AUUGN

Australian Unix User Group Newsletter

Volume 5 Number 4

Registered by Australia Post Publication No. NBG6524

The Australian UNIX* Users Group Newsletter

Volume 5 Number 4

July 1984

CONTENTS

* UNIX is a trademark of Bell Telephone Laboratories.

Editorial

I have reproduced the draft AUUG constitution at the end of this issue. This draft has been prepared primarily by the convenor of the interim executive committee, John Lions, with input and comments from other members of the committee. Please read the constitution since we hope to have it approved at the Melbourne meeting.

I have also included some forms which will become valid when the constitution is adopted and membership costs are finalised.

AUUG Meeting in Melbourne

Final details have been mailed out, but if you missed out refer to the end of this newsletter.

Contributions

As usual, more contributions please!

Opinions expressed by authors and reviewers are not necessarily those of the Australian UNIX Users Group, its Newsletter or the editorial committee.

Books

There are another four books to add to the list. If you are not keeping your list up to date, don't worry, I will publish the complete list in the last issue of volume 5.

Books on UNIX

16. A Practical Guide to the UNIX System
Mark G. Sobell
The Benjamin/Cummings Publishing Co. Inc. (1984)
(Australian Distribution through Addison-Wesley Publishing Co.)

17. A Practical Guide to XENIX
Mark G. Sobell
The Benjamin/Cummings Publishing Co. Inc. (1984)

18. A Practical Guide to UNIX System V
Mark G. Sobell
The Benjamin/Cummings Publishing Co. Inc. (1985)

Books on C

10. The C Programming Tutor
Leon A. Wortman and Thomas O. Sidebottom
Prentice-Hall (1984)

Related Books

6. Principles of Compiler Design
Alfred V. Aho and Jeffrey D. Ullman
Addison-Wesley Publishing Company (Talks about Yacc and Lex)

Nets

The following pages contain what amounts to a world network map. It is left as an exercise for the reader to cut them out and fit them together in the most imaginative way.

Usenet Logical Map
July 24, 1984

Below is the Usenet Logical Map, done by Bill and Karen Shannon, for July 1984. The Usenet Logical Map is based on the directory sent out in net.news.map at the beginning of July by Mark and Karen Horton and incorporates all changes and corrections through July 22. The map is considerably different than it has been in the past. Due to the growth of the network, it has been split into several different pieces (nine to be exact). Connections to sites on other pages of the map are in upper case, with the page appended to the name of the site (e.g., DECVAX.6). The groupings are roughly geographical or by organization.

Usenet is defined as all sites receiving newsgroup net.announce. A Usenet link between two sites is one that net.announce is sent over. Note that this is different from a UUCP link, over which mail and file transfers may occur but not necessarily news.

The map is up to about 940 sites. The June 1984 map had about 900 sites; the previous Logical map (done in July '83) had only about 550.

Reading the map: -, |, and + indicate connections. Lines never pass through a +, which always indicate where the connecting line ends or changes direction. Logical buses are represented by =. This means that a site such as sitel is indicated by the entire string of characters as follows:

```
+====+=sitel=+====+
```

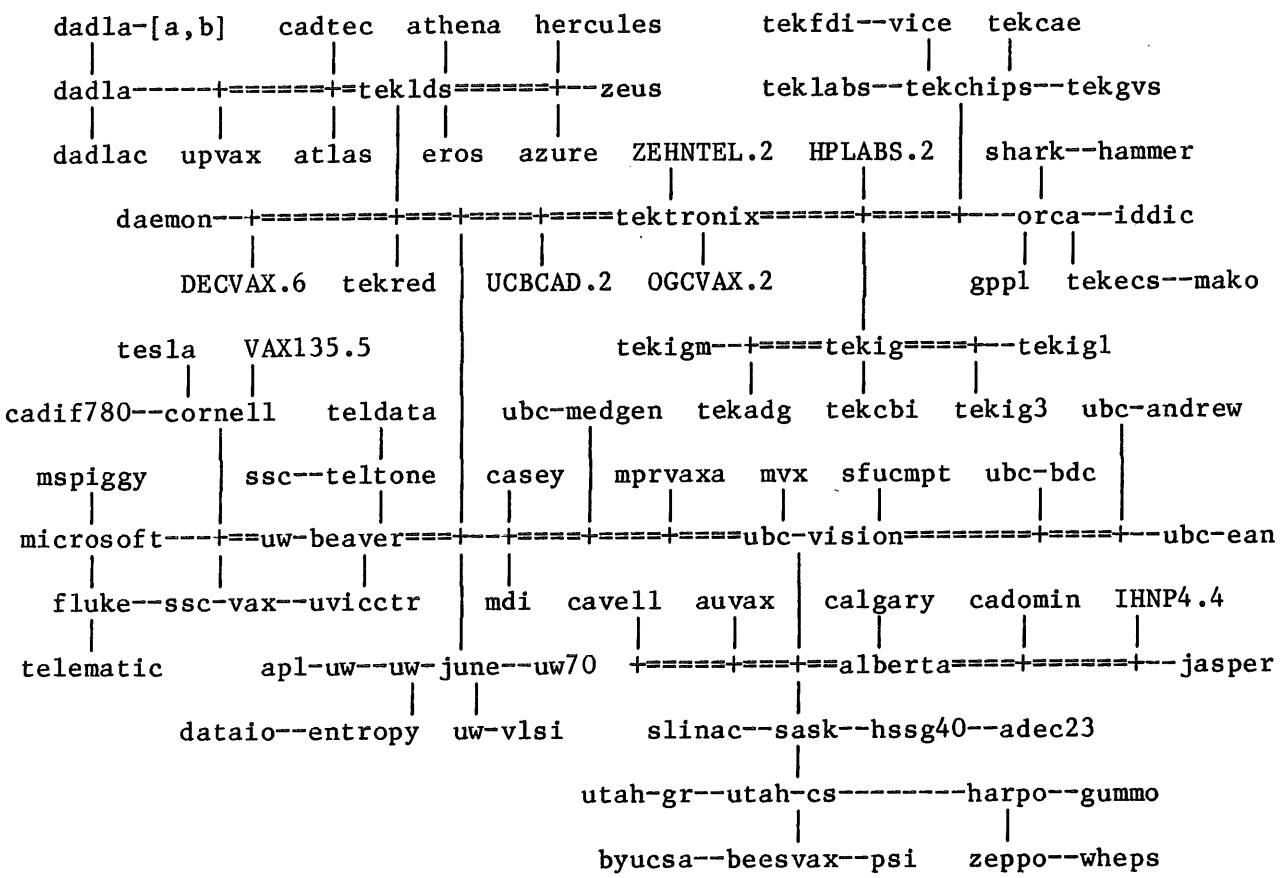
We've added one notation to this map: off-page connections are occasionally shown as branches off of a vertical line from a site. So,

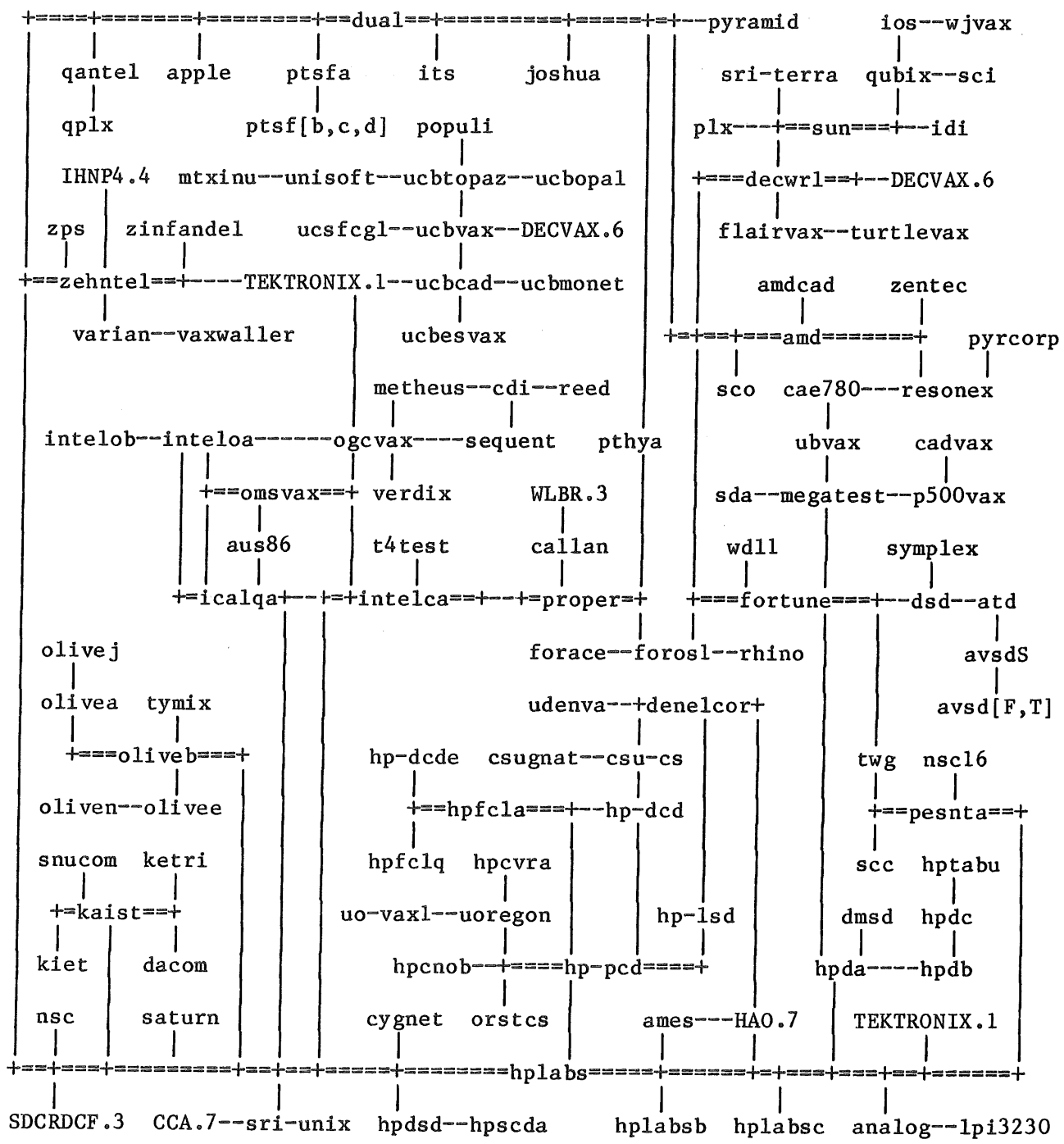
```
|--SITE2.4
|--SITE3.1
|
+====+=sitel=+====+
```

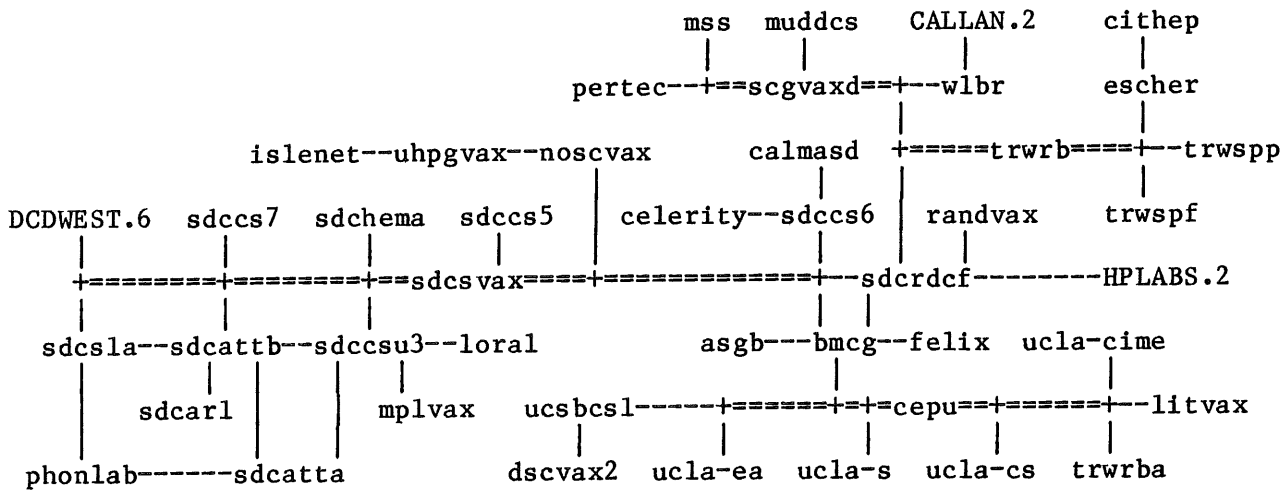
indicates that sitel has off-page connections to both site2 and site3.

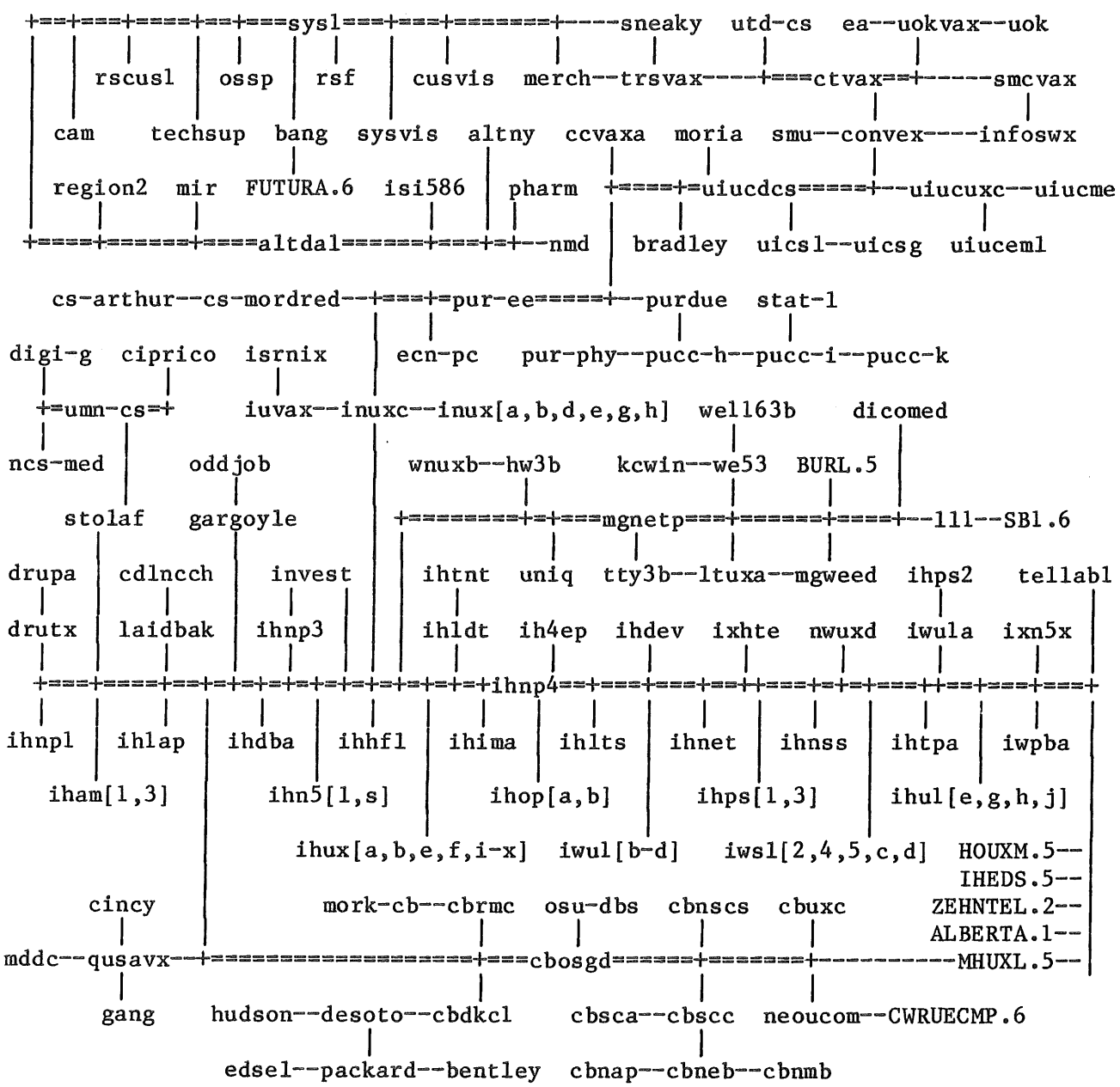
If you have any comments or suggestions, please send mail to the address below. Corrections should be sent to cbosgd!map. Thanks again to Mark and Karen for their help.

