May 1982 Vol IV No III

AUUGN

Australian UNIX Users Group

N E W S L E T T E R

Well folks, AUUGN is not really dead, we have decided to save a little money this year and publish three copies in very quick succession. This saves on postage and allows us to have a little continuity of theme between successive issues.

The editor's have experienced a few technical bugs with the newsletter. Firstly, I left for a short holiday in London. Then Bob did a short, but very productive tour of the USA. This left Messrs Maltby, Long, Ellis and Jason to carry on the good work. Upon returning from places afar Bob and I found a general strike. Try as we might we could not force the temporary publishers back to work so now really are forced to publish the next few issues of AUUGN alone. The second major upset is that the newsletter is now being published for the University of Sydney, rather than UNSW. This means that all letters send to the other address suffered a standard Australia Post delay in being redirected here.

An attempt has been made to clear the backlog of correspondence. I found a box of unanswered mail waiting for me when I returned. Most inquiries have now been answered, and I hope that most of the subscription requests have also been honoured.

While on the subject of subscriptions, let me remind you that the annual subscription rate is $AUS24.00 for Australian subscribers and $AUS30.00 for international subscribers. PLEASE, pay be cheque made out to AUUGN, not B. Kummerfeld (P. Ivanov) or anyone else. PLEASE do not send paper money look alikes. The effort to overcome the paper warfare means that you will not get your newsletter until after I get your money. This is sometimes a standoff whereby some of you will not pay until the first issue arrives, and I will not send the first issue until I see the colour of you money. In such cases you lose, even if I do not win. Please throw the order forms away and send a cheque.

Well thats all for this issue. Its nice to be back home.

Chris Rowles
AUUGN subeditor.

Environments made easy

Dave Horsfall (dave:csu40)
Computing Services Unit
University of NSW

The environment facility under level 7 UNIX (and recently implemented under level 6 by Lindsay Harris and myself) although useful leaves a lot to be desired. There is one library call getenv(3) that retrieves the value of a variable, but no easy way of modifying or deleting a variable. Currently the only way of doing this is to reconstruct the new environment piece-meal, usually from the environ(2) pointer. There obviously needs to be a better way. Two new routines are proposed, and comments are solicited. Familiarity with environments is of course assumed.

The most obvious deficiency is the difficulty of changing the value of a variable. I propose a routine setenv(name, value) which will enter the named variable into the environment with the specified (possibly null) value. Of course, if the variable already exists it will be updated. It could return an error indication (for example, (char *) -1) should it fail e.g. no room left for more strings. On success it could possibly return the old value (if it existed; otherwise (char *) 0).

The second deficiency is the impossibility of deleting a variable from the environment while at the command level. At least, the Bourne Shell won't - the C Shell it still a mystery to me. I propose a routine delenv(name) which simply deletes the named variable from the environment. It will of course return an error should the variable not exist, and possibly return the old value on success.

I have not mentioned the method of implementation of this facility, as it would depend a great deal on things like average size and content of environments and the properties of the programs that use them. I can either keep two large buffers permanently allocated and copy from one to the other; or I can update the string pointers in place, dynamically allocating and releasing small buffers wherever necessary.

It may be argued that these routines are of limited use: that you would only modify the environment before exec'ing some other program. Examples of these programs are GETTY, SHELL and SU. However, I claim that within this limited application they are still useful. Who wants to go through the same tortuous process each time when modifying environments? Wheels should be invented once, then merely bought off the shelf.

Surviving with 4.1a bsd

Bill Joy
Sam Leffler

Computer Systems Research Group
Department of EECS
U.C. Berkeley
Berkeley, Ca. 94720
(415) 642-778

Introduction

This short paper describes what "4.1a bsd" is, and what you need to know to make best use of this release of the system if it is running on your machine.

What is 4.1a?

4.1a is an intermediate version of the VAX system distributed by our group. It is an *experimental* version of the system whose most noticeable improvement over the 4.1 system is its support for local networking. It also includes a few other features which you may find useful, such as "symbolic links" and an improved group scheme.

4.1a is not a full distribution of the VAX system. It is being used at a *very* limited number of sites, and is **not** being made available to the general VAX/UNIX user community.

We currently plan to have two more intermediate versions of the VAX system before the next full release. The next intermediate version 4.1b will tenatively include a higher-performance file system, a more fully integrated scheme for inter-process communication, and page-sharing virtual memory facilities. These facilities are described in a draft new system manual which is available from our group.

The rest of this document consists of a number of sections describing new commands and changes to existing facilities in the new system.

1. User commands for networking

The new system supports the DARPA standard TCP, IP and UDP communications protocols. These are used to provide for the transfer of data between machines interconnected via local networks (such as an ETHERNET) or a long-haul network such as the ARPANET. Currently most hosts on the ARPANET speak older protocols, so you are likely to find the local networking facilities to be of the most immediate use.

Most of the networking commands provided with this 4.1a release begin with a letter **r** standing for "remote". These commands are all provisional, and subject to change.

1.1. Status of the network

Each machine on your local network is kept informed by other machines as to the status of those machines. The command *ruptime* prints a status line for each machine on the local network. At our site, its output looks like:

```
monet % ruptime
arpa      up  1+10:22,  1 user,    load 0.94, 0.80, 0.70
calder       up  0:57,  0 users,   load 0.20, 0.75, 1.08
dali         up  2:23,  0 users,   load 0.08, 0.10, 0.09
ingres    up  1+17:26,  0 users,   load 0.32, 0.49, 0.67
kim       up  1+10:27,  0 users,   load 0.74, 0.75, 0.77
matisse   down 13:36
monet        up  7:01,  1 user,    load 0.73, 0.79, 0.78
monet %
```

Here "monet % " is my prompt. This command shows 7 hosts on the local network. For each host the number of users logged in, the load average and the amount of time the machine has been up (or down) is shown. As it is very late (about 3 am), there are essentially no users logged in. In fact, the users which are logged in will be shown by a

```
monet % rwho
root      arpa:ttyh9      Mar 31 23:42
root      monet:ttyp0     Mar 31 23:39
monet %
```

command. Currently this shows only me (logged in a root on the *monet* machine), and another *root* user (logged in on the *arpa* machine.)

