Mach will become the parallel UNIX system standard because of its parallelism features. (Mach extends the BSD 4.3 UNIX operating system to a multiprocessing, shared-memory environment.) Deitel further speculates that AT&T's plan for System V Release 5 could include distinguishing features of Mach. This could obviate the increasing need for Mach operating systems.

10.3 *Distributed UNIX systems*

The need for interconnection of software products developed by different vendors as well as hardware components manufactured by different companies, has led to an increasing interest in distributed systems. A common operating system and communication protocols will simplify the task of developing a distributed system. UNIX systems have evolved to a special position of prominence in distributed systems.

There are several major approaches to distributed operating systems that have increased the viability of UNIX systems as a base for distributed systems. These approaches include BSD UNIX *Sockets*, AT&T *Streams*, and Sun's *Network File System (NFS)*. Mannas [8] analyses the merits of Streams and Sockets while Walsh *et al.* [16] describe Sun Network File System.

Another major approach to the distributed operating systems has been the Chorus approach. Chorus has been a research project on distributed systems at INRIA (France). Three versions have been developed. The latest version is called Chorus-V3. Chorus-V2 had a UNIX-based interface and was used as a basis for several experimental distributed applications [13].

Chorus technology integrates inter-process communications in its real-time operating system and provides transparent distribution of, and access to, programs and files over a network. The Chorus systems are composed of a small real-time *Nucleus* and one or more operating systems, called *subsystems*, offering application programs standard operating system interfaces. One such subsystem is the UNIX. It is included in a product called Chorus/MIX. UNIX programs can run unchanged under this subsystem. The Chorus/MIX is compatible with UNIX at three levels: *system calls*, *file structure*, and *device drivers*. The Nucleus manages the exchange of messages between communication entities (called *ports*). Figure 1 shows the basic architecture of the Chorus system.

For an excellent overview of the Chorus system, please refer to [13].

11. Conclusions

UNIX has become the operating system that represents the link between the diverse and vast majority of computer systems, from supercomputers to microcomputers. According to Deitel, there is a rush to UNIX now, with virtually every significant computer and communications vendor pointing to UNIX as the unifying theme in the field of operating systems for the 1990s.

Why is there such a rush? Perhaps it is because UNIX has had the ability to draw the industry together, to standardize this operating system—something that has never been considered for other operating systems in the market place.

We have found that UNIX systems offer a flexible and very effective environment for program development. To those who are MS-DOS users, and specifically those who find DOS limiting and who need a multitasking, well-supported operating system that does

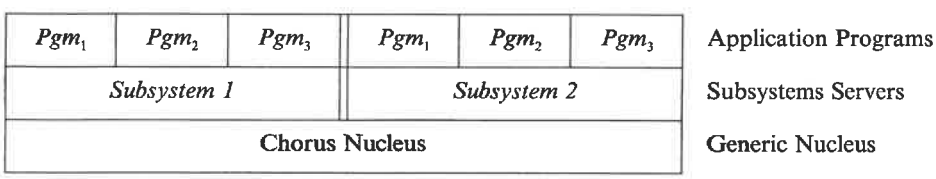

Figure 1. The Chorus Architecture.

not have DOS limitations, we recommend that you try using a PC UNIX-based operating system. We believe you will like it. Many of the newer functions and utilities of DOS as well as OS/2 are either UNIX commands or closely mimic the UNIX commands. Perhaps you will find it more effective to use these commands and integrate them with other UNIX commands in an actual UNIX environment than to continue using MS-DOS.

Acnowledgement

We are grateful to anonymous referees for their constructive comments.

References

1. S. Bourne, October 1978. The UNIX shell. *Bell System Technical Journal*, **57**(6), 1971–1990.
2. Tom Cuthbertson, 1991. *The ABC's of SCO UNIX*. SYBEX, San Francisco, CA.
3. H. M. Deitel, 1990. *Operating Systems*, 2nd edition. Addison-Wesley, Reading, MA.
4. C. Ellis & N. Naffah, 1987. *Design of Office Information Systems*. Springer-Verlag Publishing Co., New York.
5. W. Joy, August 1980. *An Introduction to the C Shell, UNIX Programmer's Manual*. Department of EE and CS, University of California, Berkeley.
6. D. Korn, 1983. A shell programming language. In *Proc. USENIX Assoc. Conference*, pp. 191–202.
7. J. Lazarus, 1988. Sun 386i overview: A technical report. Technical report, Mountain View California: Sun MicroSystem, Inc.
8. J. Mannas, October 1988. Two views one vision. *UNIX Review*, **6**(10), 77–83.
9. T. O'Reilly, V. Quercia & L. Lamb, 1988. *X Window System User's Guide*. O'Reilly & Associates, Inc.
10. R. Pike & B. W. Kernighan, October 1984. Program design in the UNIX environment. *AT&T Bell Laboratories Technical Journal*, **63**(8), 1595–1605.
11. J. Quarterman, A. Silberschatz & J. Peterson, December 1985. 4.2BSD and 4.3BSD as examples of the UNIX system. *ACM Computing Surveys*, **17**(4), 379–418.
12. D. M. Ritchie, October 1984. The evolution of the UNIX time-sharing system. *AT&T Bell Laboratories Technical Journal*, **63**(8), 1577–1593.
13. M. Rozier, V. Abroissimov *et al.*, 1988. Chorus distributed operating systems. *Computing Systems*, **1**(4), 000–000.
14. M. G. Sobell, 1989. *A Practical Guide to the UNIX System*. Benjamin-Cumming Publisher.
15. Hal Stern, 1991. *Managing NFS and NIS*. O'Reilly & Associates, Inc.
16. J. Walsh, R. Lyon & G. Sager, 1985. Overview of Sun Network File System. In *Proceedings of Usenix Winter Conference*, pp. 117–124.
17. Brude Webster, 1989. *The NeXT Book*. Addison-Wesley, Reading, MA.
18. J. Woodcock & M. Halvorson, eds., 1986. *SENIX at Work*. Microsoft Press.

Hossein Saiedian is currently an Assistant Professor of Computer Science at the University of Nebraska at Omaha (USA). He received his PhD in Computer and Information Sciences from Kansas State University in 1989. Dr Saiedian has numerous journal and proceedings publications in both Computer Science and Information Systems areas. His areas of research include office automation, message passing semantics, and formal methods. Dr Saiedian is a member of several professional societies including the ACM and Sigma Xi.

Stanley Wileman, educated at the University of Houston and Kansas State University, has been on the faculty at the University of Nebraska at Omaha for 16 years. He has been an active member of the ACM Scholastic Programming Contest Finals Group, and has primary interests in operating systems and data communication.