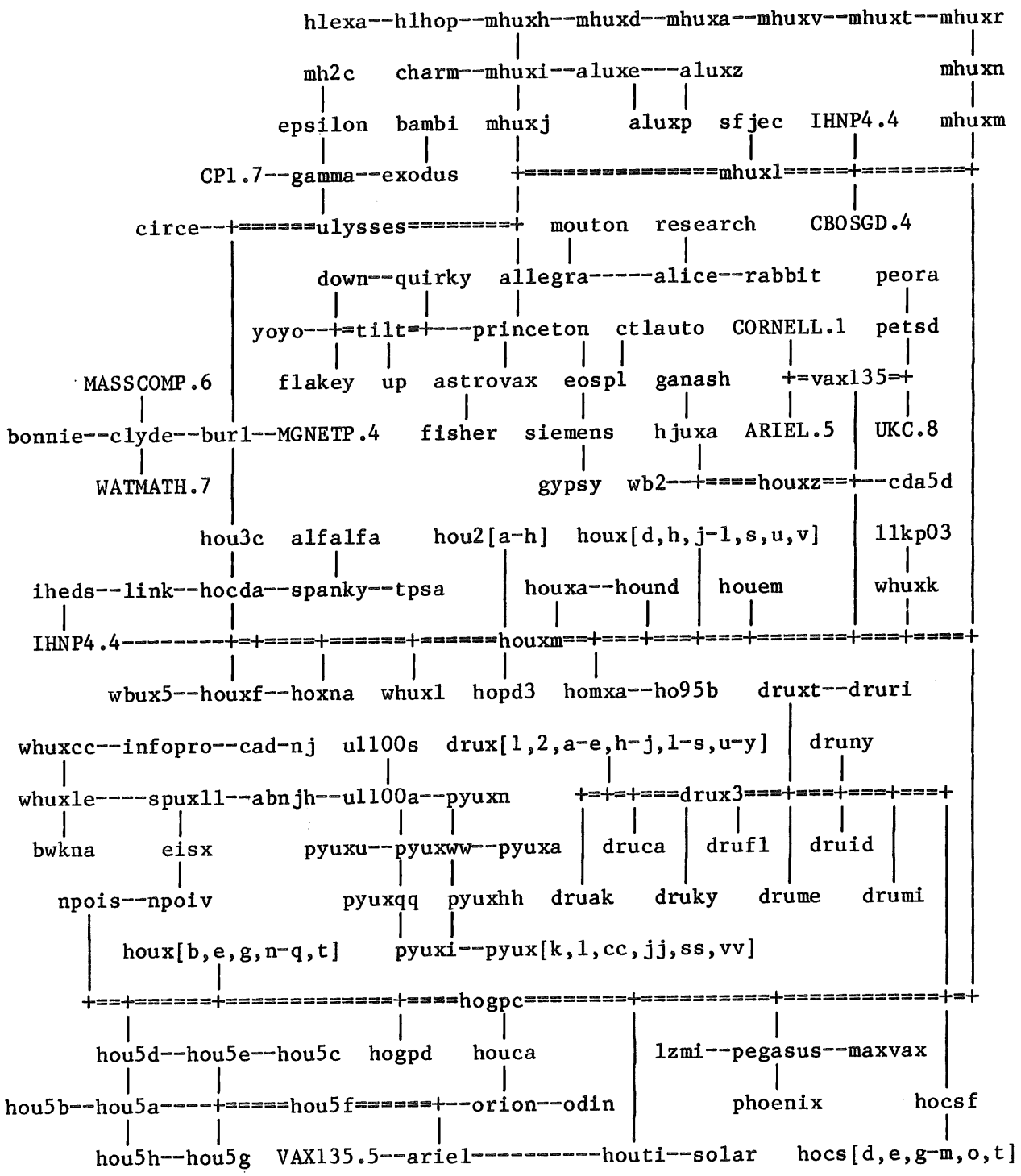
Bill and Karen Shannon
sun!{shannon,kas}

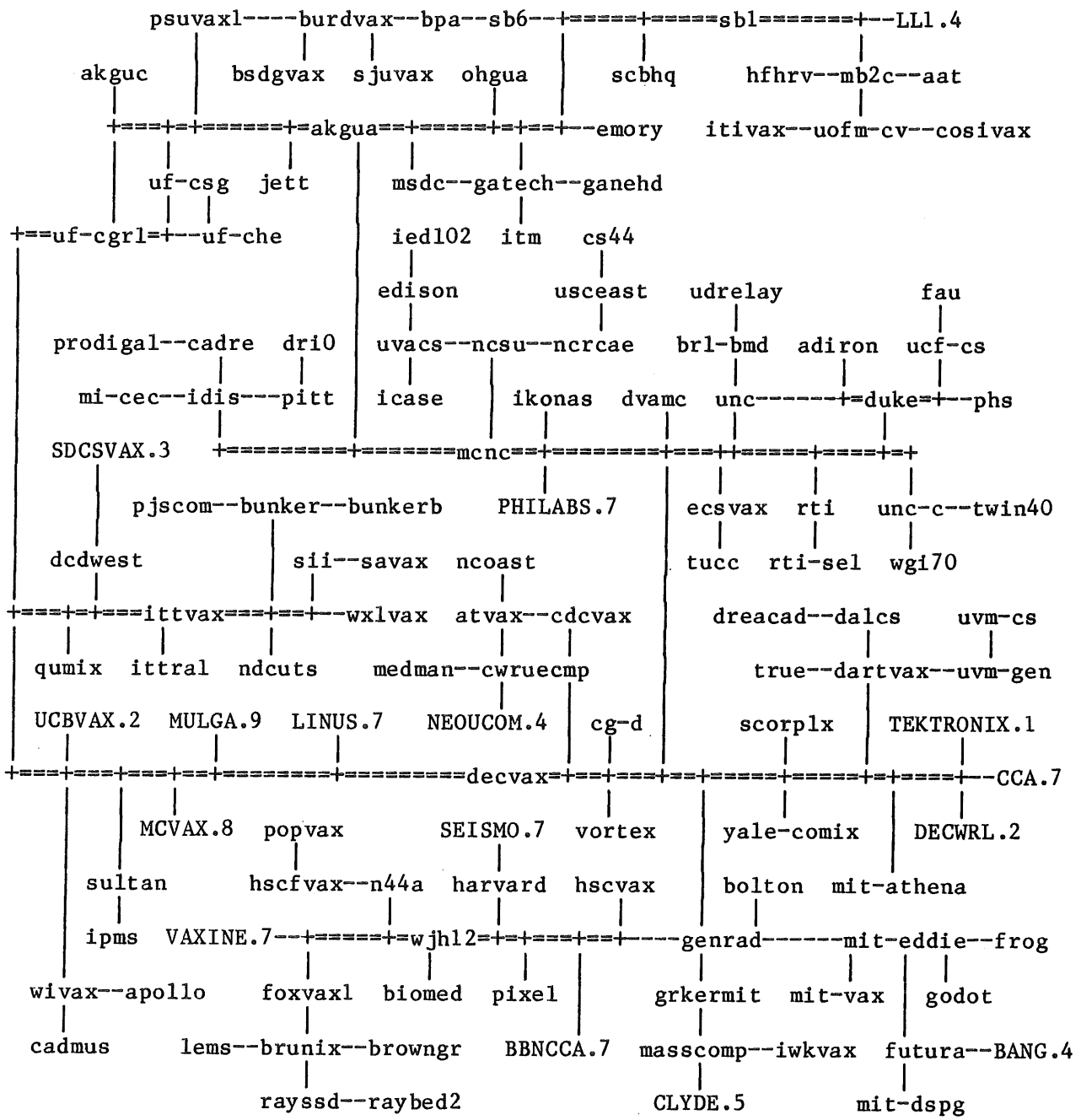

dadla-[a,b]
cadtec
athena
hercules
tekfdi--vice
tekcae
dadla
teklds
zeus
teklabs--tekchips--tekgvs
dadlac
upvax
atlas
eros
azure
ZEHNTEL.2
HPLABS.2
shark--hammer
daemon
tektronix
orca--iddic
DECVAX.6
tekred
UCBCAD.2
OGCVAX.2
gppl
tekecs--mako
tesla
VAX135.5
tekigm
tekig
tekig1
cadif780--cornell
teldata
ubc-medgen
tekadg
tekcbi
tekig3
ubc-andrew
mspiggy
ssc--teltone
casey
mprvaxa
mvx
sfucmpt
ubc-bdc
microsoft
uw-beaver
ubc-vision
ubc-ean
fluke--ssc-vax--uvicctr
mdi
cavell
auvax
calgary
cadomin
IHNP4.4
telematic
apl-uw--uw-june--uw70
alberta
jasper
dataio--entropy
uw-vlsi
slinac--sask--hssg40--adec23
utah-gr--utah-cs
harpo--gummo
byucsa--beesvax--psi
zeppo--wheps

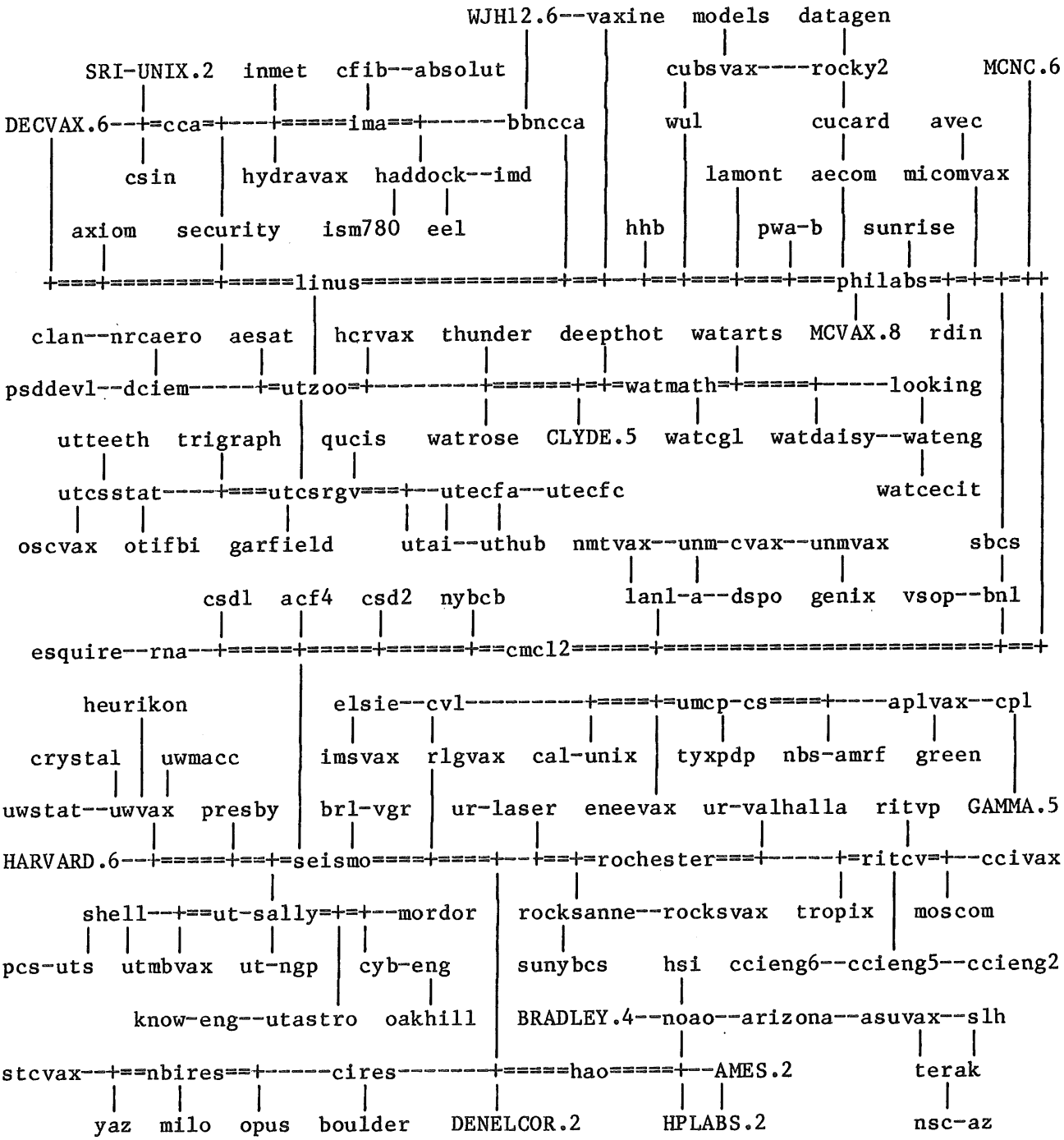


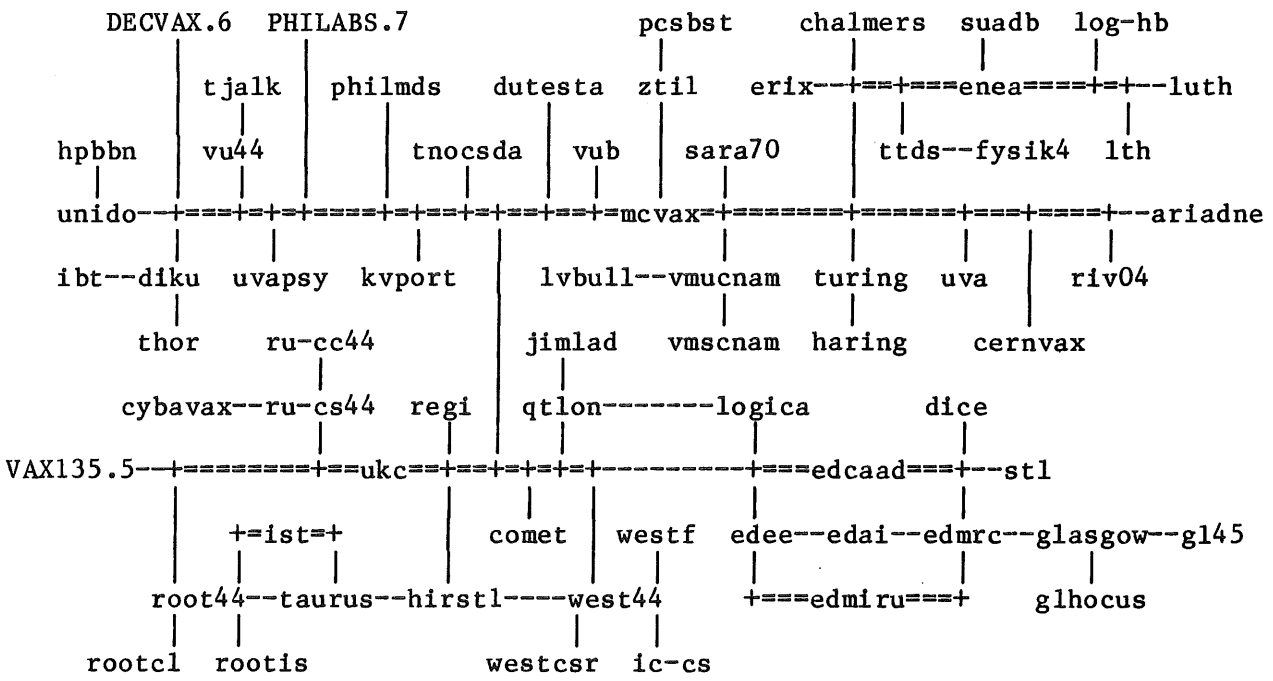

mss
muddcs
CALLAN.2
cithep
pertec
scgvaxd
wlbr
escher
islenet
uhpgvax
noscvax
calmasd
trwrb
trwspp
DCDWEST.6
sdccs7
sdchema
sdccs5
celerity
sdccs6
randvax
trwspf
sdcsvax
sdcrdcf
HPLABS.2
sdcsla
sdcattb
sdccsu3
loral
asgb
bmcg
felix
ucla-cime
sdcarl
mplvax
ucsbcsl
cepu
litvax
phonlab
sdcatta
dscvax2
ucla-ea
ucla-s
ucla-cs
trwrba

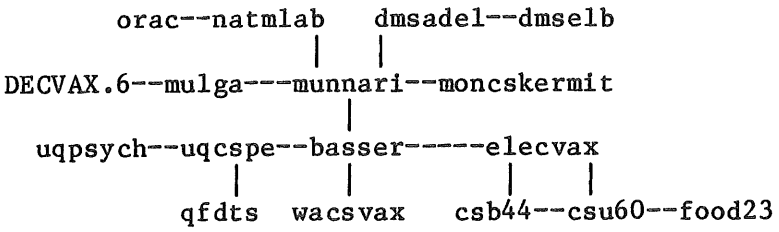


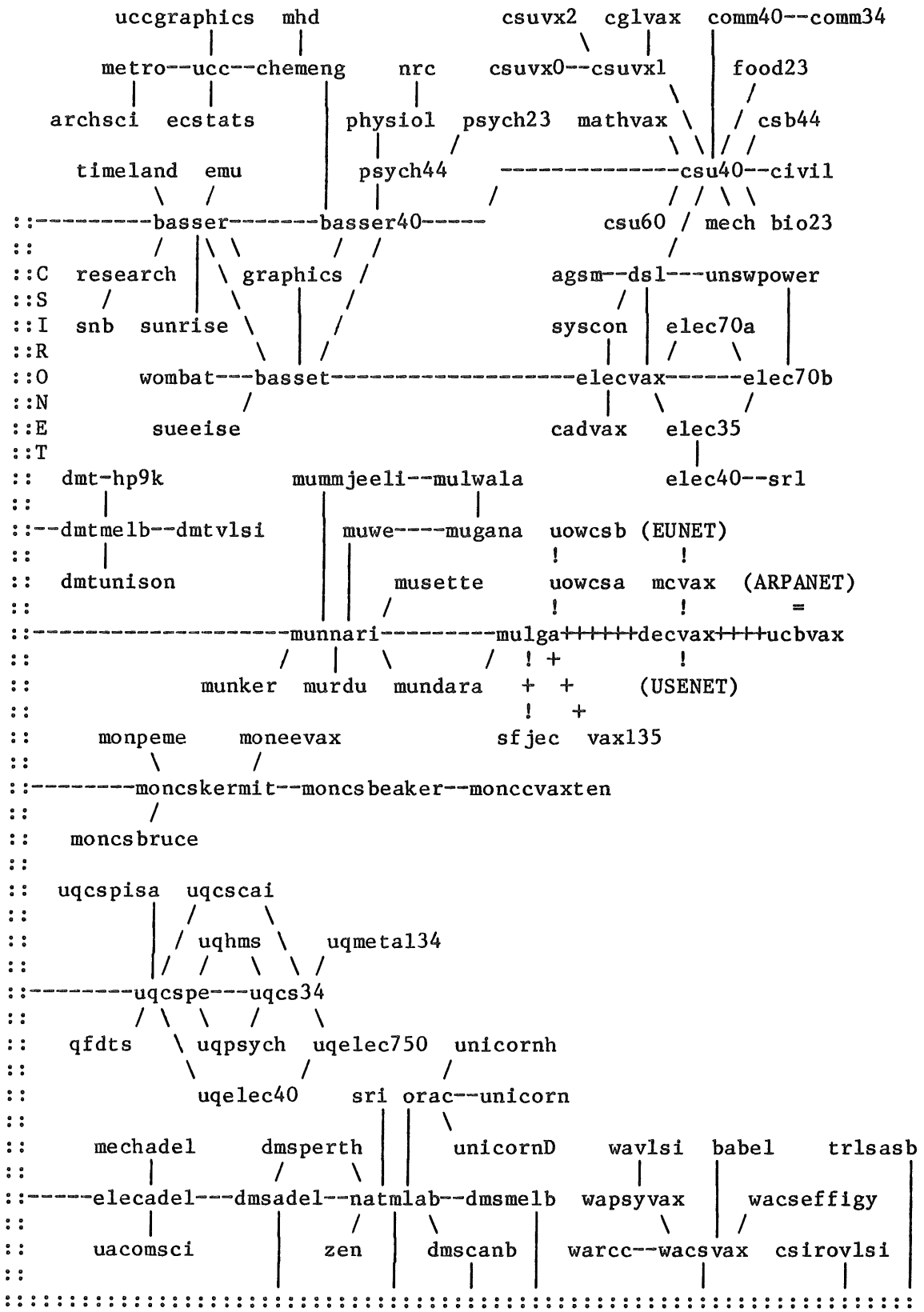

hlexa--hlhop--mhuxh--mhuxd--mhuxa--mhuxv--mhuxt--mhuxr
mh2c
charm--mhuxi--aluxe---aluxz
mhuxn
epsilon
bambi
mhuxj
aluxp
sfjec
IHNP4.4
mhuxm
CP1.7--gamma--exodus
mhuxl
circe--ulysses
mouton
research
CBOSGD.4
down--quirky
allegra-----alice--rabbit
peora
yoyo--tilt---princeton
ctlauto
CORNELL.1
petsd
MASSCOMP.6
flakey
up
astrovax
eospl
ganash
vax135
bonnie--clyde--burl--MGNETP.4
fisher
siemens
hjuxa
ARIEL.5
UKC.8
WATMATH.7
gypsy
wb2--houxz--cda5d
hou3c
alfalfa
hou2[a-h]
houx[d,h,j-l,s,u,v]
llkp03
iheds--link--hocda--spanky--tpsa
houxa--hound
houem
whuxk
IHNP4.4
houxm
wbux5--houxf--hoxna
whuxl
hopd3
homxa--ho95b
druxt--druri
whuxcc--infopro--cad-nj
ul100s
drux[1,2,a-e,h-j,l-s,u-y]
druny
whuxle----spuxll--abnjh--ul100a--pyuxn
drux3
bwkna
eisx
pyuxu--pyuxww--pyuxa
druca
drufl
druid
npois--npoiv
pyuxqq
pyuxhh
druak
druky
drume
drumi
houx[b,e,g,n-q,t]
pyuxi--pyux[k,l,cc,jj,ss,vv]
hogpc
hou5d--hou5e--hou5c
hogpd
houca
lzmi--pegasus--maxvax
hou5b--hou5a----hou5f--orion--odin
phoenix
hocsf
hou5h--hou5g
VAX135.5--ariel
houti--solar
hocs[d,e,g-m,o,t]