Both of these commands suppress and do not count users who have been idle more than an hour. They both take –a commands to force these users to be printed. Thus

```
monet % rwho –a
fabry     arpa:ttyh3      Mar 30 17:59   10:51
kateveni  kim:ttyh4       Mar 30 18:42    9:10
kridle    ingres:ttyh0    Mar 31 18:48    6:32
mckusick  arpa:ttyj1      Mar 31 19:50    6:50
mosher    arpa:ttyh1      Mar 31 08:26    2:48
root      arpa:console    Mar 31 12:19    1:55
root      arpa:ttyh9      Mar 31 23:42
root      monet:console   Mar 31 20:54    6:45
root      monet:ttyp0     Mar 31 23:39
root      monet:ttyp1     Apr  1 01:45    2:11
rwh       arpa:ttyh5      Mar 31 14:34   13:20
sam       arpa:ttyh2      Mar 31 11:36   15:50
wilensky  kim:ttyib       Mar 31 11:17   10:22
wnj       monet:tty00     Mar 31 21:17    6:38
monet %
```

produces more, if rather pointless, output.

1.2. Logging in and executing commands on remote machines

It is quite easy to log in on another machine and execute commands there. You can do this by saying

monet % rlogin *othermachine*

If your current machine and the *othermachine* name that you give are under

common administration you may find that you are simply logged in on the other machine with no fuss. If not, you may get a prompt for a password.* Here, for example, is what is printed when I log into the *kim* machine from my current machine *monet*:

```
monet % rlogin kim
Last login: Wed Mar 31 20:32:25 on ttyp0
TERM = (c1004p) h19
Erase is Delete
Kill is Ctrl-U
 4:10am  up 1 day, 10:34,  3 users,  load average: 0.64, 0.67, 0.73
kim % who
kateveni  ttyh4          Mar 30 18:42
wilensky  ttyib          Mar 31 11:17
wnj       ttyp0          Apr  1 04:10
kim % date
Thu Apr  1 04:10:33 PST 1982
kim % hostname
ucbkim
kim % logout
Lost connection.
monet %
```

On the *kim* machine my login prompt includes the machine name, just as it does on the *monet* machine. I do this while having a single *.cshrc* file by using the *hostname* command, in a sequence of the form:

```
if ($?prompt) then
        set prompt="hostname|sed s/ucb//' %'
endif
```

If the machine I wish to log in on is not under the same administration as the machine I wish to log in from, then I can make it possible to log in without giving a password by creating a file *.rhosts* in my login directory on the remote machine, containing a line with the name of the machine from which remote logins are to be accepted. This name **must** be the full-name of the machine, as printed by "hostname". Thus if *kim* and *monet* are under different administrations the existence of a file *.rhosts* on *kim* in my **wnj** account's home directory containing:

```
monet % cat .rhosts
ucbmonet
ucbarpa
ucbingres bill
monet %
```

would allow remote logins from user *wnj* on machines *ucbmonet* and *ucbarpa*, and from user *bill* on machine *ucbingres*. While it is possible to use passwords after being connected to the remote machine with *rlogin*, *.rhosts* is much more convenient (and necessary for *rsh* as we shall soon see.)

Actually, there is a more convenient way to log in to a remote machine: you can include the directory **/usr/hosts** in your shell search path. This directory includes commands which have as names the names of all the machines on the local network, obviating the need to type "rlogin".

*If there is not an account on that machine with your current login name, then you will likely be prompted for a login name. Other possibilities are mentioned in *rlogin*(1x).

1.3. Remote shell

Another very useful command is the remote shell command *rsh*. This command executes another command which you specify on a host of your choice. Thus

```
monet % kim date
Thu Apr  1 04:20:36 PST 1982
monet %
```

executed the date command on the *kim* machine. Note here that I am taking advantage of the fact that the command *kim* in **/usr/hosts** invokes the *rsh* command when given arguments. This command is the same as

```
monet % rsh kim date
Thu Apr  1 04:21:28 PST 1982
```

The standard output and error output of the remote command are returned to the local standard output and error respectively, and interrupt and quit signals are propogated to the remote process. The remote commands execute in your home directory, running your normal shell and your shell startup file. You can use remotely executed commands in pipelines quite naturally. Beware of shell metacharacters such as ">" and "*" in the remote command: they will be evaluated locally unless you protect them by quoting them, perhaps by enclosing them with " characters, e.g.:

```
monet % kim "echo *"
ls.c mbox
monet % kim echo *
a.out bin changes.3-82 core.c relibig.c rt.c
monet %
```

Here the first command echoed the names of my files on *kim*, while the second command ran an *echo* command on *kim* just to print the names of my files on the current machine which is *monet*.

1.4. Remote copy

Another useful command is the remote copy command. For example to make a copy of the password file from *kim* in my current directory I can do

```
monet % rcp kim:/etc/passwd .
monet % ls -l passwd
-rw-r--r--  1 root       2962 Mar 29 17:35 /etc/passwd
monet %
```

The *rcp* command takes any number of arguments, each of which may either be a simple path name or a name of the form "host:pathname".

For 4.1a, both the *cp* and *rcp* commands take **-r** options, to recursively copy directory hierarchies. This is very convenient for moving a large number of files across the network.

2. Symbolic links and changes to ls and du

The new system includes a new kind of link: a symbolic link. This link differers from the more traditional UNIX link in that it is evaluated much later (in a programming language sense). That is, the link is actually a file containing a path name, which is substituted into the path name under evaluation when the link is encountered. The importance of symbolic links is that they can span file systems, and that they allow one to make links to directories. Eventually this

will become **very** important as it will allow links to refer to files on other machines.

A symbolic link is created by giving the **-s** option to the **ln** command, e.g.:

```
monet % ln -s /etc/passwd mypw
monet % ls -l mypw
lrwxr-xr-x  1 root          11 Apr  1 04:50 mypw -> /etc/passwd
monet %
```

Here the *ln* command created a symbolic link, and the *ls* command printed it by showing the contents of the link.

Except for a very small number of programs which take special care to notice symbolic links, the links are transparently equivalent to normal links. This list of programs which "see" links is: the *ls* command, which prints them out so you can find them; the *du* command, which reports disk usage and does not chase links; the *find* command, which likewise does not chase links; and the *rm* command, which given the **-r** option does not follow links, and in any case removes a link rather than the linked to file.

So that you can see the files actually referred to, the *ls* command takes a **-L** option to cause it to print out the files referenced by the links, rather than the links themselves, thus:

```
monet % ls -Ll mypw
-rw-r--r--  1 root        2962 Mar 29 17:35 mypw
monet %
```

One other change to the *ls* and *du* commands must also be noted here: these comands no longer print "block sizes" in 512 byte units; rather they communicate sizes in kilobytes. This change is being made for sanity, since the file system in the 4.1b version of the system will have variable size blocks in the different file systems, and prevents utter confusion.

3. Changes to the group scheme

4.1a places you in all of the user groups to which you belong. This list of group names is printed by the *groups* command; for me on *monet* it prints:

```
monet % groups
staff operator
monet %
```

showing that I belong to the *staff* and *operator* groups.

In the new system a newly created file is given the group of the directory in which it is created. Thus a group of users can share files by making a directory owned by their common group. It is also possible for users to use the *chgrp* command to change the group associated with a file which they own to any group to which they belong.

4. Changes for programmers

A number of new system calls and library routines have been added in 4.1a to provide a provisional interface to the networking facilities. As this interface is temporary, we are not providing a detailed introduction to the use of these facilities here. If you are interested in using the facilities, read the new pages in sections 2, 3, and 4, and find out from your system administrator the name of someone at your site who has learned about the new calls and interfaces. If you can, look at some of the programs we provided to use the network. But in any

case beware! We are already planning the demise of a number of the hooks which are provided now, in favor of the interface described in the "4.2bsd system manual" which is available in draft form from our group.

We should note, however, a couple of incompatibilties between 4.1 and 4.1a. First, the signal SIGTINT and the multiplexor facility *mpx*(2) are removed. The new facilities described in the attached section 2 manual pages have so far proved adequate to replace these.

Finally, to prepare for the future, read and use the facilities of *directory*(3) and *gethostname*(2x). These facilities should be used to build programs which don't depend on fixed length file names, or on the host machine on which they are running. In particular, 4.1b bsd will likely support 255 character file names and support binary compatibilty with 4.1bsd machines.

Appendix: changed files

This is a list of the files, outside of the /usr/man and /usr/src directories, which have changed in this 4.1a system. While we have made every attempt to make this list complete, there may be a few omissions; don't take it as gospel.

For each file we indicate whether the impact of the change to the file is minor, major (e.g. noticeable to normal users), or that the file is new or has been deleted. The synopses here are very cryptic, sorry about that.

/bin/adb	minor	changed to support kernel debugging
/bin/as	minor	fix bug in input buffer handling
/bin/cat	minor	fixed because pipes aren't files
/bin/chgrp	major	normal users can do chgrp
/bin/cp	major	now has a -r flag doing recursive copies
/bin/df	minor	fields made wider
/bin/du	major	now prints sizes in kilobytes
/bin/hostname	new	prints/sets name of current host system
/bin/ln	major	can create symbolic links
/bin/login	minor	changed for remote logins
/bin/ls	major	rewritten for symbolic links
/bin/mail	minor	machine independent; no /dev/mail
/bin/ps	minor	recompiled for new system
/bin/rm	minor	doesn't follow symbolic links
/bin/sh	minor	newgrp command removed
/bin/su	minor	machine independent
/bin/wall	minor	machine independent
/bin/who	minor	machine independent
/bin/write	minor	machine independent
/dev/MAKE	major	now creates pseudo-terminals
/dev/mail	dead	should be removed
/dev/pty[pq]?	major	new pseudo-terminals for network
/etc/comsat	minor	no longer uses mpx
/etc/config	minor	handles new kernel source
/etc/dmesg	minor	recompiled for new kernel
/etc/fsck	minor	understands about symbolic links
/etc/ftpd	new	ftp protocol server
/etc/getty	minor	machine independent
/etc/hosts.equiv	new	defines other systmes with same users
/etc/hosts.local	new	gives name of local system/local networks
/etc/implogd	new	error logging daemon for arpanet interface
/etc/init	minor	fixed bug in handling of open lines
/etc/mount	minor	cleaned up and new -f and -r options
/etc/pstat	minor	prints out pty's with -t
/etc/rc	minor	now invokes /etc/rc.local for per-site stuff
/etc/rc.local	new	new adjunct to /etc/rc for local stuff
/etc/rdump	new	dump that will dump to another machine
/etc/rexecd	new	daemon for rexec(3x)
/etc/rlogind	new	daemon for remote logins with rlogin(1x)
/etc/rmt	new	protocol handler for /etc/rdump
/etc/route	new	command to make routing table entries
/etc/rshd	new	daemon for remote executions with rsh(1x)
/etc/rwhod	new	daemon for rwho(1x) and ruptime(1x)
/etc/shutdown	minor	machine independent
/etc/telnetd	new	daemon for telnet protocol
/etc/ttys	minor	add pty's so who(1) will show remote users