DECVAX.6
PHILABS.7
pcsbst
chalmers
suadb
log-hb
tjalk
philmds
dutesta
ztil
erix
enea
luth
hpbbn
vu44
tnocsda
vub
sara70
ttds
fysik4
lth
unido
mcvax
ariadne
ibt
diku
uvapsy
kvport
lvbull
vmucnam
turing
uva
riv04
thor
ru-cc44
jimlad
vmscnam
haring
cernvax
cybavax
ru-cs44
regi
qtlon
logica
dice
VAX135.5
ukc
edcaad
stl
ist
comet
westf
edee
edai
edmrc
glasgow
gl45
root44
taurus
hirstl
west44
edmiru
glhocus
rootcl
rootis
westcsr
ic-cs




uccgraphics
mhd
csuvx2
cglvax
comm40--comm34
metro--ucc--chemeng
nrc
csuvx0--csuvx1
food23
archsci
ecstats
physiol
psych23
mathvax
csb44
timeland
emu
psych44
csu40--civil
basser
basser40
csu60
mech
bio23
research
graphics
agsm--dsl---unswpower
snb
sunrise
syscon
elec70a
wombat---basset
elecvax
elec70b
sueeise
cadvax
elec35
dmt-hp9k
mummjeeli--mulwala
elec40--srl
dmtmelb--dmtvlsi
muwe----mugana
uowcsb (EUNET)
dmtunison
musette
uowcsa
mcvax (ARPANET)
munnari
mulga
decvax
ucbvax
munker
murdu
mundara
(USENET)
monpeme
moneevax
sfjec
vax135
moncskermit--moncsbeaker--monccvaxten
moncsbruce
uqcspisa
uqcscai
uqhms
uqmeta134
uqcspe---uqcs34
qfdts
uqpsych
uqelec750
unicornh
uqelec40
sri
orac--unicorn
mechadel
dmsperth
unicornD
wavlsi
babel
trlsasb
elecadel---dmsadel--natmlab--dmsmelb
wapsyvax
wacseffigy
uacomsci
zen
dmscanb
warcc--wacsvax
csirovlsi

ACSNET - The Australian Alternative to UUCP

Piers Dick-Lauder

Basser Dept of Computer Science
University of Sydney

R.J. Kummerfeld

Basser Dept of Computer Science
University of Sydney

Robert Elz

Dept of Computer Science
University of Melbourne

ABSTRACT

ACSNET is a network with goals to serve a function similar to that currently served by the UUCP network. Routing is implicit, and addressing absolute, with domains. The network daemons attempt to make use of full available bandwidth on whatever communication medium is used for the connection. Messages consist merely of binary information to be transmitted to a handler at the remote site. That handler then treats the message as mail, news, files, or anything else. Intermediate nodes need not consider the type of the message, nor its contents.

1. Introduction

ACSNET is a loosely coupled network of heterogeneous machines, and has a purpose and function similar to that provided by the UUCP network in wide use in most of the UNIX* world.

Before continuing we must make two points. First the matter of naming. Perhaps ACSNET's biggest problem is the lack of a suitable name. The developers (PD-L and RJK) call the software `The Sydney Unix Network' or SUN for short, while the network built using SUN is called ACSNET. Unfortunately, `SUN' may be confused with a Unix based workstation of the same name and `ACSNET' suggests some relationship, or at least similarity with the U.S. CSNET network. No other label has gained sufficient approval to catch hold, so ACSNET serves for the time being.

Second, a note on our use of UUCP for comparisons with ACSNET in this

* UNIX is a Trademark of Bell Laboratories.

paper. The authors have no intention to discredit UUCP, or to belittle its achievement in linking the world's UNIX systems in a manner never before attempted. However, its presence as a current de-facto standard for UNIX to UNIX communications places it in a position where we cannot avoid making comparisons in order to illustrate certain points with far greater economy of words than would otherwise be possible.

2. Overview

ACSNET provides a message passing service, from one host to another, possibly utilising intermediate hosts in a store and forward manner. Messages may be mail, files, printjobs, news, or almost anything that can be transferred in a string of bytes.

Routing in ACSNET is implicit, users need only be concerned about the name of the host at which the message is to be delivered. They need not be concerned about which hosts the message might visit on its journey to its final destination. If, for some reason, a message is undeliverable, the network will make every attempt to return the message to its original sender.

Messages can be transported over any medium capable of supporting a connection between two hosts. This may be phone lines, ethernet, X.25, twisted pairs, etc. Preferably, the link should provide a transparent 8 bit data link, but links with only 7 useful data bits can be accommodated (at a slight loss in throughput). The network daemons make good use of full duplex communication channels, transferring messages in both directions simultaneously, providing, in ideal conditions, effective throughputs up to twice that attainable with UUCP.

3. Messages and Handlers

A message is a string of bytes addressed to a handler at one or more hosts. A handler is a process that will receive the messages at the final destination. Typically the handler will impose some further protocol, often recognising a user name (in some form of representation) that the message is directed to. A message, of itself, is addressed merely to a host and a handler.

The notion of messages addressed to handlers is one of the primary differences between ACSNET and UUCP. UUCP functions as a remote command execution system built upon a file transfer protocol. Mail, news, etc. are transmitted by sending the content of the item to the remote system as a file, then sending a request to execute the remote mail, or news, receiver with the file as standard input. ACSNET simply transfers a message addressed to the mail handler on the remote system.

ACSNET uses trailer protocols, where the header follows the message. This allows file copying to be avoided on intermediate hosts when routing statistics are updated. ACSNET only ever performs disk to disk copying of a message when it is being copied to its final destination, and optionally, when queueing the message in the first instance.

Currently handlers exist for mail, news, file transfer, and remote printing. A remote command execution handler could be added if the security issues could be adequately solved. Any other handler could be created just as easily, for any purpose that the sending and receiving hosts agree upon.

Unlike UUCP, it is not necessary for intermediate hosts to know of the new

handler for correct functioning.

4. Addressing

All messages carry a destination host address. This is the ASCII name of the destination host. Messages also contain a source host address, and may contain a user address. This last item may be anything that the handler requires for its functioning.

Messages may be addressed to more than one host. A copy of the message will be sent to each host addressed, and that will be done with the minimum possible message traffic (or something approaching it). A message may also be broadcast to all hosts. This is most often used for network management messages, such as new hosts connecting, and similar events.

Users also have the option to guide their message through a specific set of hosts. Primarily this is used for network testing; loopback messages would otherwise be impossible to create.

Such a route is expressed in a notation borrowed from UUCP as

host-1!host-2!host-3 ...

However, note that there is a fundamental difference between this form of addressing and UUCP addressing. Each host-N is the absolute address of some site. The ! does not imply a link between two adjacent hosts. In the above example, the message will visit, in order, host-1, host-2, and host-3. But the route taken to travel from host-2 to host-3 is not specified, and in fact, given a suitable topology, the message might travel that route via host-1!

Strictly, in all the above, the term `host name' should be replaced by `domain specification', but that is harder to type, read, say, and think about. Any of the places where a host name was specified, ACSNET would really expect a domain specification. Domains might simply be host names, or they might specify local sub-domains, or perhaps a domain that is not within the ACSNET network, in which case the message will be sent to an appropriate gateway.

Note, nowhere here has it been specified what syntax should be used by users in communicating with user agent programs. It is to be expected that

user@domain

will be the most common format, though the older Australian net syntax of

user:host

will be supported into the distant future. Almost none of the ACSNET code either knows or cares what syntax users will use to send messages.

5. Routing

Each host maintains two tables† to contain network information. The first of these is the network state file, and contains for each known host, a list of the domains to which it belongs, a list of the hosts to which it is directly linked, and the cost and current status of each of those links. One of the domains is special, and is considered to be the primary domain of the host.

The table can also contain various other information, such as a human understandable description of each host, and statistics on messages and bytes sent, received, and passed through. This information is optional, and

† For `table' read `file'.

probably would be deleted on a small host.

The state table is considered public information, and is sent to any host that requests it. It is broadcast to all hosts whenever a new link is added.

The second table is the routing table. This table indicates which link should be used to transmit a message bound for any host or domain on the net. It is built from the state table, usually whenever that is altered. As each message arrives on a link it is passed to a routing process. That process passes it to its appropriate handler if this is (one of) its final destination(s), and queues it to be transmitted on the next link if the message has further to go. The routing table is used to make this decision.

Information on which links to transmit a broadcast packet that originated at any particular host is also retained here. This is arranged so that broadcast packets travel over a minimum traversal of the graph, and implements Dalal and MetCalfe's extended reverse path forwarding algorithm.†

This table also contains miscellaneous information, such as local aliases for hosts on the network. It is private information, and is not exported to other hosts.

Routing messages are broadcast to all hosts in the sender's primary domain whenever a link changes status (goes up or down). These are brief messages, indicating the nature of the change, and carry a timeout age, after which they are deleted wherever found. This works well, as typically, distant parts of the net are not interested in local changes, which often might have become outdated before the message reached the host. Rerouting to avoid links that are down can usually be handled by nodes relatively close to the broken link.

6. Gateways

The routing table for any local link can indicate that a non-standard spooling program should be used to send a message over a designated link, or to deliver a message to a particular domain. This can be used to build interfaces to newer, or older, versions of the network software, or to implement a gateway to a foreign network. The spooling program is responsible for performing any transformations required of the message to meet the standards required by messages entering the new network.

This performs admirably when interfacing to a network with similar capabilities, but is less of a success connecting to UUCP for anything but mail, as there is no standard way of performing possibly multi-hop file or news transfers.

7. Calls and Daemons

ACSNET uses node to node daemons to transfer messages from one host to another. Unlike the UUCP uucico process, ACSNET daemons have no knowledge of how, or when, to connect to a remote host, except in the trivial case where that is accomplished by opening a tty compatible special file. To handle other cases, the routing table may contain the name of a process to run to establish a connection to another host. That process is expected to make the connection, then exec the daemon with standard output open (read-write) to the

† Y. K. Dalal and R. M. MetCalfe, CACM, Dec. 1978.

remote host. This permits easy expansion to a wide variety of possible connection types.

A pair of daemons transfer data between themselves over three channels in each direction simultaneously. This allows up to 6 messages to be in flight between any pair of nodes at any one instant. Messages are assigned to one of the three channels based upon their size. This allows small messages to overtake larger ones on another channel, and prevents those extraordinary delays that can occur when a particularly huge message is being transferred.

The daemons also keep track of the current position in each message in transit. This allows messages to be restarted (usually) without transmitting data that had been received correctly at its destination, should a link die prematurely, or a system crash.

Messages on each channel are sent via a windowed packet scheme, similar to that used by HDLC (and UUCP, and X.25), and are checksummed using the standard CCITT CRC-16 algorithm. This checksumming can be disabled for a link if it is known to be reliable, typically if ACSNET is used over another protocol. Messages can also contain end to end checksums, to guard against corruption while waiting at an intermediate node.

8. Status

ACSNET is currently being used on VAX 11/780 and 11/750 processors, PDP-11/40 and 11/34's, Sun Workstations, Perkin-Elmer, Plexus (P60), and other less widely known machines. These processors are variously operating under V7, 4.*BSD, System III, and System V. A VMS version has been suggested, but at this stage, not attempted.

There are about 100 hosts on the Australian network, and though most of those currently use the previous software, most will probably convert in the near future.

9. Todo

There will be a message disassembly, reassembly facility, to permit huge messages to be transmitted without overloading intermediate nodes.

A UUCP gateway is needed. This is hard because of problems with multi-hop file transfers.

Some tuning remains to be done. One possible improvement on System III and V systems, and 4.2BSD, would be to use the available interprocess communication mechanisms (named pipes, sockets) to allow the routing process and handlers to become daemons, and be created just once, rather than once per message.

10. Availability

The source code is available under license. Anyone interested should apply to Bob Kummerfeld at the address above.

11. Conclusions

ACSNET is a suitable network system for connecting comparatively large networks, of a size comparable to the UUCP network, that may operate in a relatively unmanaged environment. Its implicit routing makes it considerably easier to use than standard UUCP, and more accurate and adaptable than the

heuristic UUCP routing algorithms now becoming available.

We feel that ACSNET would be suitable as a general replacement for UUCP.

On the design of the UNIX* operating system

Peter Collinson

Computing Laboratory
University of Kent,
Canterbury, Kent, UK
ukc!pc

ABSTRACT

An extremely inaccurate view of the history of the design of the UNIX operating system is presented.

Why is UNIX successful?

The computer world seems to have gone `UNIX mad´, and it is hard to understand why. One good reason is the portability of the system but there must be more to it than that. Most people who use the UNIX system seem to like it even though it is full of idiosyncrasies, is terse to the point of unhelpfulness and consists of a very large number of totally forgettable commands. I think that the success of the system is summed up by the following paragraph.

> The UNIX system is successful because the minimum number of keystrokes achieve the maximum effort. In addition, the system says very little to explain errors and relies on the intelligence of the user to deduce reasons for failure.

The statement describes UNIX V6, which we all know is the parent of the UNIX systems running today. History tells us that the guys who designed it did their own typing into the machine. It seems to me that because of this, the main reason that UNIX enjoys/suffers from terse input and output is not through any intellectual design decisions made at some early stage but because the UNIX designers were just bad typists working on slow peripherals.

Let us examine the evidence.

Ancient history

First, there are all those incomprehensible and forgettable two letter commands: **ls**, **cp**, **mv**, **pr**, etc, etc. What delight it was to type three letter commands: **cat**, **cdb**, and **dsw**. It is interesting to note that if you were to use the text formatters, it was assumed that you could type. So, to use **nroff** you had to make five painful keystrokes, admittedly only four different letters were used. Of course, all the commands were in lower case because the designers couldn´t find the shift key on the keyboard, never mind the shift-lock.

---

* UNIX is a Trademark of Bell Laboratories.

Secondly, look at the main means of the input of text, another two letter command, **ed**. Every command in the editor is a single letter, which are just about mnemonic. Every error message is a single letter, `?´, which is just about meaningless‡. However, only a very few commands are needed to do basic editing operations. If the user wants to do something complicated, then something complicated must be typed in. This complicated thing is often very short and incomprehensible to anyone other than the author. Is there anything more minimal than regular expressions in **ed**? Learning to use **ed** is a continuous voyage of discovery, users find that less and less typing is required as time goes on. What joy it is to finally find out how \(is used. **Ed** is written in C and is available for hacking, and the alterations made to it by many hackers in many places were thought to be required to make the editor more powerful. That is, you could do more in less keystrokes. So, **ed** is aimed at bad typists, and it developed to support bad typists.

C is another example of design for bad typists; integer? - no, int; structure? - no, struct; begin/end? - no, `{ }´; `:=´? - no `=´, after all it´s easy to type `==´; long variable names? - yes, but only 8 characters were significant, so it wasn´t worth typing any more; and the list could go on and on. C is so terse that it is enigmatic, and was obviously designed for coding by bad typists who are actually typing the code themselves. Most other languages seem to be designed for people who are paid by the number of inches of code they write on paper for other people to input into the machine.

Apart from the famous patented setuid bit, the notion of the use and exploitation of software tools is the main feature of the UNIX system which has become important and perhaps even academically respectable. But, did those guys really sit down over a cup of coffee one day and say ``hey, I´ve had this really neat idea, let´s code a bunch of simple programs onto this experimental file system, provide some way of joining the output of one program to the input of another - and yes, a good name would be UNIX´´. Heck no, one of them said to the other ``I´m really up to here with this typing, can´t we use the output of that program into the input of this, it´ll save me lots of time, which I can spend in playing some really neat games´´. So, software tools were born.

So, the evidence is very strong that bad typing played a large role in the design of the UNIX system. With its terse input and terse output, UNIX V6 was a joy to use - I am a bad typist too.