/etc/ttytype	minor	add pty's
/etc/wall	minor	machine independent
/etc/whod.*	new	new data files created by rwhod
/lib/crt0.o	minor	new version written in C
/lib/libc.a	minor	new version with new syscalls
/lib/mcrt0.o	minor	new version written in C
/usr/bin/find	minor	doesn't follow symbolic links
/usr/bin/iostat	minor	recompiled for new system
/usr/bin/newgrp	dead	replaced by new group facilities
/usr/bin/sdb	minor	recompiled for new system
/usr/include/errno.h	minor	now includes more error numbers
/usr/include/ident.h	dead	should no longer be used
/usr/include/net	new	should now be a symbolic link
/usr/include/ndir.h	major	see directory(3)
/usr/include/sys	major	should now be a symbolic link
/usr/include/vadvise.h	minor	added a new definition used in kernel
/usr/include/whoami	dead	should no longer be used
/usr/include/whoami.h	dead	should no longer be used
/usr/include/signal.h	minor	no more SIGTINT
/usr/hosts	major	commands for connection to local machines
/usr/lib/adb	minor	adb command files for kernel debugging
/usr/lib/hosts	minor	new host table
/usr/lib/tmac/tmac.an.new	minor	revised .UC macro for 4.1a bsd
/usr/local/emacs	minor	new version needed since no mpx(2)
/usr/local/emacs/maclib	minor	new library for emacs
/usr/ucb/ftp	new	file transfer command
/usr/ucb/groups	new	new command printing group memberships
/usr/ucb/implog	new	prints out log created by /etc/implogd
/usr/ucb/man	minor	understands new 1x, 2x, 3x and 4 sections
/usr/ucb/more	minor	machine independent
/usr/ucb/page	minor	see /usr/ucb/more
/usr/ucb/rcp	new	remote file copy
/usr/ucb/rlogin	new	remote login
/usr/ucb/rsh	new	remote shell commands
/usr/ucb/rstat	new	remote interface status
/usr/ucb/ruptime	new	remote uptime status
/usr/ucb/rwho	new	remote who is logged in
/usr/ucb/script	major	now uses pseudo-terminals
/usr/ucb/telnet	new	arpa telnet protocol user program
/usr/ucb/trpt	new	tcp trace log print routine
/usr/ucb/uptime	minor	recompiled for new system
/usr/ucb/vmstat	minor	recompiled for new system
/usr/ucb/w	minor	recompiled for new system
/usr/ucb/whereis	minor	machine independent

Perkin-Elmer Supports Unix
Or
A Philosophical View of Unix
in the Commercial Environment
Or
Will Success Spoil Unix?

Joseph Longo
Perkin-Elmer, Melbourne

The recent announcement by Perkin-Elmer Australia to offer Unix as a standard supported product has been received with mixed sentiments. From the PE viewpoint, Unix offers a dual-pronged attraction :-

- it offers a sophisticated time-sharing development environment for PE's full range of superminis, an environment not offered by PE software until now;

- it also acts as a catalyst with the ability to bring PE's traditionally low profile out into the lime-light.

Relative to the outside world, the PE/Unix coupling is seen as :

- legitimizing Unix as a viable operating system to be taken seriously;

- a chance for a large cross-section of computer users to combine the fabled Unix development environment with low cost, high performance 32 bit hardware. (on a PE 3210 : 8 megb/sec DMA throughput plus 0.94 Mips, concurrently for around $60k.)

But there is the negative view as well :-

- elements of the old guard within PE argue that the Unix offering is unnecessary. The argument is almost true in that there does not appear to be an overwhelming demand for Unix locally. But compare this to the U.S. where hardware vendors not able to offer Unix are out in the cold.

- a typical commercial Cobol-bound user will give you a million reasons why he doesn't need Unix. But have him work on it for a while and he'll give you two million why he can't live without it ! (Is there bliss in ignorance ?).

Very few people would seriously suggest that Australia will not follow trends in the U.S. where the adoption of Unix has snowballed. Even the worst of the Unix critics/cynics who last year were claiming Unix was useless because its I/O was not record-oriented, are this year sheepishly admitting that Unix will probably gain a foothold locally.

There are some aspects to the commercial use of Unix that are worth mentioning. Firstly, it brings the University computer departments closer to the commercial users in the real world. The two may not sleep together, but they will at least live together. The common criticisms that fly between universities and industry will become less frequent. Both of them, for the first time, will be talking the same language : Unix.

As a direct consequence, graduates will be immediately productive in their first jobs. They won't need to be 'de-tuned' or 're-trained' (brain-washed) before becoming acceptable to their employers. Job adverts may look something like :-

WANTED
PROGRAMMER/ANALYST
Must have Unix expertise,
preferably with tertiary qualifications;
Hardware experience irrelevant;
Promotion to SU plus access to
"/usr/games" after qualifying period.
Cobol programmers need not apply.

The flow of software between universities and industry will become a two-way street. Universities will make available a host of Unix-related software which until now was mainly academic, while industry will offer its own packages to the universities giving undergrads first-hand experience with the type of environment in which they are likely to work.

As far as PE is concerned, this software exchange will be something to look forward to. Hopefully, some of the PE commercial packages will be ported to Unix. The PE optimizing Fortran compiler, recognized as being one of industry's best, should be the first candidate, followed by other compilers. The PE transaction-processing/data-base package, Reliance, itself a highly successful development tool, would be an ideal Unix application. Take, then, some of the local and U.S. Unix enhancements (e.g. Melbourne U. mods, Berkeley mods, etc.) and plug them into PE's Unix. The result of all this commercial-academic interchange is a formidable operating system and development environment. Run it on a high powered PE supermini and you get a lot of people sitting-up to attention.

The concept of packaging Unix to sell in the commercial environment is an abhorrence to many Unix followers. At the same time, it creates the ground-work needed to bring the commercial users forward from 1956 and academic users down out of orbit. Acceptance of Unix by Perkin-Elmer sets a precedent which makes all of these discussions adopt a touch of realism. Undoubtedly, a lot of these forecasts have already taken place in the U.S.. With a bit of encouragement the same environment can be created in Australia within the immediate future.

SUN Microsystems, Inc.
2310 Walsh Avenue
Santa Clara, CA 95051

408-748-9900

Thank you very much for your interest in the SUN Workstation. SUN Microsystems, Inc. was formed in February 1982 to manufacture, sell, and support this high-performance, low-cost graphics workstation.