More recent history

A number of improvements were seen in UNIX V7. Notably, better typists were employed to design and implement things. The Bourne shell, **sed** and **awk** are examples of this. Those original guys had got a bit better typists too, there are a few more comments in the kernel and some longer names. C had altered and generally requires more typing. All the new things have long names: **typedef**, **unsigned** variables, and the type checking of function returns meant the input of function specifications at the start of the program.

UNIX was ported to a number of machines using the very wordy portable C compiler. However, ports of the system to machines with long names were not successful and the enduring port was to the VAX architecture, because VAX is

‡ OK folks, I didn´t forget about the immortal TMP error message and the fact that `??´ means something special.

short enough for everyone to type. And then came the Computer Systems Research Group of the University of California, Berkeley.

It is very fashionable to be rude about UCB systems these days. This rudeness all stems from typing envy, because those people at UCB can really type. It seems that applicants for posts at CSRG, UCB have to pass a typing test to get in; the test consists of typing the full title of the institution without using the delete key.

These really great typists contributed mightily to the development of UNIX. For example, they virtually invented the useful comment in the kernel code. It is suspected that the people who understood the comment You are not expected to understand this don't understand why comments are useful. However, UCB hackers tend to show off their skill somewhat and embarrass others by producing really big programs. Most of the rudeness about UCB system stems from the scorn poured onto these big programs, people seem to forget that other UNIX influences have generated really big programs. For instance, you can't get the original f77 V7 compiler into a PDP-11 unless it has separate I/D space.

AT&T headed towards re-organisation and the oddly named UNIX System III made its appearance. It was odd, too, because it is based on the PWB UNIX system which not many people had used. It didn't feel like UNIX because lots of things had moved and had their names changed. With System V, AT&T tell us that this is **THE** UNIX operating system and hopes it will rule the world. On initial inspection, System V is well typed and has long manuals. Error messages are now wordy and it is almost possible to say that you don't need the source to find out why something happens the way it does. In truth, the current situation is that the UNIX systems which the outside world can obtain are in the hands of people who employ others to type - or so it seems. Typists have moved into the UNIX world, and those of us who are still bad typists complain that UNIX V6 was the last real UNIX system.

Meanwhile, back in Bell Labs, those guys have conquered the typing problem by designing high quality terminals where all you need to do is point. But that's another story.....

Summary of C-Standards Workshop at USENIX

Henry Spencer
University of Toronto, Zoology
{allegra,ihnp4,linus,decvax}!utzoo!henry

July 7, 1984

The following is an informal report on what was said at the C Standards workshop at Usenix. The workshop essentially consisted of a presentation by Larry Rosler (of the ANSI C effort) plus question-and-answer afterwards. I apologize to Larry for any errors in the following. (Incidentally, he deserves a vote of thanks from everyone who attended the session. He flew in from the East Coast, at considerable inconvenience, basically just to give that talk.)

The ANSI C standards effort is X3J11. It's split into three subcommittees: environment, library, and language. Rosler is chairman of the language subcommittee.

The environment subcommittee is wrestling with a whole mess of very fuzzy things about how C relates to its surroundings. Alone of the three subcommittees, this one has no existing document to work from, so they're sort of feeling their way. Among the things they're trying to cope with are how a C program gets run (tentatively "main(argc, argv)", but the question of environment variables is very difficult on non-Unix systems) and how to resolve problems with European character sets.

The library subcommittee is working from chapters 2 and 3 of the Unix manual. Most of chapter 2 is gone because it's Unix-dependent, although a few things like "signal" are still there. Most of chapter 3 is still present: stdio, chars and strings, memory allocation, basic math functions (nobody feels like standardizing the Bessel functions!). They are looking at things like error handling in the math library.

The language subcommittee is the one all the detail following is about.

Their basic goals are:

- portability
- preservation of the "spirit of C", i.e. the ability to get right down into the bits if you want
- minimizing the impact on existing valid programs
- formalizing proven enhancements (emphasis on "proven")
- producing precise but readable documents

The specific approach to that last item is to tidy up and tighten up the existing C Reference Manual. The idea of defining C by use of a mathematical formal definition was discussed, but it was rejected on the grounds that the audience for a definition written in English is several orders of magnitude larger.

They've started from the System V.2 C Reference Manual. There have been three major areas of change in that since the "white book":

1. Long identifiers. The problem with Berklix-style arbitrary-length

names is that they break existing tools and file formats. The breakage is much less severe if one simply cranks up the limit instead of making it infinite. Internal names (including preprocessor names) are now significant to 31 characters. External names are, alas, significant only to 6 characters and case is not significant in them; this cannot be improved without making the standard incompatible with most non-Unix object-module formats.

2. Void and enum. "void" is the type returned by a function that doesn't return a value. You can also cast things to "void" to throw away an unwanted value. The keyword is also used in a couple of other places, discussed later, to avoid having to introduce too many new keywords (any of which has the potential to break existing programs). Enums are as in V7; improvements to permit things like ordering comparisons (>=, etc.) on enums are still being thought about.

3. Structure/union improvements. Structure assignment, passing, and returning are as in V7. Structure comparison isn't there, at least not so far. Member names are now local to the particular structure, instead of all being in a global name space; this means that you have to be more careful about getting the type of (e.g.) the left-hand-side of "->" correct, or the compiler will object.

The committee has introduced three major changes since the V.2 CRM:

A. Function-argument type declaration and checking. Instead of just saying "extern int fread();", you can now say:

```
extern int fread(char *, int, int, FILE *);
```

so the compiler can do proper type checks. In the event of a type mismatch, the same conversions as for the assignment operator apply. (Hooray, no more casting NULL pointers!) Variable-argument functions like printf can be declared like:

```
extern int printf(char *,);
```

It is admitted that the comma is not all that conspicuous, and that this syntax makes it impossible to declare a function which has *only* variable arguments. These things are, of necessity, compromises. [Please note that neither Larry Rosler nor I necessarily *like* all the things I'm reporting.] There is an ambiguity when it comes to declaring no-argument functions, since "extern int rand();" looks like an old-style declaration which doesn't say anything about the arguments. The convention for this is:

```
extern int rand(void);
```

which means "no parameters".

B. "const". A new keyword (sigh) which is used to mark things that are read-only, with run-time assignments forbidden. These things might be put in ROM or in text space. Some examples, with notes:

```
const float pi = 3.14159;
```

This is a real, live, named constant, which will show up in the symbol table (unlike #defines).

```
const short yacctable[1000] = { ... };
```

An obvious case.

```
const char *p;          /* pointer to constant */
char *const q;          /* constant pointer to something */
```

Illustrating two different uses: the first is a pointer that can be changed but can't be assigned through; the second is a pointer that can be assigned through but can't be changed. It is agreed that the syntax is less than ideal. Note that const is *not* a storage class, it is part of the type.

```
extern char *strcpy(char *, const char *);
```

Illustrating telling the compiler that strcpy doesn't change its second argument.

C. Single-precision arithmetic. If all operands in an expression are float, the compiler is allowed (not required!) to evaluate it in float rather than double arithmetic. The choice is explicitly implementation-dependent. Casts can be used to force evaluation in double. Numeric constants, e.g. "1.0", are double, *not* float! This last isn't ideal, but trying to fix it invariably makes life much more complex.

The original double-only rule was partly a concession to the pdp11, partly just plain simpler, but partly a way of avoiding multiple versions of all the library routines. With declarations of function argument types, the last problem is pretty much fixed. All the library functions in the standard want "full width" types, so that if you don't declare them, you're still safe.

Some lesser issues:

I. "Promiscuous" pointer assignments are illegal. You must use casts when mixing pointer types or mixing ints with pointers.

II. "void *" is a new kind of pointer, which cannot be dereferenced but can be assigned to any other type of pointer without a cast. The idea here is that "char *" is no longer required to be the "universal" pointer type which can point to anything. So for example, the declaration of fread earlier really should go:

```
extern int fread(void *, int, int, FILE *);
```

(People who have machines where all pointers have the same representation, don't complain. You are lucky. Others aren't.)

III. "volatile" (the choice of name is tentative) acts like "const" in the syntax, but with different semantics. It means that the

data in question is "magic" in some way (e.g. device registers) and that compilers should not optimize references to such things. This resolves a long-standing problem with writing optimizing compilers for C.

IV. "signal" is in the library. This means that reentrancy is explicitly part of C.

V. The preprocessor is part of the language. The committee has opted for a simple and clean definition, which does not perpetuate some implementation accidents of some of the existing ones. There are some minor improvements, like permitting space before the "#".

Some trivial additions:

i. Hexadecimal string escapes. [Retch.] "Here's an ESC \x1b ".

ii. String constant concatenation. Two string *constants* occurring adjacent to each other in the source are considered concatenated. Note that this is constants only. Among other minor things, this makes string continuation across line boundaries less ugly.

iii. "unsigned char", "unsigned short", "unsigned long" are all part of the language. Plain "char" is *not* required to be signed or unsigned (requiring either would make efficient implementations impossible on some machines). The question of a "char-sized int" type, of whatever syntax, has not yet been resolved.

iv. The unary + operator. Same conversions and type restrictions as unary -. Does nothing. This is partly consistency with other languages, and partly consistency with things like "atof". (At the moment, "+3.14" is valid when atoffed from a string but not when compiled into a program!)

v. Initialization of unions and automatic aggregates. The latter is just removal of an existing restriction. The former is tricky; there is *no* clean way to define it. The committee has opted to do something not necessarily good, but simple: the type of the initializer is that of the lexically-first member.

vi. The selection expression of a "switch" can be of any integer type. (E.g. it can be a "long".)

vii. #elif. An added bit of preprocessor syntax, to simplify using #if's like a "switch".

Some things are gone:

01. "entry", "asm", and "fortran" keywords. (Although the last two will probably be mentioned in a "recognized extensions" appendix.)

02. "long float" is no longer a synonym for "double". Nobody ever used it. There was discussion of using "long float" and "long double" to cope with machines having more than two floating-point types, but conversions and such are an unknown swamp in such a case, and the committee decided not to try.

03. 8 and 9 are not octal digits.

04. Pointer-integer conversions now are strictly type-checked, as I mentioned earlier.

05. The following code fragment is illegal:

```
foo(parm)
int parm;
{
        int parm;
        ...
```

Some compilers interpret such a situation as nested scopes, so the inner declaration hides the outer one. In this particular case, this seems both useless and dangerous. The scope of the arguments of a function is now identical to that of the local declarations, so this is a duplicate declaration and illegal.

06. Nothing is said about the alignment of bitfields, not even the K&R guarantee that they don't straddle word boundaries.

07. Some existing compilers permit taking the address of a variable declared "register" if the variable is not in fact placed in a register. This is now outlawed; "register" and the unary "&" operator don't mix.

All in all, the current draft standard doesn't sound too bad to me. I will be getting a copy of it shortly, and may have some more comments at that time. A number of things are still unsettled. The committee's (very tentative) notion of schedule is a final draft for public comment by the end of the year, and a real standard by the end of next year. [Sound of crossing of fingers.]

Comments on this should *not* be addressed to me; I'm just an interested observer, not a participant. Write to:

Lawrence Rosler
Supervisor, Language Systems Engineering Group
AT&T Bell Laboratories
Summit, NJ USA

No, I don't have a network address for him.

;login:

The USENIX Association Newsletter

Volume 9 Number 2 **April 1984**

CONTENTS

The deadline for submissions for the July issue of *;login:* is June 30

EUUG
European UNIX† Systems User Group

Newsletter Vol 3 No 3
Autumn 1983

† UNIX is a Trademark of Bell Laboratories.

EUUG Meeting
Trinity College, Dublin, 7/9 September 1983

Peter Collinson
Secretary

Introduction

These reports are becoming somewhat redundant because most speakers have submitted a paper. The idea for future conferences is to have a formal call for papers which will be sent to referees and published as proceedings. This report is derived from my somewhat sketchy notes and will serve to give a guide to what went on.

Even on close inspection, I could find no reference in my notes to the word 'Guinness'. This is a pity because the best Guinness in the world is to be had in Ireland, the stuff called Guinness elsewhere just isn't the same.

Day 1 - 7th September

The first session of the conference was devoted to vendor presentations. As usual, these notes may be incorrect and aim just to give the flavour of the talks. For further (and correct) information, contact the vendor.

Item 1: 10.35am **Eddie Bleasdale, Bleasdale Computer Systems Ltd**

Bleasdale Systems

Eddie started with a brief history of his company and how they relate to UNIX systems. The hardware which the company is offering is based on the M68000 processor, which can be either 8Mhz or 10Mhz. There will be an upgrade to the M68010. Memory sizes are from from 768Kbytes up to 3.5Mbytes, the backplane is Multibus. Peripherals include: a high performance Winchester disc as a main disc subsystem; a floppy disc; ½" magnetic tape and ¼" streamer tape. It is also possible to have customised hardware options.

The UNIX system is the Unisoft version, which is a UNIX V7 system plus some Berkeley enhancements. The company also does some software work; it has enhanced the disc driver for the hardware and provides some diagnostic software in a monitor ROM. They have support for an array processor, a floating point processor and specialist I/O interfaces.

Bleasdale now offer the following languages: C, F77 (Silicon Valley Software, this can use the floating point processor), Pascal (two flavours: UCSD and Silicon Valley Software), BASIC+, RM/COBOL, Whitesmith's cross compilers, cross assemblers from Real Time Systems, APL, POP-LOG and LISP.

The company has collected an impressive list of applications packages, some of which are: the **xed** word processor, a spread sheet system from View Comp, the **Mistress** relational data base system, a payroll system called CPAY and a ledger system from UNIGEM. There were others but my hand would not move fast enough.

Item 2: 10.55am **John Olford, NCR Ltd**

The NCR Tower

Not a building but a computer, the TOWER 1632 is a M68000 based UNIX system. Its name is derived from the tall box in which it is packaged. The box is designed to sit quietly by a desk in an office, it does not have any noisy fans which would annoy people. The main internal bus is Multibus, but it does have its own memory bus which supports up to 2Mb. Discs are either 5½" or 8" Winchester, the main cabinet can support up to 60Mbyte and an expansion cabinet can support another 160Mb. Other devices include: floppy disc, a streamer tape drive and variety of terminal and printers.

The UNIX system is the Unisoft port and supports C, RM/COBOL, SMC/BASIC, PASCAL (an NCR compiler to the ISO standard) and F77.

Price: entry level £ 11,000; typical configuration £ 18,000 and a full configuration will be £ 50,000.

Item 3: 11.10am **John Roseman, Urban Software**

Urban Software

Urban Software are a software development house based in New York, they have one main product called **Leverage**. This is a screen based data capture system intended to generate ASCII files suitable for post-processing by all those UNIX tools which you know and love (or hate). They do not sell to end-users but prefer to sell through a third party - Sphinx Ltd in the UK.

The main theme of John's talk was precisely that there exist a lot of tools on the UNIX system and so why don't commercial people people use them!

Urban also produce a quarterly newsletter, I have two editions which contain extremely useful information on who produces what UNIX system on what machine and for how much.

John then went on to talk about the recent UNIX history, this has been documented in these pages before, so I will omit it.

Q. How does Urban Software make money?

A. Most revenues come from the CP/M version of the **Leverage** package. This is sold through a third party - Perfect Software.

Item 4: 11.25am **Frank Graham, Software Ireland**

SIBOL

Software Ireland are a subsidiary of the National Westminster Bank and have offices in Belfast, Dublin and California, employing 30 people. Their main areas of interest are software products for UNIX, RSTS and CP/M; training and various consultancy.

Their main UNIX product is SIBOL which is a commercial programming language which is compatible with DEC's DIBOL. DIBOL is DEC's main supported commercial language and has a lot of applications programs written in it.

SIBOL comprises a compiler, linker, editor, debugger and a library of external routines. It is written in C and so runs on all UNIX and UNIX look-alike systems.

Software Ireland have ported a number of commercial applications packages which they now sell under the title of PHENIX.

Item 5: 11.34am **Brian Rees, DEC**

DEC UNIX Systems in Europe

Well, after all this time..... a representative of DEC at a EUUG conference talking about DEC selling and supporting UNIX!!!

DEC now fully supports a PDP11 UNIX system called V7M-11 which has been announced and can be ordered. A UNIX system will be available to run on the PC350 some time towards the end of the year.

The 32-bit VAX native system (which was called ULTRIX on the slides) will be 4.2BSD running on the range of VAX processors from the VAX11/730 to the VAX11/782. The system will have support for the new distributed file architecture (HSC50).

DEC are also committed to support VNX which will run a UNIX system under VMS, they are also busily engaged in installing programs into VMS which show the 'cultural aspects of the UNIX system' - for example, the shell. They have some other programs which are UNIX utility look-alikes, for instance they have a package with the same functionality as SCCS.

All VAX processors running VMS, VNX or native UNIX systems will be supported by DEC. Although it is unclear whether UNIX licensees who are licensed by other means than DEC will be able to get hold of the appropriate device drivers in order to support DEC's field service engineers.

There will be a full array of software support service options for VNX. The educational services will concentrate on VAX/VMS.

Item 6: 11.50am **Matthew Wallace, Hewlett/Packard**

The HP9000

The HP9000 system is a 32-bit processor which was launched in September. It runs a UNIX system and HP are committed to put UNIX systems onto a wide range of their processors. Currently, the HP9000 is targeted towards the fields of Computer Aided engineering (e.g. Graphics - CAD/CAM) and areas which involve software productivity.

The UNIX system, called HP-UX is System III plus the usual Berkeley enhancements. HP have their own graphics package and their own Data Base Management system. They intend to support an Ethernet working to the IEEE 802.3 standard.

HP are launching a M68000 system running UNIX in the next two months.

Item 7: 12.03pm **Mike Salmon, Taurus Technology**

Taurus Technology

Taurus are a case of *mv 'Structured Methods' Taurus*. They are engaged in training of various sorts and can generate 5 day workshops in UNIX and C. The majority of their business is on-site training for customers using the local machines. Recent customers include British Telecom and Rediffusion Computers.

Internally, they use a 32-bit Gould machine and an ONYX.

☞ Lunch (but no Guinness!) ☜

In the afternoon, there were two parallel sessions. Fortunately, I could only manage to go to one. The session which I attended appears first.

Item 8: 2.00pm **Emrys Jones, Cgram Software**

EUUG Business

The EUUG business is very hard to take notes about, largely because it tends to be vague acrimonious discussion.

The main item informed the membership of the proposed change in the structure of the organisation. Currently, the composition of the EUUG board does not really reflect the actual structure of the group. Apart from the UK, all other countries have a national group. It is felt that it would be better to (a) set up a national UK group and (b) change the EUUG board to comprise the chairmen (or nominated members) of the national boards. Since this full EUUG board would be too big and unwieldy to actually take decisions, a smaller unit called the *EUUG Executive board* should be formed from the EUUG board, this small group would be responsible for day-to-day running of the EUUG.

David Tilbrook (Imperial Software Technology/Imperial College) undertook to establish a group to get the UK National Group off the ground. David said that the main aim of the UK group should be to get all UK sites connected via EUNET. There will be a preliminary discussion about this on September 23rd, 1983.

The second item of business informed the membership of the formation of a UK Chapter of /usr/group. /usr/group is an organisation in the USA which consists of commercial vendors and users. The organisation is a prime mover in the UNIX standards debate and has produced a draft standard for system calls. There was some (extremely acrimonious) discussion about why the UK /usr/group needed to be formed in the first place. In general, my personal opinion (note: personal) is that the aims of EUUG are to *promote* UNIX and the aims of /usr/group are to *vend* UNIX. These aims are not mutually exclusive, but the groups may need to to go about things in different ways.

The next spring EUUG conference will be in Nijmegen, Holland from 16th→18th of April, 1984.

Item 9: 2.34pm **Jim Kennedy, Bill Murphy, AT&T**

Licensing and other issues

There was not much new stuff from AT&T but I'll include most of the information I have because licensing is always one of the hot issues when talking about UNIX. Some of this is derived from the Bonn Proceedings.

System V licensing is just taking a long time, there are lots of applications in the pipeline. Academic applications have been shelved until all the commercial licences are issued. The table below gives the licensing costs:

System V licence & Fees ($)			
	Initial CPU	Additional CPU	Upgrade from SIII
Commercial	43000	16000	1000
Educational/Administrative	16000	400	0
Educational	800	400	0

For educational licences, AT&T will charge $800 to license as many machines as you want as long as all machines are licensed in one application. For this they will supply a standard distribution. Any additional requests will then cost $400.

In response to a question about royalty payments for selling time on a machine, Bill Murphy said that there would be no further timesharing agreements and there would be no cost for selling time.

Bill then went on to talk about the software support which AT&T are selling. The base for this 'phone-in' system is Lisle, Illinois. Level 1 support will cost $150 ($50 for an additional CPU) per month and will entitle the purchaser to a newsletter, a method of problem reporting and periodic updates. Level 2 will cost $350 ($100 additional CPU) and will supply Level 1 support plus a hotline reporting system (2pm→11pm GMT) with a 2 hour response to problems. Requests outside these hours will cost $100 per hour.

Documentation for System V can be bought separately. The two volumes of product release documents cost $30, three volumes of UNIX Operating system manuals cost $60 and the 7 volumes of Operating System guides cost $145.

AT&T now sell UNIX training, this is based in the USA but they are willing to send an instructor and a course to Europe. There are five courses:

1) Operating System implementor's curriculum
2) Application developer's curriculum
3) Manager/supervisor's curriculum
4) System administrator's curriculum
5) User's curriculum

Bill also announced three new product releases: the S statistical package, the UNIX Writers Workbench and The UNIX Instructional Workbench.

Telephone numbers below are from the USA - I'll leave it to you to decode them for your country.

UNIX System Contacts for AT&T International

	City	Name	Telephone	Telex
USA	Basking Ridge,NJ	W.J.Murphy	201-953-7581	219345 ATTIUR
UK	London	R.R.Hall	441-930-0691	24660 ATTIUK
Europe	London	A.Ursano	441-930-0691	24660 ATTIUK
Ireland	Dublin	M.Felton	353-1-515-411	25815 TCRNEI
Greece	Athens	S.P.Tsitouris	301-3232-797	219245 AP5GR
Egypt	Cairo	I.N.Shalaby	160-200-726-998	92444 ATTIUN

☞ Tea (but still no Guinness!) ☜

Item 10: 4.09pm **Rich Graveman, Bell Laboratories**

UNIX in the future

Rich's talk gave an overview of the way Bell Labs are thinking about the development of UNIX.

Bell now see UNIX systems as a set of programs and utilities which can be unbundled. Rich showed the following information which shows how the unbundling may take place. A UNIX system comprises:

- The Operating System
  consists of:
  - ☐ The Kernel: which provides a core interface to the hardware; implements system calls; supports the file system structure; and copes with resource sharing and allocation
  - ☐ Device drivers

- ☐ A set of basic utilities: which comprise the Shell; system administation tools to set the system up, maintain the file system and to configure the system for the hardware; directory and file management utilities; and an editor.

- Optional utilities
  - ☐ Functional Command groups: the utilities are split into groups of commands which are the requisite tools for certain well defined tasks. For instance, a set of utilities which allow interuser communication within one machine (**mail, mesg, news, wall, who, write**) may not be required on a single user machine.
  - ☐ Optional device drivers
  - ☐ Program support libraries
- Workbenches
  - ☐ Programmer's workbench: provides a development environment for programming and debugging.
  - ☐ Document preparation workbench: has an input mechanism, a formatter, **tbl**, **eqn** and some form of production capability.
  - ☐ Writers workbench: proofreading and style analysis programs.
  - ☐ Instructional workbench: computer assisted instruction currently supporting 4 courses (this is not **learn**!).
  - ☐ Analyst's workbench: contains the S package, data entry and analysis tools.

Rich then talked about the rationale behind the AT&T agreements with the four micro-processor companies. The micro-processor area is the fastest growing segment of the computer industry and it seemed sensible to get an up-to-date UNIX system available on quality chips as soon as possible. There is also a need to standardise ports of UNIX systems in order to encourage the development of applications. The agreements which have been worked out enable the μ-processor companies to port UNIX System V with Bell guidance. At the end of the day, WeCo will own the ported system, and it will be possible to get source licences from WeCo. Binary licences will be available from the μ-manufacturers.

The direction for the future development of UNIX systems hinges on the main goal which is 'to make the UNIX operating system a common basis for applications on a wide range of machines from micro-computers to mainframes.' Current activities to fulfill this goal include: (a) the provision of a single supported UNIX system for both the Bell system and commercial customers; (b) Bell is working with users to achieve a standard; (c) Bell will continue to release new products and (d) intends to unbundle UNIX into functional pieces.

In the unscheduled future, Bell is looking at various areas:

- User friendliness, items of interest here include: a visual shell, some methods of forms input, a help facility, operation aids for running UNIX in a large machine environment and finally, some graphics routines and systems.
- Standards: items here range from a standard for error messages from programs (which is fairly easy to do) to a standard for command syntax (which is very hard).
- Application programs and packages
- Security

Continuing concerns are: reliability, performance and documentation.

There were several questions:

Q. In the future, the main limitation will be the memory requirement of programs. Are there any plans to look into memory utilisation or is memory so cheap that it doesn't matter?

A. It probably doesn't matter - but the splitting of UNIX into several options will help with this.

Q. Will UNIX ever support dynamically loadable device drivers?

A. Not in the short term.

Q. Will the 'Newcastle connection' be supported under Bell UNIX?

A. (Answer from Rob Pike, Bell Labs research) In the research section of Bell labs there are 16 VAX systems with a connected file system (the UNIX system is called Edition 8). The system does not support the moving of processes from one machine to another which Newcastle does. There are NO plans to release Edition 8.

Q. What's in release II of System V?

A. Bug fixes, some enhancements to performance.

Q. Is there a firm date for Bell Labs UNIX systems supporting virtual memory?

A. Not this year.

Q. Will there be an enhancement of the performance of the file system in order to support virtual memory.

A. There will be no change to the file system.

☞ UNIX Training session ☜

This went on in parallel with the talks above. This section of these voluminous notes has been supplied to me by Cornelia Boldyreff.

The official programme included five balanced presentations of half an hour duration. But these began to run over as the afternoon crept on. So that the last presentation was completed just before 5.30pm, when we were due to be thrown out of the room. Despite the rush through, the response of the audience was enthusiastic and positive. In future issues of the newsletter, notes by the various presenters will be published, if they all keep their rash promises to send copies to the editor.

Item 11: 2.00pm **Ben Salama, Sphinx**

UNIX information and Training Survey

Ben reviewed UNIX publications - books, magazines and market surveys. His personal preference for the book: *UNIX - The Book* was applauded by one of the authors in the audience. Ben also reviewed training options - live and recorded.

Item 12: 2.30pm **Alex Osadzinski, Quantime Systems**

UNIX fundamentals for non-programmers

Alex reviewed the history of UNIX and its development as a commercial product. He then let the Bell Labs UNIX stars perform on video: Ken Thompson, Dennis Ritchie, Brian Kernighan *et al.* Great stuff.

Item 13: 3.00pm **Terry Crook, C-Gram Software**

Shell programming

All those features in the Bourne shell which mystify tyros were explicated by Terry. There is a lot to learn about the shell, this session started late and ran on. Some people managed to get tea before the next session.

Item 14: 4.00pm **Andy Rutter, Taurus Technology**

UNIX fundamentals for programmers

This talk was subtitled *UNIX - The Presentation.* In 22 slides, Andy covered: how commands work, system calls, the process environment, file access, inter-process communication, and the standard I/O package. He also managed to answer questions, too, in just over half an hour.

Item 15: 4.45pm **Peter Griffiths, Taurus Technology**

C Language programming

You've heard about the books, seen the movies, been shell shocked, totally immersed in UNIX; so, to complete the session, all there is to C. This was obviously a moving experience for some Pascal programmers in the audience.

☞ End of Day 1 (and onto the Guinness!) ☜

Day 2 - 8th September

It is hoped that most of the people who gave talks will have either submitted a paper or will do so before this wonderous report reaches you.

Item 16: 9.00am **J.Marino, Universitau di Genova**

Experiences with BERKNET

The University of Genoa wanted to connect a VAX11/750 and a number of small PDP11 systems and rather than connecting with **uucp**, it seemed sensible to use some form Local area network (LAN). The main problem with networking is to convince people that they need it, no-one will pay any money towards setting up a LAN until they have seen the benefits. The strategy adopted was to demonstrate the need for a network by installing a low cost one with the expectation that when users need more speed they will be willing to pay for it.

The UNIX systems were all running some version of Berkeley software (4.1BSD on the VAX and 2.8BSD on the PDP11s), and these releases include the software for a LAN system called **berknet**. It seemed sensible to pick up this software and use it.

The **berknet** software is able to drive several types of connection which trade speed of operation with the amount of intelligence which is required in the kernel. The simplest type of protocol uses a standard teletype line in COOKED mode and needs no kernel modification, the next step up involves the installation of the **bk** packet driver as a line discipline in the kernel but still uses a standard teletype line. More expensive links use the **bk** packet discipline via a high speed link like a DMC11.

Problems with the implementation were mostly due to the PDP11s. First, the deamon which runs the network did not fit into the PDP11, this was cured by moving a number of strings into a separate file and also by removing redundant code by eliminating unused routines. This was done in a cunning way by putting all the routines into an archive library and recompiling. The second problem is that the **bk** line driver did not work on a PDP11, this was fixed by recoding and making the kernel *copyout* routine able to return an odd number of bytes.

The result was a network whose raw transfer rate is greater than 100 bytes/sec (on a 1200 baud connection - faster than that and the PDP11s fall over). The network has a good error rate and has proved a good investment in time.

In the future, it is expected to extend the LAN to other machines and then to use an Ethernet connection to improve the speed on the connection.

Item 17: 9.33am **Peter Jones, University of Essex**

Touch Input speeds up interaction with UNIX tools

With a title straight out of *Electronics* magazine, Peter talked about the work he has been doing with a touch panel attached to the screen of a normal character VDU. His touch panel works by infra-red, although there are many technologies (capacitance, conducting surface, sound, pressure and light beams). The main disadvantages of a touch panel as compared with other pointing devices is that (a) input is physical and tiring, (b) the hand tends to obscure the display, (c) the input has a low resolution (which is no problem on a normal character display), and (d) there is some problem with parallax because the screen glass is thick. Finally, hands are sticky things and the screen tends to get dirty.

However, the pointing is a very natural action and using a touch panel does not involve picking anything else up. Peter seemed really keen on his system.

He has written a screen editor where positioning is done by pointing and other selective inputs are done by having user definable touch buttons on the screen.

☞ Coffee (black, no Guinness) ☜

Item 18: 10.37am **Sam Leffler, Lucasfilm**

4.2BSD, Reflections on a large software project

Sam subtitled his talk '4.2BSD. The morning after'. The main aim was to give some assessment of the work involved in the 4.2BSD development.

Sam started with a chronology of UNIX VAX development:

December 1978	UNIX 32V released
December 1979	3BSD released
	demand paging - Bill Joy
January 1980	DARPA meeting
	the great VMS vs UNIX controversy
	VMUNIX adopted by ARPA
July 1980	Computer Science Research Group (UCB) funded
October 1980	4BSD (later 4.0BSD)
	performance enhancement
	Pascal, long identifiers
	void in C, job control, stdout buffered
June/July 1981	Draft 4.2 system manuals circulated
July 1981	4.1BSD released
	VAX11/750, auto configure,
	VM performance enhancements as a result
	of the Kashtan benchmarks
August 1983	4.2BSD released

He pointed out that there had been a long gap in time between the 4.1BSD release and the issue of 4.2BSD. This had been partially filled by the release to certain sites of intermediate systems (4.1a→4.1c). The 4.1BSD system has several advantages: (a) it is well tuned; (b) it is stable and very robust; (c) except for the virtual memory management and job control, the system is V7 at the system call level; and (d) the MTBF of the system is very high, typically as much as 2→3 months. Various figures are quoted on how many 4.1BSD systems are running. Sam said that he estimates that 15% of all VAXs sold by DEC are running 4.1BSD, estimates vary from 10%→25%.

The goals for 4.2BSD were: to provide the DARPA research community with a standard operating system; to provide a distributed environment at UCB, to maintain the UNIX philosophy†; and to maintain 4.1BSD robustness.

To achieve these aims 4.2BSD has interprocess communication (IPC) primitives, fast network support, the ability to have very large virtual machines, and a fast file system. The file system performance enhancement is required to support large virtual memory for processes.

In 4.1BSD and all previous UNIX systems, IPC is achieved by pipes and multiplexed files. Pipes are not really good enough because they have to share a parent process and multiplexed files are limited to a single machine by their use of the UNIX file system name space. When designing the new mechanism it was felt necessary to have: (a) straightforward semantics, (b) the ability to make an efficient implementation, and (c) the new system must allow higher level objects to be constructed.

In the area of VM enhancements, it was felt that 16Megabyte processes supported by 4.1BSD were too small, and also there were no mechanisms for taking advantage of the large address space.

† Jeers from some members of the audience

The aims for 4.2BSD were to provide a 2 Gigabyte address space and to provide facilities for using this address space by allowing shared files and mapping files into memory. To do this it is necessary to: page the page tables, have a better swap management and to increase the file system performance so that mapped files work efficiently.

4.1BSD had increased the basic file system block size from 512 bytes to 1024 bytes and this had significantly increased the performance of the file system. But there still are problems: the file system fragments very quickly, the disc layout is not very good because it is a linear array of blocks and the I/O transfer size is too small. Solutions to the problems were to (a) implement a larger block size and allow block fragments to minimise disc wastage; (b) to implement a better disc layout policy; (c) to parameterise to allow tuning and (d) to replicate essential information (super blocks) to permit better robustness.

Sam then talked about the 'Good stuff' in 4.2BSD. In the IPC, the user interface is sound and it generates a clear set of primitives. In the implementation, it has turned out that datagrams and 'out of band data'† are useful. It is also possible to make a clean extension from on-machine work to the network environment. The networking interfaces work well and the internal model has proved to be successful (with some caveats). There is a lot of latency and the hardware support works well.

The file system has met the bandwidth requirement and performance is now limited by the hardware (it proves that UNIBUS discs are bad news). 4.2BSD have **mkdir, rmdir** and **rename** implemented as system calls and this is a big performance win. The idea of symbolic links (i.e. a file containing a name of another file) has been very heavily used. The impact of the changes to the directory format (255 character names) has been small.

Having praised the good stuff, Sam went on to point out the 'Bad stuff' in the implementation. In the IPC, debugging facilities are non-existent; he felt that the addressing *vs* naming issue had not been tackled properly; there is a problem with connection shutdown and the documentation is totally insufficient.

The file system is a lot more complex and will take some time for system administators to learn how to use it effectively. A bigger problem is that interaction with the new storage architectures (DSA) is not good.

The signal mechanism is 4.2BSD is totally incompatible with the old system.

Finally, Sam said that it is possible that 4.2BSD may probably be the last UNIX software release from UCB. The people there are getting interested in different areas.

Item 19: 11.30am **Teus Hagen, Maths Centre, Amsterdam**

Pros & Cons for different LAN techniques

This was a free-for-all with Teus saying procative things and attempting to get a discussion going, it was partially successful. I was unable to take notes, which means that I must have been paying attention to what people were saying rather than reading the overhead projector slides.

☞ Lunch (Even if you wanted Guinness by now, there was none) ☜

† *expedited* in 'Yellow book' terms.

Item 20: 1.45pm **Film by Lucasfilm**

Computer Graphics and the Return of the Jedi

This film was a 'short' showing how some scenes for the film had been developed. It proved a bit of an anti-climax because we needed to see the 35mm final scene. Still it was interesting to see the nice hardware/software.

Item 21: 2.01pm **Peter Collinson, University of Kent**

UNIX at UKC

See the paper elsewhere.

☞ Tea (after the last speaker everyone needed lots of Guinness) ☜

Item 22: 3.35pm **Marja-Riitta Kiovunen,Timo Kunnas, Helsinki University**

HUT APL, an APL System for UNIX

Marja-Riitta gave a brief description of the features of APL and Timo talked about their implementation. I quote from their abstract:

HUT APL, an APL system compatible with IBM VS APL, has been developed in Helsinki University of Technology. The system runs under UNIX 4.1BSD running on a VAX computer. It consists of an APL language interpreter, workspace routines and associated commands, facilities to use auxilary processors with shared variables and an editor modified for APL use.