The SUN Workstation† is a high-performance graphics workstation aimed primarily at scientific, engineering and CAD/CAM applications. It consists of a high-resolution bitmap display, an Ethernet local network interface, and a powerful processor executing the UNIX‡ operating system. Using the Intel Multibus‡‡ as the input/output bus, the SUN Workstation is expandable with a variety of peripheral controllers and off-the-shelf board-level components. The system is packaged as a desk-top unit with a detachable keyboard. A mouse pointing device may be optionally connected for graphical input.

SUN Workstations are interconnected via the Ethernet into clusters forming a distributed computing system. A typical SUN Cluster† combines five to ten SUN Workstations with a fileserver, printer server, backup server, etc. This gives each user a high-quality display and a powerful local processor while sharing the cost of the peripherals (disks, tapes, printers, etc.). It also provides a quiet and cool office environment because the peripherals can be located remotely.

SUN Microsystems is porting the Berkeley 4.2bsd UNIX to the SUN. This allows a full UNIX system to be run on SUN Workstations without local disks, accessing files and programs over the network. 4.2bsd supports very high-performance networking and file system. In addition, it includes a large number of utilities as part of the system. A FORTRAN and a PASCAL compiler is also available. The system is scheduled for release in 1Q83 and will be made available to betasites in 4Q82.

Currently available software for the SUN Workstation is a UNIX Version 7 distributed by Unisoft. To run this UNIX, a disk is required for each workstation. This system inlcudes some of the Berkeley UNIX enhancements, such as *vi*. An optional Pascal and Fortran compiler is available. SUN Workstations shipped with UNIX V7 will be field upgradable to the 4.2bsd UNIX system through a hardware/software upgrade package.

The workstation can also be used as a programmable graphics terminal. A PROM-based DEC VT100 emulator and a Tektronix 4014 emulator is available, as well as an implementation of the SIGGRAPH CORE standard of graphics routines.

The basic SUN Workstation, consisting of keyboard, display, and processor, and graphics terminal software, has a list price of $ 8,900. A SUN Workstation equipped with a 12 MByte Winchester disk (6 MByte fixed, 6 MByte removable), SMD Disk controller, a 128K Multibus Memory Board, and V7 UNIX costs $ 19,900. Lead time is 60 to 90 days.

† SUN Workstation and SUN Cluster is a trademark of SUN Microsystems, Inc.

‡ UNIX is a trademark of Bell Laboratories. ‡‡ Multibus is a trademark of Intel Corporation.

MICROSOFT™

THE XENIX™ OPERATING SYSTEM

DIFFERENCES BETWEEN V7 UNIX OS and XENIX OS

(Microsoft's Value-Added to UNIX)

Obviously, it takes a significant effort to take a research-oriented minicomputer operating system and turn it into a general-purpose commercially viable product — even if it IS portable. THAT is Microsoft's value-added to the UNIX OS. The Table below briefly summarizes the improvements and enhancements made to the UNIX OS to date in creating XENIX.

Differences between XENIX and UNIX are either classified as IMPROVEMENTS or ENHANCEMENTS. Improvements consist of changes that are made to enhance reliability and performance of the OS but are transparent as far as the user software is concerned, i.e. they make what is already there work better. Enhancements on the other hand, are new features to the OS which applications software can use.

Microsoft is committed to continually improve and enhance XENIX in accordance with the needs of the marketplace and the interests of our OEMs. Some of the improvements and enhancements that are currently implemented appear below. The order and timing of future enhancements will greatly depend on the needs of the OEMs as we adapt XENIX to their systems.

XENIX Improvements	XENIX Enhancements
* XENIX Has Been Ported to the Z8000, 8086, 68000	* Record and File Locks
* Automatic File System Recovery	* Semaphores
* Performance-Tuned to Micro's	* Absolute Priority Assignment
* Device Error Logging	* Scatter-Loaded Kernel
* Interactive System Configuration	* Non-Blocking "reads"
* Miscellaneous Bug Fixes	* Synchronous (blocking) "writes"
	* Enhanced Inter-Process Communications

XENIX OS Improvements to UNIX

XENIX Has Been Ported

The most obvious advantage to working with XENIX is that it HAS been ported. When implementing XENIX on a Z8000, 8086, or 68000 an OEM STARTS with XENIX for the Z8000, 8086, or 68000.

Because Microsoft IS porting XENIX to multiple processors, and adapting XENIX to multiple memory management configurations, XENIX has been made more portable by isolating (and where possible removing) machine dependencies. This makes XENIX more uniform, more dependable, and easier to support across system lines than is standard UNIX or UNIX only ported once in the most expeditious fashion. This also means that companies planning to port UNIX from the PDP-11 or VAX (the only versions available from AT&T) to proprietary processors will find it easier to start with XENIX.

Automatic Disk Recovery

One of the most widely voiced concerns about UNIX is the difficulty it can have with file recovery after a power failure. In improving the UNIX file system, Microsoft has put in several features and utilities as well as general design modifications to allow the system to recover the file system automatically from a crash, performing the functions that UNIX's designers assumed would be handled by a systems programmer.

Bad Spots on Disk

Another typical complaint about standard UNIX is that it has never been taught how to deal with disks with bad spots. On the PDP-11 and VAX, AT&T simply buys disks guaranteed to be perfect. The Winchesters used on micro's offer no such option. There are two ways bad spots can be handled. The best way is for the disk controller to actually map around the bad spots. XENIX device drivers have been modified to make use of this feature in a controller if it exists. If the controller has no such capability, then XENIX has the ability to map around the trouble spots entirely in software.

Size and Speed

Two of the more critical characteristics of any operating system are its size and speed of execution. Microsoft is applying its experience in compiler optimization and code generation to optimize the C compilers used in supporting XENIX on various hosts. The size of the code generated by these compilers will vary depending on the power of the instruction set of the target machine.