So far, APL systems commonly available for UNIX have been incomplete implementations, lacking some features and often giving wrong results, especially in special cases such as empty arrays. VS APL is the *de facto* standard used by most manufacturers, it also conforms with the ISO and ANSI proposed standards which are being developed. The HUT APL system is downward compatiblewith VS APL, with minor exceptions which are mostly sue to the underlying architecture. It has features typical of many UNIX sub-systems, such as the possibility of using UNIX commands, editors and files; and the ability to start asynchronous or synchronous processes which can communicate with the APL system.

The system is based on the APL available from USENIX, this was originally developed in Yale University and later at Purdue and Berkeley. These implementations are somewhat inefficient. For instance, all numbers are represented internally in 64 bit floating point form, which is neither efficient nor preserves space. Also, the APL syntax analyser is written in yacc, and APL syntax is too dynamic for this to work properly. However, it was more important for the developers at Helsinki to generate a system which worked correctly rather than one which worked in a super-efficient way.

The HUT APL system is completely written in C, and can thus be ported to other architectures. It has a clean environment into which any new features of the APL language can be implemented easily. The system is in the public domain and is available on request.

Item 23: 4.05pm **Michael Tilson, Human Computing Resources Corp**

A Tutorial on C language portability

This main aim of this talk was to demonstrate that C is an unsafe language but with a bit of care it is perfectly possible to port C programs between different architectures without a lot of difficulty. Portability is an important issue because software is expensive to write and if the program is 'portable' then it costs less to move it. If software is portable, then the software is not locked into one hardware vendor. Mike also said that portable software is more reliable, since it tends to be

less 'dirty', and use fewer 'tricks'. Errors in type matching often indicate errors in thinking.

Mike was concerned about source code portability and not any related issues such as binary portability, portable library routines, the operating system interface and the command interface.

The talk was illustrated by small examples, all of which have happened in practice and all of which passed happily through **lint**. I won't attempt to reproduce the examples here, but the main points which the examples made are:-

- Type casts and **lint** do not solve all problems.
- There is no guarantee that the bit representation of a **char** pointer is the same as that of an **int**.
- It is not safe to assume that the value 0 is the same as a pointer containing zero. This is especially hard when passing a zero value into a routine which expects a character pointer as an argument. The expression NULL should always be used rather than the constant 0.
- Whenever an **int** variable contains a number which is greater than an **int** on a PDP11, the variable should be defined as **long**.

Mike says that non-portable C cost HCR about $100,000 in the last year.

Item 24: 4.36pm **Mike Banahan, Taurus Technology**

Benchmarking - report and discussion

Mike had followed up some discussion about EUUG generating a benchmark package which had occurred at Bonn. He announced the release of the *EUUG Benchmarks* on the 16th September. The benchmarks are a complete package consisting of a shell script, a re-written **time**(1), a makefile and a set of benchmarking programs written in C.

The aim of the package is to provide a simple standard set of programs which are easy to run and will not generate too many figures for comparision. The programs are deemed to be the core of the benchmarking package and any further programs will be added to the package. Existing programs in the package will not be removed.

The package currently consists of the following operations:

1) Check that the new **time** works, this uses a human with a watch to ensure that the clock on the machine is running at the correct speed.
2) Size the machine and run some tests to determine the CPU speed. This gives a fiddle factor which is then used to scale the tests so they don't take a very long or very short time.
3) CPU tests: include integer arithmetic, register arithmetic, arithmetic and function calls and some *real* maths.
4) I/O tests: include pipe throughput, pipe throughput using 1 byte writes, some writes to /dev/null and disc writes and reads.
5) Compound load test: run a C compilation sequence once, and then run many in parallel.

The report from the package gives the normalised real/user/system times for each test, plus the actual times for the compilation of each of the programs.

Mike ended with a dire warning; people expect too much from benchmarks. The benchmarks will not predict machine/system performance, nor are they intended to. The main aim is to allow a very broad comparision between machines.

☞ End of Day 2 (and onto the Mansion House for drinks) ☜

Day 3 - 9th September

Well, finally the Guinness and general lack of sleep caught up with me. I must confess and apologise for missing the first speaker. I reproduce his abstract.

Item 25: 9.00??am **H.Ludwigs, Quantum**

Portable Forms Management System

Forms management systems are usually fixed onto one type of host computer and one type of screen device. Therefore, it is impossible to move a program which relies on screen form I/O from one machine to another without heavy changes in the source code. PFMS is a tool which provides a screen interface which will run under UNIX as well as VMS, RSX11-M or RT11.

PFMS is modelled after a widely used system, FMS†, and implemented in Fortran 77.

DEC's FMS is a program to build, maintain and use screen forms on a VT100 or similar terminal. The FMS interface may be used from programs written in different programming languages, the user interface to FMS is language independent.

Interaction with the user program is via subroutine calls which specify the name of a form to be used, the name of a field within the form to be manipulated, and a string where the input or output value of this field is deposited. The actions and the placement of the cursor are specified by so-called field terminators in the FMS interface calls.

Item 26: 9.30am **Rob Pike, Bell Laboratories**

Unix style or *cat -v* considered harmful

I quote from Rob's abstract:

'UNIX is spreading rapidly throughout the commercial computing world. This spread is due largely to UNIX's portability and practicality. Its original popularity is the academic computer science community, however, was due to its embodiment of a number of new and simple ideas, cleanly implemented.

The passage of time and programmers over every line of UNIX source code has made the file system more complicated to learn, to use, and to maintain, and much bigger - the current VAX kernels are about a factor of 10 larger than the 5th edition kernel, but certainly not a factor of 10 better. Most of that growth has not improved the system, but merely added to it.'

Rob's thesis of over complicated growth of the system started by discussing the **cat** command. **Cat** started as a simple command which copied its input to its output, it is a general file copying utility which can be use to concatenate files. However, its major use is probably to display a file on the terminal. It is a simple *command* and then along came Berkeley and made it into a *sub-system* by adding switches to it. The switches which were added were mostly related to **cat**'s use as a general program for screen output. Rob maintains that the switches are not required, since their action can be achieved in other more general ways. The switches are

-s strips multiple newlines, this can be done by **sed**.

-n numbers the lines, this should be done by **pr** or a special utility.

-b which numbers the lines but suppresses blank line numbering is just yucky.

-v prints non-visible characters in a visible way, this should be done by a special command.

Rob's contention was that the original **cat** did not alter it's input in any way and the -n, -b and -v switches simply contavened this basic rule.

Rob then switched his attention to the problem of mis-placed functionality. Most UNIX

† FMS is a trademark of DEC.

systems do not have any mechanism to handle screen scrolling. Instead of tackling this problem by perhaps inserting a terminal driver which did it† or buying a BLIT which has scrolling in the terminal. Berkeley tackled the problem by distributing it thoughout lots of utilities. The **ucbls** which normally pretends to be **ls** on most UCB systems is a case in point. It generates its output in columns, this can be done either by *echo* * or by using **pr**. However, the program now has a problem, in that when you type something like *ls|wc*, you want it still to generate the correct answer. So, **ls** now looks at its output stream and says 'If this is a terminal, I will output in multiple columns; else I will output in a single column as before.'

A second case of this problem is illustrated by the **more** command. This command started out as a program to perform screen scrolling. The problem was that it found its way into other UCB commands, notably the **man** command. There was a problem now when the **man** command was run to a file. So, now **more** looks at its output to find out whether it is talking to a screen or not.

Rob's contention was that a problem which should have been solved simply either by getting a terminal which scrolled or by changing the terminal driver has given rise to a large number of *ad-hoc* solutions which have been applied without any regard to the original problem. So, when you're fixing something make sure you are solving the right problem.

☞ Coffee (no Guinness, thank goodness) ☜

Item 27: 10.36am **Jaap Akkerhuis, Maths Centre, Amsterdam**

Typesetting and Ditroff

Jaap's talk started with a brief history of typesetting from Guttenberg to the present day. I did not really take notes on this bit and I hope he will write it out for the newsletter.

The original **troff** was severely limited by the CAT typesetter at which it was aimed. Device independent **troff** is a recoding of this original program to remove all the CAT dependencies and allow **troff** to be used with other typesetters and typesetter-like devices. Some extra graphics commands have been added to **troff** to allow pre-processors like **pic** to draw pictures in the text. Jaap's experiences with the new package have generally been good.

Item 28: 11.32am **Tom Horton, National Semiconductor Corp**

Virtual Memory management in GENIX

I reproduce the abstract, not because I was asleep, but because the talk was too technical to take adequate notes on.

'We have recently ported UNIX to a NS16032-based system. The port is based on 4.1BSD UNIX and is called GENIX‡. We discarded most of the 4.1BSD virtual memory support code and developed our own, in order to support the virtual memory architecture provided by the NS16082 Memory Management Unit.

A GENIX process has a virtual address space of up to 16 megabytes and is mapped by a two level hierarchy of page tables. Both process pages and second level tables can be paged out of and faulted into memory.

The kernel's address space is mapped by its own set of page tables. Because the user and kernel page tables have the same structure, the kernel manages its own address space with the same algorithm which is used to manage the user address space.

† As the UK people know, we have had a scrolling terminal driver for years because it is very annoying to see your output disappear up and off the screen

‡ GENIX is a trademark of National Semiconductor Corporation.

Sharing of texts is done by mapping executable files. When a file is executed for the first time, the kernel creates a file map for using level 1 and level 2 page tables. The process's page table entries are then set up using the file's map as a template.'

The overall impression is that the hardware looks to be correct and does not have the fatal lack of page reference bits which the VAX-11 processors suffer from. Also, because the hardware is about right, the software implementation can then be resonably elegant.

Item 29: 12.06pm **David Tilbrook, Imperial Software Technology**

Hacking vs Engineering - The UNIX Dilemma

David gave a vastly amusing talk with 35mm slides of Ada, Lady Lovelace and also pictures of sneakers. His main point, which I hope he will expound in these august pages, was that the majority of programmers in the world are working on extremely huge software projects and they really do not have the tools to cope adequately with what they are doing.

The set of UNIX tools which we are used to having around are not viable when scaled up to very large projects and committees which are set up to attempt to solve the problems of scale generate abominations like ADA.

He is getting interested in making tools to solve some of the problems associated with these large projects.

☞ Lunch ☜

Item 30: 1.45pm **Rob Pike, Bell Laboratories**

Video tape on the Blit terminal

The Blit terminal has a bit mapped display, a keyboard and a mouse. The software runs many UNIX shells in separate windows on the screen. The terminal allows local editing of input and output. The user interface is very elegant and watching the tape, it seemed to me that this was the way I want to talk to computers.

Item 31: 2.03pm **Chris Corbett, University of Essex**

Figure processing within Nroff

There are a couple of pre-processors which will allow the user to insert figures into text generated by **troff**. The main problem is that suitable output devices are extremely expensive. Chris has done some work to insert figure drawing capabilities into **nroff** where the output is on low-cost printers.

The system uses a new preprocessor called **fig**, which is used in the same way as other **troff/nroff** pre-processors. The program can either be used to insert (say) raster bit map images into the output from **nroff** or white space can be left if the output device will not draw.

The package can also incorporate images which were specified in **plot(5)** format. This is very useful because the images which are generated can be previewed easily.

☞ Tea ☜

Item 32: 3.35pm **Ed Keizer, Vrije Universiteit**

The Amsterdam Compiler kit

The Amsterdam compiler kit is an attempt to solve the problem of creating compilers for several languages for a wide range of different machines. The solution which has been adopted is to translate the language into an intermediate language (EM) which in turn is translated into code for the target machine.

EM was designed to be an efficient intermediate code for many languages (Pascal, C, Plain, Algol 68 and Ada) which easily runnable on very many target machines (8080, Z80, 6800, 6809, 8086, PDP-11, Z8000, M68000 and VAX). Generating target code is also easy since EM virtual architecture is a stack machine and all instructions have a single operand. It is also easy to add new machines and compilers because the programs which comprise the package are all table driven. There is also an EM optimiser which is machine and language independent. The package comes with good debugging tools and many test programs.

As distributed, the package consists of a version of **cpp** which can drive either the C→EM or Pascal→EM 'front-end'. This phase can be followed by an EM peephole optimiser and (in the future) an EM global optimiser. The EM is then fed into a number of backends which can generate code for the various target machines. This code is then assembled and linked with libraries.

The Pascal compiler has some deviations from the ISO standard: only the first 8 characters of a name are significant, standard procedures are not allowed as actual parameters in procedure calls, and the scope of a variable begins at the declaration. There are also some extensions to the standard: program modules can be compiled separately, assertions have been implemented, as have mark and release.

The restrictions on the C compiler are: the stack size must be less than 4K bytes, automatic variables must not occupy more than 4K and parameters must not occupy more than 4K bytes. Identifiers have 8 significant characters and external identifiers have only 7 significant characters. The standard says that the type of the **char** can be left to the implementor and for the Kit, **char** is unsigned.

The Kit is now available to academic institutions only for $500, although contacts to the address below from commercial sites will be greeted with the usual outstretched hand. The address is:

A.S. Tanenbaum,
Vrije Universiteit,
Dept. of Maths & Computer Science,
Postbus 7161,
1007 MC Amsterdam, The Netherlands.

Item 33: 4.35pm **Euug Committee**

Panel Session

This was a discussion on the conference, asking people what they liked and disliked about the proceedings and attempting to canvas opinion on what should go on a EUUG conferences. Again my notes are very thin but these two suggestions were made.

There was a *yes* vote for the starting of 'Birds of a Feather' sessions. There should be something in future conferences about artificial intelligence and functional languages.

The End

All that remains is to hand out the awards for the conference. Taurus Technology get *Tee Shirt of the Conference* award. For their tasteful little rose, Zilog get the *Best badge or Logo* award. Tobias from Holland gets the *Most revolting and obscene badge* award. Dave Tilbrook gets the *Most interruptions during presentations* award, this was a close thing. Guinness get the *Most drunk drink*

of the conference award.

Seriously though, thanks should go to Helen Gibbons and Tim Murphy for good organisation of a a good conference.

EUUG
European UNIX† Systems User Group

Newsletter Vol 3 No 4
Winter 1983

UniForum January 1984
Visiting the Zoo in Washington

Teus Hagen

Centrum voor Wiskunde en Informatica
Amsterdam, The Netherlands

The Zoo

During the week of the 17th of January 1984 I visited the zoo in Washington. Everyone at my institute thought I was attending a conference in Washington. Well, they were wrong. Instead I visited the zoo in the basement of the Washington Hilton Hotel. According to the figures of the organiser of the conference (/usr/group, USENIX was co-sponsoring it), the floor was crowded with about 9000 visitors. Well, I had three badges, so there were at least 8997 other 'Homo Sapiens' around. The zoo also exhibited some elephants (like Pyramid, Gould, and DEC VAXes). Some of the elephants just showed their new offspring, like the MicroVAX. Other cages had the giraffes (NCR tower, Cadmus, Zilog and Gould families), lions (Altos, Zilog and Pixel), work-horses (SUN, Dual, Ridge) and even the little mouses were on show (IBM PC/IX, Tandy, Apple and Gould again). In total, 150 different cages in the zoo to bring joy into the life of a simple programmer. Pity, after one and a half days the zoo was closed, the joy was over for Bill.

The exhibition

Some aspects of the exhibition were worthwhile to visit. Choice enough, for nearly all the old UNIX families (hardware and software) presented their goodies. One of the newcomers to the exhibition floor was: AT&T. They showed their Blit (Robert Pike), called the Teletype 5620 and some of the AT&T workbenches.

Amazing was what one can do in a year to have a better quality of UNIX running on a little machine. Let's say experience showed that it cost one year to really have UNIX running after the port is done.

All the big companies had small booths, IBM with the PC/IX machine (based on the Intel 8088 chip) of which the rumour is that the software price will be around $900!, the three chip manufacturers, Motorola (two types of machine), National Semiconductor (Genix) and Intel (who allowed me this time to run some benchmarks), and the old guys like Amdahl (really fast), Hewlett Packard (neat hardware), NCR (the 'Tower'), Honeywell (two types of machine) and Texas Instruments (a newcomer with TI NU). And DEC, who were really the first in the UNIX world, presented their MicroVAX and talking chip. Again no sign of BBN at the exhibition, have they left the UNIX battle field?

The real news was coming from some new manufacturers like Pyramid, Ridge and Silicon Graphics (Steve Bourne is there now). The last showed their space-shuttle flight simulator (3D colour and real time!). That machine filled my eyes with dollar notes. The main trend is to machines based on the Motorola chip, with lots of cache space (4K bytes at least), colour rasters, lots of memory (like 2Mb), and of course super fast (more than 1 MIP).

In general, the market can be divided into four categories of machine:

- A display (white, green, yellow or blue) with a built-in 10Mb Winchester and a floppy. Most do not have real LAN hardware or software capabilities. Personally, I wouldn't like to have one of those awful looking monsters on my desk, even if some of the monsters look like a stack of books (Gould). Most have a dialler, modem and uucp possibilities; these WAN capabilities will be discussed later.
- To try to get away from the awful packaging, the tower-profile was invented (by DEC or NCR?). Almost all manufacturers produce machines in this form (SUN, Pixel, Zilog, Cadmus, etc). Only because of the noise of the fan you will notice the machine under your desk. I could not find the TI NU machine until I saw an advertisement for a low noise fan from Texas Instruments, one does not expect that from someone from Texas.

- Real intelligence can be found in the 'workstations'. Many of them are sold with a raster display. Some examples are the SUN and Cadmus (PCS in Europe) machines.
- The old computers, which you should hide in a computer room: big boxes like VAX (DEC), Pyramid, Arete and the fast Ridge machine. The Amdahl machine was really hidden somewhere.

Some remarks about the (new) machines:

MicroVAX from DEC: it looks like a Micro-11 and behaves like a child bred from a VAX730 mother and VAX750 father. The child walks away with some short Q-bus based legs.

The new Pixel P/35 breed runs just fine. Like Altos and Cadmus they have a LAN implementation based on the Newcastle ideas.

The Texas Instruments NU machine is nice and quiet. They did some good work on the fans and have a new bus architecture (from MIT).

The Pyramid machine was facing the world with 128 registers (or was it 256) in mind. Another fast brother was the Ridge machine, advertised with a speed of 8 MIPS. It was so fast that I good only discover 4 of its MIPS. Anyway, fast enough for me.

The National Semiconductor 16032 chip runs at 6 MHz now, so the IBM PC/IX machine can now run on the 8088, 68000 and 16032 (both from Sritek). One IBM PC was running with a Fujitsu Eagle (440 Mb) at the Sritek booth.

The new Cadmus firm sure know how to advertise, their black box looked like it came from John Player, and it looks like all you can get out of it is cigarettes. I prefer the same machine in a different shaped box and with software that is really running, perhaps they should ask PCS in Germany how to do it.

On some of the machines a little known benchmark program was run. Due to some requests the results will be reprinted in this newsletter.

Software

AT&T, Unisoft, Mark Williams, Microsoft, Wollongong, Whitesmith and HCR are trying to take the lead in software. Certainly it is not longer true that there is no software available under UNIX, see the new edition of the '/usr/group' catalogue to get an impression, but also look at the small interesting ones: Sphinx, Unify, and ...

Give me a name and I'll give you a database management package. All kinds of editors, like the interesting 'Interleave' editor, were presented, and Emacs in all shapes, colours and flavours. Network software as if nobody was communicating at all.

Journals and Other Paper Work

Yes, there can't be a fashion without a journal totally devoted to it. Two new journals totally devoted to UNIX: UNIX Review and UNIX/World. In the latter journal I noticed the best articles. Both carry the same advertisements you find in in Byte, Electronics, Mini Systems, etc. None of the two is pro deo.

Yates were presenting their market overviews.

The Decease of the Zoo

What do you bring to home from a zoo? Well, besides some strange disease I collected the following items: collection of T-shirts (thanks to NIXU and SOL); muppets (Computer Automation, Unisoft); the 'grep' milk cup (Amdahl); disco cap, screw driver and poster (DEC); screw driver (Sritek); yellow matchbox bus (TI NU); plastified business card (Yates); USENET map (plotted) (HP); the name badge of Bill Joy (Usenix); the name badge of Otis Wilson and Bill Murphy (AT&T); and I'm still waiting for the free IBM PC/IX machine (IBM). Others had nothing but joy for me.

The Dinner Problem Solved

After one day of walking through an exhibition you get hungry. Well that didn't turn out to be a problem. Every 'good' company organises a hospitality suite, and you get invited with an invitation card, but even if you don't get a card, it is still no problem. Every floor in the Hilton Hotel had about three hospitality suites, and as there are only ten floors available in that hotel, it was impossible to leave the hotel without being full of cheese, fruit and liquid. The only problem arose on Friday night, no hospitality suites left.

Yes, AT&T arranged, of course, the biggest 'brunch' (or better lunner) than ever.

Technical Talks

Due to a fully filled agenda with meetings, I had no chance to get a good overview of the talks, some of which promised to be quite interesting.

Some of the panel sessions were guided by managers and (technical) directors of companies. Well, I discovered now that marketing UNIX or talking about the future aspects of UNIX can be made really boring. And surprisingly, the technical panel sessions tend to be the same exercise. It is becoming difficult now to attend some real interesting talks.

So some real new announcements from AT&T: the document workbench (another, better, ditroff and lots of document processing stuff), BASIC interpreter, and the MC 68000 software generation system.

The new System V.2 release is out now. It has a few new programs as f.i. mailx, pg, qasurvey and trenter. Lots of improvements like f.i. job control, F77 compiler, shell and new flags for the 'ls'-command. A better documentation set and lots of bugs solved again.

System V will have a library with 'high quality' application programs (editor: third party software) and software for several different computers. The package will be made available in co-operation with a newcomer to the UNIX world, **Digital Research Inc.**

For re-sellers of the System V software there are more flexible royalty schedules. AT&T is talking now to the different OEM's to make new arrangements.

And if we call the initial System V release V.0, and have now V.2 announced, what can we expect if AT&T has internally V.4 and V.6 running? Is V.8 the next one? Or is it just Edition 8 from Bell Labs?

The UNIX Network is Changing

Most of an afternoon session was dedicated to networking. UUCP has been re-written, most of the speed-ups are about the same as is distributed with the EUUG D2R3 software. That is: all data about a site is moved to a special 'site' sub-directory, the error messages make more sense, the login-protocol handler is table driven, the table has entries for all kinds of modems, diallers, port-selectors and internal line configurations. Special care has been taken as regards security, per directory one can specify read, no-read, write, or no-write permissions. Also, the specifications for a site can be made more easily. It works in the same way as the 'termcap' file: f.i.

```
logname = uucpa \
machine = you \ # we call you
validate = raven \ #if you say you are "raven", check for logname "uucpa"
commands = rmail:rnews:uucp \
write = /usr/spool/public:/usr/spool/news \
nowrite = /usr \
noread = /etc \
sendfiles = yes
```

Some capabilities have been added: public encryption to identify remote sites, automatic internal data storage/transmission encryption and gateway facilities.

What is still missing was 'grading', to get mail sent before news is shipped, and 'nicknaming', to

facilitate the change of names.

Mark Horton expects to see a fast growth of the total number of sites connected to the UNIX network. Every PC sold now has a modem, sometimes a dialler, and UUCP. Tandy, IBM and Apple will have a production of some thousands of machines per week, so a figure of 100,000 sites (and names) by the end of 1985 is possible. To try to meet the problems which will come with it, a decision was made to structure the UNIX network in the US in the way it is done in Europe. Within a year one hopes to have created domains of countries, of states or of geographic surroundings, of big companies (AT&T?) and of course with subdomains. The address scheme will be much simpler: *teus@mcvax.UUCP* instead of the string *cbosgd!philabs!mcvax!haring!teus*. The routing information will be kept in the backbone sites, because the database will simply not fit in a PC (yet). Also, time-costs will be included in the routing scheme. An inquiry form to get all the information will be published shortly. The hope is to have a domain oriented addressing scheme in production by the end of this year.

User Group Arrangements

After the talk of Jim McKie 'Where is Europe' at the Toronto conference, the EUUG got ten minutes in the opening session of /usr/group to explain that 'Europe is Here' (Emrys Jones). The representatives of the EUUG had some discussions with the US users groups and made the following agreements:

USENIX: all services given by a group are available (at cost price) to the members of the other group, so in essence the only thing you don't get is voting rights in the associated group. For members of the EUUG:

- Publications like newsletters, announcements and proceedings from USENIX are available via the EUUG secretary. The EUUG secretary can arrange bulk shipping of that paperwork, f.i. the EUUG secretary already has copies available of the Toronto Conference Proceedings (500 pages per book). The price is $30 per copy, if you order it via USENIX you have to pay $15 extra for overseas postage.
- Software distributions are available via the EUUG Distribution Center. Special arrangements have been made with AT&T to make this exchange of software across the Big Puddle possible. A new distribution is on its way now.
- The attendance fee for USENIX conferences is the same as for USENIX members. All arrangements for those conferences should go via the EUUG secretary.

STUG: The same arrangements were made with the Software Tools Users Group. The EUUG is already distributing the STUG software tapes, and a new tape with the communication software to exchange data between machines with different operating systems will be available soon.

/USR/GROUP: With /usr/group arrangements were made to exchange publications. Final arrangements for bulk shipping of their 1983 catalogue are being made now.

Summary:

- The newsletter **;login:** and the **Toronto Proceedings** can be ordered now from the EUUG secretary ($30 for the Proceedings).
- The USENIX **Software Distribution 84.1** can be ordered from the EUUG Distribution Center, CWI, Kruislaan 413, 1098 SJ Amsterdam, Netherlands. This distribution contains licenced material, so you have to enclose a copy of your licence (preferably SV and 4.2BSD). If necessary, the EUUG will verify the licence agreement with the licensor. The tape is expected in a few months.
- The /usr/group **Catalog Edition 83** can be ordered from the EUUG secretary. The EUUG secretary will try to organise bulk shipping. The price is $25 (exclusive of postage costs). Price to non-members is $50.

In the future, the secretary of each national group should do the ordering. This will save some time delay as those secretaries will (probably) forward the order electronically to the source.

Conclusions of the Visit to the Zoo

The exhibition is going in the direction of the Comdex fair shows, some booths had only salesmen around with no real knowledge of UNIX at all (I wonder if in the future there will be any technicians around, just in case a screwdriver is needed?).

The 'technical' talks were of a low quality. If not organised in a different way, the talks will end up at the same level as at the Comdex fairs.

A one day course about some 'internal aspects of UNIX' (C style and portability, advanced shell programming, advanced editing, UNIX and LANs) for $100 is really cheap. The general feeling was that those courses were good.

I would like a change in the way the conference is organised now in the US. It is only possible once to get so many people on their feet and give them so little for their money.

Two programs, many UNIX systems (reprint)

Andrew S. Tanenbaum
Vrije Universiteit, Amsterdam

Teus Hagen
Mathematical Centre, Amsterdam

UNIX meetings gives a splendid opportunity to run test programs on the machines present at the exhibition. At the recent meetings and exhibitions in Europe and the US, we have run two test programs on a wide variety of machines. Test program #1 measures CPU/memory speed; test program #2 measures I/O speed. Program #1 was tested six times, with the 'TYPE' declared in six different ways: short, register short, int, register int, long, register long. On the small machines, the test were generally made in single user mode; on the large mainframes we had the share the machine with other users.

The programs:

```
/* Test 1 - CPU/memory */
main()
{ TYPE i, j, k;
  for (i = 0; i < 1000; i++)
        for (j = 0; j < 1000; j++)
              k = i + j + 1983;
}
```

```
/* Test 2 - I/O */
main()
{ int i, j, n;
        char a[512];
        if ( (n=creat("foo",0755)) < 0)
              perror("error: foo");
  for (i = 0; i < 500; i++)
        write(n, a, 512);
}
```

Notes:

The times reflect a combination of several factors, among them, the CPU type, the clock rate, the speed of the memory management unit, the speed of the memory itself, the width and speed of the bus, and last, but certainly not least, the quality of the C compiler used on the machine. Also, the times were obtained using the time(I) command. There is reason to believe that not all vendors understand that 50 Hz != 60 Hz, which makes some of the times slightly suspect.

Conclusions:

None. You should take these measurements with a grain, or better yet, an imperial gallon, of salt. For example, comparing the PDP-11/70 with the SUN, we see that for test #1 and register short, the PDP-11/70 is nearly three times faster, but comparing register long for the same two machines, the SUN is twice as fast. The difference can be explained by the fact that the PDP-11/70 really is faster, but uses memory instead of registers for register longs, whereas the the SUN uses the 68000's hardware registers.

Goal:

Our goal in making these measurements is to stimulate you into making your own measurements and to make you cautious when looking at (carefully selected) comparisons thoughtfully supplied by vendors. Remember: Figures don't lie, but liars figure. If anyone wants to run the tests on other machines, we would appreciate hearing from you.

times in seconds: machine	MHz	usr short	usr short reg	usr int	usr int reg	usr long	usr long reg	real cc#1	real cc#2	#2	year	rem
DEC:												
VAX 780		8.8	8.6	6.7	4.7	6.7	4.7	3	3	2	83	
VAX 750		16.2	16.5	10.9	7.3	10.8	7.3	11			84	
VAX 750		18.9	19.3	13.0	8.5	11.6	8.6	4	4	2	83	
MicroVAX		28.2	28.3	22.2	12.4	22.2	12.4	12			84	
VAX 730		37.4	37.5	23.9	14.4	24.1	14.4	11	12	6	83	
11/70		7.4	2.8	7.4	2.8	14.3	14.3	11	12	14	83	
11/60		10.4	4.2	10.4	4.2	20.2	20.2	17	19	16	83	
11/44		11.2	5.7	11.2	5.7	21.8	21.8	16	8	21	83	
11/45		19.1	7.9	19.1	7.8	38.5	38.5	12	20	12	83	
11/34		22.0	10.9	22.0	10.8	44.8	44.8	18	19	14	83	
11/24		27.7	12.2	27.4	12.4	57.0	57.5	21	20	19	83	
AEDS11(/23)		34.5	14.9	34.6	14.9	72.2	72.2	28	30	8	83	
micro 11(/23)		36.8	16.2	36.9	16.2	76.8	76.8	29	31	24	83	
68000's:												
Plexus P/35	12.5	8.9	5.3	10.0	5.1	10.1	5.2	12			84	
Ch Rivers	12.5	10.1	6.0	11.8	5.9	11.8	5.9	12			83	
QU68030	10	12.0	6.0	13.0	6.0	17.0	8.0	8	11	4	83	
Parallel	12.5	11.6	7.0	13.5	6.7	13.5	6.7	17	18	8	83	
Altos	8	13.9	13.9	13.9	13.9	17.8	17.8	18	25	5	83	
SUN 1	10	13.0	7.8	14.6	7.6	14.6	7.5	9	10	5	83	
Cyb	8	15.3	9.0	15.3	8.2	17.5	8.2	27	28	12	83	
Momentum 32E	8	14.5	8.6	16.7	7.6	16.7	7.6	22			83	
TI NU	10	13.8	6.7	18.2	6.5	17.7	6.6	16			84	3 usrs
Codata	8	17.1	10.5	19.3	10.5	19.3	10.5	28	30	28	83	
CCI	8	20.5	9.3	20.5	9.3	29.9	13.2	13			84	
Pacific	10	18.1	10.7	20.7	9.7	20.8	9.7	24	23	13	83	
Power 520	8	21.1	9.2	20.7	9.2	31.4	12.5	16	16	19	83	
Hawk 32/E	8	19.0	11.3	21.5	10.3	21.5	10.3	29	19	27	83	
Pixel 100/AP	8	18.6	11.0	21.5	9.6	22.9	9.6	44	47	10	83	
Corvus	8	19.8	11.6	22.5	10.1	22.6	10.1	40	42	no sp	83	
QU68000	10	24.5	11.9	24.5		33.0	15.1	10	10	7	83	
Fortune	8	21.7	12.8	25.0	12.4	25.0	12.3	18	20	6	83	
Apple Unisoft	5	22.6	13.7	25.9	12.1	25.9	12.1	41	43	28	83	
Apple Xenix	5	22.7	13.9	26.0	13.0	25.9	13.0	58	65	no sp	83	
Altos-12	5	26.7	14.1	26.7	14.1	53.7	53.9	31	33	13	83	
Plessey S68	8	23.9	13.9	27.5	12.8	27.5	12.6	30			83	
Wicat WS150	8	24.8	14.4	28.1	13.1	28.1	13.1	19	21	12	83	
Victory 68K	10	23.7	15.0	28.3	12.6	29.1	12.5	19			83	
TRS80	8	25.2	14.9	28.3	14.0	28.4	14.2	23	28	16	83	
Codata	8	25.5	14.9	29.2	12.8	29.2	12.8	16			84	
Dual	8	26.9	15.6	30.7	13.3	30.7	13.3	21	20	27	83	
Four Phase	8	27.4	16.2	31.1	14.9	31.1	14.9	32			83	
Ch Rivers	8	28.2	15.8	31.7	15.8	31.7	15.8	28	42	14	83	
Unistar 200	8	28.7	16.5	32.8	14.3	32.8	14.3	36	40	28	83	
IBM Sritek	8	30.3	17.6	34.2	17.3	35.1	16.9	31	35	18	83	ut>rt?
IBM PC Idris	8	37.9	22.7	37.9	22.7	73.3	73.3	18	26	39	83	
Cosmos Antaris	8	34.1	19.5	38.7	16.4	38.7	16.4	26	27	9	83	
ULAB	8	37.2	21.0	44.1	18.9	44.1	19.0	38	44	15	83	

times in seconds: machine	MHz	usr short	usr short reg	usr int	usr int reg	usr long	usr long reg	real cc#1	real cc#2	#2	year	rem
Z8000's:												
Z6000	5.5	13.6	6.3	13.6	6.2	23.2	12.6	20			83	
Zilog	6	14.7	7.3	14.7		25.7	13.3	20	20	8	83	
Plexus	5	15.2	7.0	15.4	7.0	27.5	27.6	24	26	13	83	
ONYX	4	15.9	7.2	15.9	7.3	23.8	14.1	14	14	8	83	
Bleasdale	4	33.3	15.6	33.3		56.2	56.2	20	21	16	83	
8086's:												
Altos	10	13.7	7.2	13.7	7.2	27.8	27.7	18	20	7	83	
Intel	8	29.2	17.6	29.0	17.5	59.1	59.0	17			84	
SBC 86/12A			68.1	83								
8088's:												
IBM XT	4	53.4	30.0	53.4	30.0	108.8	108.8	25			84	
286's:												
Intel	5.5	17.3	10.9	17.3	10.7	34.6	34.6	27			84	
16032's:												
National	6	27.4	25.6	29.3	14.0	32.4	11.7	20			83	
IBM Sritek	6	27.4	27.5	32.1	13.4	30.7	13.6	103			84	
National	4	49.9	45.0	56.7	23.3	57.4	27.2	47	49		83	mem flt
others:												
Amdahl		0.5	0.5	0.3	0.3	0.3	0.3	10	5	1	84	1 usr
Concept32/87		1.5	1.5	0.9	1.4	1.0	1.4	10	16	2	83	1 usr
Pyramid		3.3	3.3	2.0	1.9	2.0	2.0	9			84	
Ridge		7.1	2.9	4.0	1.6	4.0	1.6	19			84	
Eclipse		4.6	4.6	4.4	4.4	4.4	4.4	28			84	
Arete		5.7	5.7	5.7	5.7	5.6	5.8	15			84	
HP 9000		9.4	9.4	7.4	7.4	7.4	7.4	28	26	3	83	
Concept32/27		12.0	11.0	10.0	10.0	10.0	10.0	18	20	5	83	
BBN C/60		14.5	8.2	14.6	8.2	47.2	30.3	7	9	3	83	
PE3210		16.7	6.7	15.9		15.9	6.7	4	4	3	83	
Perq 1		44.5	15.6	22.2			15.0	25	25	7	83	
Perq 2		46.7	15.1	23.0	15.0	22.9	15.1	20			83	
IBM S/1 4954		37.2	37.1	37.1	37.1	62.3	62.3	28	30	32	83	

THE AMSTERDAM COMPILER KIT

Ed Keizer
Andrew S. Tanenbaum
Hans van Staveren

Dept. of Mathematics and Computer Science
Vrije Universiteit
Postbus 7161
1007 MC Amsterdam, The Netherlands

Telephone: 31 (20) 548 5392
UUCP: ..!mcvax!vu44!keie

1. INTRODUCTION

The cost of producing compilers for high-level languages is becoming one of the major cost factors in producing new systems. Traditionally, a separate compiler was produced for each combination of language and machine. The Amsterdam Compiler Kit is designed to ease the production of new compilers. This paper describes the components of the kit and some experiences with a few of the compilers we produced.