To further reduce the size of XENIX and improve its efficiency on machines that do not support separate Instruction and Data space (e.g., the LSI-11/23), the XENIX kernel and user space have been overlayed using a dynamic memory map technique, allowing a full-feature XENIX to operate in the limited addressing space of the non-I&D space machines. Prior to overlaying the kernel (which takes up 70Kb memory on the larger PDP-11's), it had to be stripped down to 48Kb to run on the smaller machines. This reduced the utility of the OS and reduced the number of users that could be handled comfortably to one. The version of XENIX Microsoft has running on the 11/23 now will support four users.

Performance-Tuned to Micro's

Standard UNIX, and the more popular University versions of UNIX (e.g., Berkeley UNIX) are being developed on PDP-11's and VAX's. Consequently, much of the work done on these developments is done with that environment in mind. One of the key implications of this development environment is that the software that is developed will be free to use disk I/O rather indiscriminately since CPU and memory resources are relatively dear on these systems and disk I/O is extremely fast. Quite the opposite is true on micro's. These small machines are typically I/O bound with CPU and memory to spare. The Winchester disks on these systems are painfully slow and inefficient in comparison with the disk drives on the DEC gear.

In tuning XENIX to the micro environment, Microsoft has identified portions of the UNIX software where these types of trade-offs have been made. The overhead of the OS has been reduced by modification and/or replacement of various internal algorithms and parameters. For example, the character and sector buffering scheme is being modified to significantly increase the data transfer rate. The swapping algorithm is also being modified to make more judicious use of the memory available and reduce the amount of swapping to a minimum.

Hardware/Software Integrity

To help the user maintain hardware integrity, Device Error Logging has been added to XENIX. This feature generates a log file of hardware errors allowing the user to recognize that a device may be going bad before it causes a serious problem. In addition, XENIX is being made more intelligent to avoid panic situations.

System Configuration

To make it easier for the user to generate a new XENIX configuration, there is an interactive configurator which will allow the user to select device drivers, specify machine type, and adjust several key parameters such as swap device, root device, number of disk buffers, etc. CONFIGURE will then automatically generate and install a new XENIX system.

Floating Point

Under UNIX, each program that makes use of floating point must have the floating point routines linked in. Under XENIX, floating point has been moved to the kernel, so there is never more than one copy on the system, nor do programs need to change based on the presence of floating point hardware.

Miscellaneous Bug Fixes

In porting XENIX multiple times and adapting it to multiple memory management configurations, Microsoft has subjected UNIX to far more rigorous use and testing than ever before. This process, along with making the OS more portable, has flushed out a number of UNIX bugs that might never have been found in normal use on a PDP-11. In addition, Microsoft has drawn upon the extensive UNIX user base to identify and fix scores of bugs in UNIX as distributed by AT&T.

XENIX OS Enhancements

While Improvements have made XENIX a commercially-rigorous system, it is Microsoft's Enhancements that will make UNIX commerically-viable. Microsoft is committed to work with its OEMs in anticipating standard commercial operating system requirements, and implementing those features seen as necessary in an upward-compatible fashion. Microsoft will evaluate the features requested by OEMs (including inclusion of UNIX software from University sources such as Berkeley). Those seen as generally beneficial to the XENIX market will be merged for Bell UNIX compatibility, commercialized, and incorporated in the standard XENIX product.

One area needing early attention in commercializing UNIX is support for network and database (especially transaction-processing) environments. The initial enhancements described below are designed to give the systems developer the tools needed to address these environments with systems built on XENIX.

Record and File Locks

One of the more desirable features in a multi-user commercial operating system is that it be able to arbitrate between multiple access requests to the same record or file. Programs running under XENIX are able to request access allowing locks to be extended to a single record, group of records, or the whole file. The program can request Read-Only access, Write-Only access, or Multiple Read access. The lock request can return immediately if not granted due to another user's lock request, or the second requestor could be put to sleep until the request can be granted.

Inter-Process Communications

There are several enhancements being considered to improve XENIX's inter-process communications abilities. The first of these to be implemented is an in-memory message buffer which is a device driver that allows multiple-process access to whatever information has been placed there for the process's use. This feature allows communications between non-related processes since each process's message box is identified by a logical name (much like a file).

Semaphores

For process synchronization, XENIX incorporates a semaphore mechanism. The semaphores provide an interlock and simple communication facility for multi-task synchronization. Ungrantable semaphore requests will cause the process to be put to sleep on a queue until the resource becomes available.

Scatter-Loaded Kernel

Under standard UNIX, a process must reside in contiguous physical memory. This means that if the system wants to load a 128Kb process but does not have a 128Kb chunk of contiguous free memory, the system must swap something to make room. The situation is further aggravated if one or more processes have been locked into memory (not allowed to move or swap). Process locking is often done for those very critical processes that HAVE to be serviced in a relatively real-time fashion. Unfortunately, locking processes in memory will tend to fragment memory, making it even MORE difficult to provide contiguous memory for a process without swapping. On systems supporting memory management with a page-mapped architecture, XENIX allows a process to reside in NON-contiguous physical memory, thus eliminating the need to swap as long as there is enough memory to run the process — regardless of the memory's location.

Non-Blocking Reads

A call has been added so that a process can tell if a read call directed to a specific open file (or the user's terminal) will suspend for lack of data. This call can be used, for example, to poll multiple input files (terminals, etc.). Under standard UNIX, if a process issues a call to read data from a device and data is not yet available, the process will hang until the data is provided.

Synchronous (Blocking) Writes

A synchronous mode can be set so that all pending (but buffered) writes will be completed, and all subsequent write requests will suspend the caller until actual completion of the I/O transfer. This mode guarantees file completeness, order of write execution, and timely notification of I/O errors. This mode can be separately set on each individual open file. With this feature, database and transaction processing applications that depend on having data written out in the order input can be accommodated.

Microsoft Support

Microsoft's most significant value-added where its OEM's are concerned is on-going XENIX support. Introducing a product to address the XENIX market is as complex an undertaking as has ever been faced in the small computer marketplace. Microsoft provides its OEM's with support through all stages of system development.

Hardware for these new systems is several orders of magnitude more complex than the 8-bit systems. No longer can a manufacturer put a chip on a board with a bit of memory and call it a computer. The primary area of difficulty on the part of manufacturers migrating up from the 8-bit world, is in the understanding of memory management issues and multi-user systems design. Microsoft provides the functional consultation on these and other topics to its XENIX OEM's.

For OEM's doing their own ports or adaptations, Microsoft provides the tools developed internally to assist in the process.

Microsoft is developing a series of new documents for XENIX. All these documents will be available to XENIX OEM's. They range from educational seminars, to marketing materials, to system administrators guides to improved documentation for programmers.

Last but not least by any means, XENIX OEM's will gain the advantage of the continual maintenance and development being done at Microsoft on XENIX.

UNIX System III Support

At the COMDEX show in Las Vegas in November 1981, AT&T announced the release of UNIX System III. There are three major benefits to the new release: 1) The first (and by far most significant benefit of the System III package) is new pricing. The UNIX royalties are now ⅓ to 1/10 of what they were under Version 7. 2) New software. The Programmer's Workbench and VAX versions of UNIX are now included in the System III license. 3) New features. UNIX now contains several utilities to make it easier to bring up and take down. It also contains new interprocess communications facilities and miscellaneous bug fixes.

Microsoft has already upgraded its licensing of XENIX to the new System III licensing program. Even though the current release of XENIX is basically Version 7-derived, XENIX is licensed as a System III product giving XENIX customers full benefit of the improved licensing terms offered by AT&T. The XENIX royalty schedule has been lowered to reflect the new prices offered by AT&T. The new software will be offered to OEM's as part of the XENIX package. They will decide on how they will be offering it. Many of the new features and utilities are capabilities Microsoft has already added to the XENIX product as part of our enhancements to UNIX Version 7 and are therefore already available. The capabilities in System III not currently in XENIX will be integrated into the product throughout the last half of 1982.

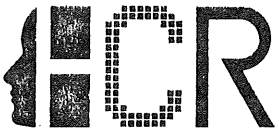
LANGUAGE SUPPORT

Over time, Microsoft will be making all its standard language offerings available under XENIX. Since the process of rewriting this software in a high-level language for portability is a long, slow task, not all languages will be available at the same time over all processors. It is anticipated that all versions of XENIX (except possibly the PDP-11 versions) will have full language support by the end of 1982. All versions of XENIX are currently delivered to OEM's with a C compiler as one of the standard utilities. Other languages to be made available separately are BASIC Interpreter, BASIC Compiler, COBOL, FORTRAN, and Pascal. All these languages will be largely upwards compatible with their 8-bit versions. This will allow applications written in Microsoft languages on other machines to be adapted to run under XENIX with minimal effort.

"What is C? The de facto standard for C is D.M.Ritchie's compiler. The portable C compiler by S.C.Johnson (which runs on a number of dissimilar machines) is remarkably close to the Ritchie compiler. The compilers for C are permissive in that they compile and run more programs than the C reference manual would allow. The permissiveness of the compilers is balanced by a program called lint, which checks C programs and complains about dubious constructions ranging from type mismatches to possibly non-portable constructions. Lint is restrictive in that it complains about programs that the C reference manual allows.

The C programming language therefore lies somewhere between the compilers, lint, and the C reference manual"

Ravi Sethi
in "A case study in specifying the semantics of a programming language"
7th Annual Symposium on
Principles of Programming Languages



Human Computing Resources

Toronto - HCR has announced that it has commenced its UNIX* development effort to port the UNIX* operating system to the National 16032 microprocessor.

HCR has started its development effort in cooperation and with full support of National Semiconductor, Santa Clara, CA., the manufacturer of the microprocessor.

It is expected that the project will result in a complete UNIX* system (based on System III release from AT&T) to be available to end users and OEMS early in 1983.

HCR announced that this development places the National 16032 microprocessor with its unique demand paging under UNIX* in a competitive position with the other major microprocessors, the Z8000, Intel 8086, and MC68000 for which versions of XENIX (UNIX*) have been developed by Microsoft of Bellevue, WA.

For further information, please contact:

Edward Borkovsky
Vice President, Marketing
Human Computing Resources Corporation
10 St. Mary Street
Toronto, Ontario M4Y 1P9
Canada
Tel. 416-922-1937

*UNIX is a trademark of Bell Laboratories.

From 8239088 Thu Jun 17 07:55:02 1982
To: abby
Subject: what is UNIX, what is IBM, what do they stand for ?

From abby Thu Jun 17 23:15:05 1982
To: 8239088

UNIX stands for freedom, beauty, and elegance.
IBM stands for the money, anti-trust, and lawyers.

Seriously, it is allegedly an acronym for "International Business Machines", "International Big Mother", "I've Been Moved", and "International Buttock Manipulators", (according to various informed sources). There is also another firm called "Itty-bitty machines", but that is not the one most commonly referred to.

UNIX is a trade mark of Bell Laboratories.
USA is an trade mark and wholly owned subsidiary of IBM.

I've had far too little mail lately, and most of it is unsuitable to print, for various reasons ranging from unintelligibility to obscenity. Some of it is simply too distressing -- the computer industry seems to have some attraction for the lunatic fringe. Only last week I received a letter from a reader asking if there was a UNIX "shell" available which simulated a CP/M environment. The man was obviously in drastic need of professional help. Every day we meet people who bear permanent mental scars from their FORTRAN-based education. Everywhere we see fellow computer workers whose understanding, skills and ambitions are fixated at far too low a level, due to the dominating influence of a particular language or piece of hardware. Only last month a programmer wrote to me asking how he could arrange to be buried with his PDP-11. Or take the following plea from the wife of a DP manager: "My husband burns the hairs out of his nose with a cigarette lighter and he thinks I'm crazy because I use a manual typewriter." Or the man who hid his wife's dentures so that she couldn't go out and buy a Basic compiler. TIME magazine recently cited the case of a married man whose wont it was to sunbake naked on the roof of his apartment with his personal computer.

Sometimes the stresses of a computing job become too taxing, and overworked employees should take a break before the revulsion reaches critical levels. Programming has been compared to playing chess for a living, and also to cocaine addiction for its "burnt out at 30" syndrome. The following excerpt from a reader's letter highlights the crisis experienced by many people: "I'm a systems programmer and I want some information on how to become a shepherd".

This sort of problem can be nipped in the bud by early attention. That's why I'm here. So PLEASE, let's have some newsworthy personal problems. No photographs please.

Abby

Is Your Plug Compatible?

Join the Network Revolution!

Date: 22 Oct 1982 1033 (Friday)
From: chris
To: auugn
Subject: dave horsfall
Full-name: Chris Maltby

------- Forwarded Message

Replied: <<22-Oct-82 10:32>>
Date: 22 Oct 1982 0958 (Friday)
From: dave:csu40
To: chris:basservax
Subject: benchmarks
Full-Name: Dave Horsfall
Room: 1408
Building: Law Library
Tty: ttyo
Terminal: Teleray 1061
Position: Seated
Weather-outside: Fine and sunny
Condition: Still a bit sleepy

I'll forward it on to Veronica at Maths - she can try it single user.

------- End of Forwarded Message

From: ucbarpa!CSVAX:G:asa
Newsgroups: net.general,net.chess
Title: Can You Top This?
Article-I.D.: populi.198
Posted: Mon Jun 7 23:30:54 1982
Expires:
Received: Mon Jun 7 23:34:17 1982

>From the SAN FRANCISCO CHRONICLE (Mon., June 7, 1982; reprinted from the WASHINGTON POST):

Russia Off Limits

U.S. IMPOUNDS CHESS COMPUTER

Washington

The U.S. Customs Service, intent on stopping the flow of sensitive technology to the Soviet Union, has seized and impounded indefinitely a machine called Belle, the world-champion chess computer.

Belle won the title in 1980 at the most recent world computer chess championship tournament in Linz, Austria.

The Commerce Department says the computer might be of military use to Moscow. The frustrated scientist who wanted to take it to a Moscow chess exhibition -- and now isn't sure he'll even get it back -- has a different view: "The thing plays chess. That's all."

Customs officials said a squad of special agents spotted Belle's computer case bout three weeks ago t John F. Kennedy International Airport in New York. The destination stamped on the crate: Moscow.

Agents quickly detained the shipment and sent for instructions from Washington. Customs then turned to the Commerce Department, whose International Trade Administration decides whether a piece of equipment might be of use to the Russians. In this case, the answer was yes.

The seizure of Belle is part of Operation Exodus, a major new program to halt what officials have called a "hemorrhage" of the nation's best technology to the Soviet Union and its allies.

Customs officials are delighted with the new program, which they say has tripled the number of seizures of illegal exports of sensitive equipment and technology. They say Exodus has produced 1150 leads and 370 seizures, including computers, aircraft parts and communications equipment, in its first nine months.

The Commerce Department would not comment on why the chess computer could be considered militarily sensitive, but Kenneth Thompson, the scientist at Bell Laboratories who was responsible for the shipment, says the only way it could be used militarily would be "to drop it out of an airplane. You might kill somebody that way."

A spokesman at the Commerce Department said Thompson would be subject to a penalty for violation of the Export Control Act. The possibilities range from a cash fine to losing the computer altogether.

Thompson said the parts of the machine said to be sensitive are available for purchase in this country. "I just don't see the point of all this," he said.

auugn?

(Message netmail:4)
Date: 12 Apr 1982 1956 (Monday)
From: richardg:elecvax
To: abby:basservax
Subject: Competition for the next AUUGN

Here are some simple competitions that might interest your
otherwise idle UNIX readership.

1)
What is the shortest sequence of binary characters that do not
occur in any of your local file systems?
What is the shortest printable sequence that does not occur anywhere?
(This string might have all sorts of temporary uses until people
find out what it is).

2)
Using only Bourne or C shell (its probably a nightmare with pwb shell),
write an equivalent to...
find dir -name pat -a -print
That is, given 2 arguments (a directory and pattern), print out
all filenames that conform to that pattern JUST LIKE FIND.
Of course you cannot use find itself, but can use any other
existing not-too-obscure commands to implement your find. The order
in which the filenames are printed does not matter. Remember, to
be like find, the pattern must use SHELL(!) glob characters.

For the second question, I warn you, it is not easy. In fact, even
though we have thought up some really terse and 'elegant' solutions,
none are EXACTLY like find. There is at least one "one line" semi-
solution, and another that uses no external to the shell command
except "echo".
        Good luck
        Richard Grevis
(Message netmail:5)
Date: 19 Apr 1982 1413 (Monday)
From: dave:unswcsu
To: abby:basservax
Subject: Eunuchs

Why is it that the Unix User's Meetings are almost entirely male ?
Could it be that the sexual connotations (or lack thereof !) of
"UNIX" turns women away ?

AUUGN is produced by volunteers from the Australian Unix Users Group. To sustain the fine standard of journalism which our discriminating clientele have come to expect (or will soon, anyway), we solicit material from readers. This means YOU! Yes YOU! Send your material to the editors at the addresses given below. We prefer to be netsent the unformatted file, but hard copies are gratefully accepted and reproduced intact. Also, material should be in the public domain.

Now about that other thing. Money. We need to increase our readership, so get anyone you can to join the Australian Unix Users Group. For a mere $24 you will receive six issues of AUUGN over a period of one year. Please send your cheques in Australian currency. Do not send purchase orders. So mail your subscription fee to this address:

AUUGN
c/o Bob Kummerfeld
Basser Department of Computer Science
Madsen Building F09
University of Sydney
NSW 2006
Australia

Electronic corresponence should be addressed to:

auugn:basservax (on the SUN)
mhtsa!australia!auugn (in USA)