For each high-level language in our kit we have a program, called front end, that translates the source into an intermediate code, called EM. For each machine language in our kit whe have a program, called backend, that translates EM code into assembly language. The use of an intermediate language has several advantages:

- adding one front end for a new language allows use of that language on several machines.
- adding a back end for a new machine allows the use of several high-level languages.
- it is possible to use optimizers on the intermediate code, thereby lifting a burden from the shoulders of the front end writer.

One of the design goals of the architecture of our intermediate language was to ease the task of the front end writer.

The idea of producing compilers using a common intermediate code (often called an UNCOL) is hardly new. What we have done is work out the details and actually make a practical implementation that runs on UNIX and produces high-quality compilers for a variety of languages and machines.

In the following sections we will describe the various components of the tool kit in some detail. These are shown in Figure 1.

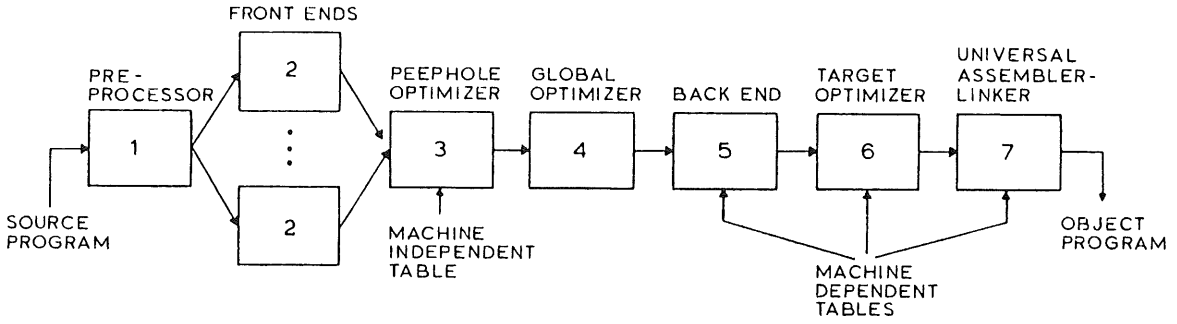


Fig. 1. The Amsterdam Compiler Kit (ACK).

2. THE PREPROCESSOR

The preprocessor is the standard C preprocessor used by the UNIX compilers. It handles textual inclusion of header files, macro expansion, and conditional compilation, making these facilities available in all the languages in the tool kit.

3. THE FRONT ENDS

The front end converts the preprocessed source program into EM, which is a simple stack machine with few of the idiosyncracies of conventional machines. Producing EM code for expressions, for example, is not much harder than producing reverse Polish. As an example of EM code, consider the following C statements, in which the letters I, J, and K all designate local integer variables:

```
I =  5*J + K;
F(I, J-3);
```

The corresponding EM code for an implementation with 2-byte integers is as follows, where semicolon is used to introduce comments:

```
LOC 5        ; PUSH CONSTANT 5 ONTO THE STACK
LOL J        ; PUSH J ONTO THE STACK
MLI 2        ; MULTIPLY INTEGERS OF LENGTH 2 BYTES
LOL K        ; PUSH K ONTO THE STACK
ADI 2        ; ADD INTEGERS OF LENGTH 2 BYTES
STL I        ; STORE THE TOP OF THE STACK IN I
LOL J        ; PUSH J ONTO THE STACK
LOC 3        ; PUSH CONSTANT 3 ONTO THE STACK
SBI 2        ; COMPUTE J-3
LOL I        ; PUSH I -- NOTE: C CALLING SEQUENCE PUSHES ARGS BACKWARDS
CAL F        ; CALL THE FUNCTION
ASP 4        ; ADJUST STACK POINTER BY 4 BYTES TO REMOVE THE ARGUMENTS
```

Each front end is written as a separate program. The Pascal front end, for example, is written in Pascal itself and uses a recursive descent parser. The C, front end, in contrast, uses a bottom up technique. The calling sequences for procedures in both languages are so similar that programs can be written in a mixture of both languages.

The EM code can either be further processed for compilation to target machine language or assembled to a binary form for interpretation. The latter route speeds up compilation and and is mostly used during program development since the interpreters can check for run-time errors and gather detailed execution statistics.

4. PEEPHOLE OPTIMIZATION

The local, or peephole, optimizer reads the EM file and produces a new, improved EM file. It maintains a small window on the file in which it looks for the patterns present in its driving table. When a pattern is found, the replacement text is also taken from the driving table.

The approximately 400 optimizations present in the driving table can be roughly categorized into the following major groups:

1. Constant folding
2. Strength reduction
3. Reordering of expressions
4. Removal of unreachable code
5. Removal of null instructions
6. Group moves
7. Use of special instructions

In addition, the peephole optimizer also does branch chain compression and a few other optimizations that are not strictly table driven.

By putting all the peephole optimization in one place, we have an optimizer that can be used with all languages and all machines in the tool kit, eliminating the need for each compiler writer to do his own optimization.

5. THE BACK END

In contrast to the front ends, which are a collection of distinct programs, the back end is a single table-driven program. It reads the EM file and generates target assembly code by simulating at compile time the behavior of the run-time stack. The driving table for a given machine has a series of lines, each one having five parts:

The EM instruction to be translated
The stack configuration when the line can be applied
The code to be generated
The new value of the stack after the line has been applied
The time and space costs of applying the line and generating its code

As each EM instruction is read, the table is searched for lines that apply to it. For example, the EM code for I = J + 3 , consists of LOL J followed by LOC 3 and then an addition. At the time of the addition instruction the simulated stack has a constant on top and a (local) variable just below. Thus the back end must find a line in its table telling what to do with an ADI 2 instruction in the context of a constant and a local variable. If such a line exists, it can carry it out.

On the other hand, if no such line exists, then it must find some other way to generate the code. For example, if there is a line telling how to do an addition with a register variable and a constant on the stack, it could then try to find a line telling how to get a variable into a register, and then apply the former line. In general, there will be several legal code sequences. In this example one could first load the constant into a register and then add the variable to it; alternatively, one could first load the variable and then add the constant to it. On some machines it may not matter, but on, for example, the TI 9900, a cheap instruction to move a small constant into a register exists, but no corresponding instruction exists to add a small constant to a register, so the first code sequence is better than the second one.

The code generation algorithm is now simple to describe. Try each of the legal code sequences. Each one has a cost and leads to a new stack configuration. From each of these configurations, a new EM instruction is read in and all the legal code sequences generated, and so on. In short, a weighted tree is built, not unlike a chess-playing program does. Using this tree, the first step along the most "efficient" path is taken, and the whole process repeated. The user may specify "efficient" to mean time or space or some linear combination thereof.

6. THE UNIVERSAL ASSEMBLER

The final component in the main path is the universal assembler. It also serves as the linker, reading in libraries and extracting routines where needed. Like the back end, it is parameterized by a machine-dependent driving table. It is surprising how machine-independent an assembler can be. Our experience is that a student who knows nothing about the tool kit or assemblers can produce an assembler for a machine such as the Z8000 in three weeks, and that an experienced programmer can write one in about one week.

7. INTERPRETERS

An alternative way of executing programs is to take the EM code from the front end and assemble it to a special, highly-compact binary representation and then interpret it. This approach yields faster compilation at the price of slower execution. Interpreters can also provide substantial run-time checking, debugging, and profiling information and are especially useful during the development phase. Currently, all our interpreters are written in the assembly language of the machine on which interpretation is to take place.

8. GLOBAL OPTIMIZER

The global optimizer is currently being developed, it will look at each program as a whole and try to find ways to speed up execution. Three distinct types of optimizations can be found here:

1. Interprocedural
2. Intraprocedural
3. Basic block

Interprocedural optimizations are those spanning procedure boundaries. The most important one is expanding procedures in line, especially short procedures that occur in loops and pass several parameters or procedures calls with constant actual parameters.

Intraprocedural optimizations take place after analysis of individual procedures. Much attention is given to loops inside a procedure. The global optimizer tries to move code outside the loop and attempts to allocate registers to variables used inside loops.

A basic block is a piece of code that can only be entered at the top and left at the bottom. The global optimizer tries to transform arithmetic or boolean expressions into forms that are likely to result in better target code. A simple example of this is transforming -B + A < 0 into the simpler A < B.

9. DISCUSSION

To get an idea of how well the compilers produced by the tool kit compare with other portable compilers, we compiled 73 programs from /usr/src/cmd using both our compiler and pcc on the PDP-11 and the VAX. This test did not use the -O flag of pcc, nor the analogous program in our system, the target optimizer. Furthermore, all the register declarations were replaced by auto declarations for both compilers to to compensate for the fact that our system currently stores all variables in memory.

The ratio of our code to pcc code on the PDP-11 varied from 0.87 (our code was smaller) to 1.28 (pcc code was smaller). The distribution was as follows:

ack size/pcc size	occurrence frequency
0.00-1.00	8%
1.01-1.05	20%
1.06-1.10	35%
1.11-1.15	24%
1.16-1.20	5%
1.21-1.25	4%
1.26-1.30	4%

On the average, our code was 9% larger than pcc code.

The ratio of our code to C-compiler code on the VAX varied from 0.91 (our code was smaller) to 1.21 (cc code was smaller). The distribution was as follows:

ack size/cc size	occurrence frequency
0.90-0.95	1.4%
0.95-1.00	2.7%
1.00-1.05	16.4%
1.05-1.10	35.6%
1.10-1.15	28.8%
1.15-1.20	12.3%
1.20-1.25	2.7%

On the average, our code was 9% larger than cc code.

Our system is still being tuned, and improvements will no doubt be generated in the future, especially when the global optimizer becomes available.

We also translated these 73 programs to Intel 8086 code. We could not compare the quality of that code with code produced by other 8086 compilers. We did compare the code size for the same programs on Intel 8086, VAX to the code produced by our compiler kit for the PDP11. The distributions were:

Intel 8086/PDP 11	occurrence frequency
0.70-0.75	1.7%
0.75-0.80	10.0%
0.80-0.85	41.7%
0.85-0.90	43.3%
0.90-0.95	3.3%

Intel 8086 code is, on the average, 17% smaller than PDP11 code.

Vax/PDP 11	occurrence frequency
0.70-0.85	4.2%
0.85-0.90	7.0%
0.90-0.95	8.5%
0.95-1.00	11.3%
1.00-1.05	14.1%
1.05-1.10	16.9%
1.10-1.15	14.1%
1.15-1.20	14.1%
1.20-1.25	5.6%
1.25-1.35	4.2%

The code for the VAX is, on the average, 5% larger than PDP11 code.

10. CURRENT STATUS

The current status of the project as of June 1983 is as follows:

Front ends: Pascal and C done; Plain and Algol 68 in progress
Peephole optimizer: done
Back ends: PDP-11, VAX, 68000, 8086 done; Z80, 8080 Z8000, 6809 in progress
Assemblers: 6800, 6809, 68000, 8080, Z80, Z8000, 6502, PDP-11 done
EM interpreters: Z80, PDP-11 done; 68000 in progress

The tool kit itself has been tested on the PDP-11 under UNIX V7, on a VAX 11/780 under Berkeley UNIX, and on a 68000 under System III. In principle, moving to other UNIX systems should not cost more than a week or two.

The tool kit will be licensed to universities holding an academic UNIX source license from Western Electric for a moderate charge (probably $500) starting in September 1983. This distribution will contain the complete sources of all the programs, and a large amount of additional information, including libraries, programs to test new back end tables, documentation etc. It will also be made available to corporations holding a UNIX source license for a licensing fee to be negotiated, depending on which rights the corporation desires (e.g., in-house use vs. sale to customers; national rights vs. world-wide rights, etc.). Interested parties should contact the authors.

MAPPING THE UUCP NETWORK

Rob Kolstad

CONVEX Computer Corporation

Karen Summers-Horton

The UUCP network encompasses those machines on the UNIX News Network (Usenet) and more: over 2100 sites as of January 11, 1984. This talk details the difference between the UUCP network and Usenet in addition to outlining the current problems in using the networks (notably those concerning reliable routing among sites: "how to get there from here"). If the UUCP network is to become an ARPA domain, reliable maps and site descriptions must be available to a "name server". This talk will also explain our current plans for mapping not only the connectivity of the network but also the "quality" of the connections for use with routing programs such as Steve Bellovin's optimal path finder. The talk will include current schema for collecting, disseminating, and using the information.

1. UUCP and Usenet

UUCP (Unix to Unix CoPy) is a file transport mechanism which also features the ability to request execution of a few commands at remote sites (e.g., rmail and rnews). It is used primarily to send electronic mail and news. The network is not always "connected": a host often calls another site only on a schedule or when traffic is waiting for the remote site. AT&T Bell Laboratories designed UUCP in 1976 with the assumption that all machines had an auto-dialer and could call all other machines directly (a valid assumption when there were only a few hosts running UUCP). Nowadays, each host connects to one or several remote machines. These hosts range in size from small microcomputers (e.g., Radio Shack TRS-80 model 16's) to giant multi-user systems.

UUCP and rmail use a clever idea to move mail to sites not directly connected to a host. The program "rmail" examines a mail header (which specifies sites connecting the original host and the mail's destination) and sends the mail along the first site listed on the list after discarding it. To send mail from site 'parsec' to user 'joe' at site 'adec23', a path like this emerges:

allegra!alberta!sask!hssg40!adec23!joe

Clearly, it is of utmost importance to know who talks to whom! These paths become more complicated when internetwork routes enter the picture. It is possible to send mail from the UUCP network to the ARPANET, DEC E-NET, BITNET, and many local networks. Today, over 2100 hosts run UUCP; this is where our problems begin.

A fraction of these machines (approximately one-third) also belong to a logical subnetwork called "Usenet". Usenet is an electronic bulletin board connecting about 700 sites in the United States, Canada, Europe, and Australia. A person at any Usenet site may post articles to any specific bulletin board (called a "newsgroup") and (eventually) reach all people who subscribe to that newsgroup.

News is transferred by various means: UUCP, local area networks, and long haul networks to name a few. Maintenance and development of Usenet software is done on a volunteer basis; the cost for transferring information is borne by each individual site. A Texas site pay $2,000 to $4,000/year to the phone company for long distance charges for news and mail. Some sites pay nothing: Southern Methodist University (with no budget) is polled by local industries. Digital Equipment Corporation maintains some overseas links; their phone bill is around $19,000 per month.

2. Problems with UUCP

Along with the advantages of low cost and ease of use, the UUCP network has problems. In summary, these problems are:

a. Explosive growth
b. Addressing reliability
c. Lack of backward error recovery
d. Unknown and unpredictable propagation delays

We will address the problems of growth and addressing here.

UUCP is growing by leaps and bounds. Our current guess of over 2100 hosts is just that: a guess. There is no official central registry of UUCP hosts. When someone joins UUCP, there is no official procedure. Once they find someone who will let them hook up to their site, they then can mail to the whole net without anyone except their immediate neighbor really knowing who they are. As a result, trying to get mail to John Smith at Framus, Inc. is difficult unless you happen to know the path through the network to the Framus computer. With over 2100 sites, (most of which are not directly connected to each other), this becomes nearly impossible. To find your way through the net, you need a road map. Our project is to provide that map.

Since the net is growing so quickly and since it is so easy to set up or tear down a link, the net's structure is never fixed. Not only must the initial map be collected, but it must be constantly maintained or it will be quickly out of date. This maintenance will entail an enormous effort.

Our current guess based on current growth patterns and computer sales is is that the net will increase to between 15,000 and 50,000 sites over the next five years. One reason is the increasing popularity of personal computers, whose owners will want the convenience and relative low cost of UUCP as an option. We must plan now for this explosion.

Once a map exists in machine-readable form, it is a simple matter to use that map to route mail. This means that instead of typing in long routes such as:

cbosgd!mhuxl!eagle!harpo!seismo!hao!hplabs!hpda!fortune!amd70!decwrl!joe

we type:

joe@decwrl.uucp

This not only saves a lot of typing but allows the machine to pick a "best" route. In this case it might have chosen decvax!decwrl!joe, a route the person may not have known about.

3. Becoming a Domain

For the past 12 years, the ARPANET has used a user@host mailing address syntax. A year ago they realized that a flat addressing structure such as this caused impractical maintenance problems; they then converted to a hierarchical user@domain syntax. The domain is a list of words separated by periods (e.g., F.ISI.ARPA) forming a hierarchy with the top level domain, ARPA, at the right. This method allows a very large number of hosts to be named without any host needing a complete list of all other hosts. Other top level domains, besides ARPA, are possible.

We propose to make UUCP a top level domain. This will enable users on UUCP machines to exchange electronic mail with users of machines in other ARPA domains, such as the ARPANET and CSNET.

In RFC 881*, the ARPANET sets forth a list of requirements for top level domains. These requirements are:

1. There must be a responsible person that can act as coordinator and answer questions. In addition to serving as a central contact point, this person will have the authority to enforce network policies and rules.

Request for Comments 881, Network Information Center, ARPANET.

2. There must be a robust name service. There will be two separate name server machines to which one can send a mail message and receive information back about a particular site.
3. The domain must be of a minimum size. Since UUCP currently has over 2100 hosts, this should be no problem.
4. The domain must be registered with the central domain administrator.

We intend to meet these requirements. In addition, since we will be keeping a list of all host names, it will be possible to ensure that there are no duplications. The UUCP software allows host names of any length, but only uses the first six (System V) or seven (others) characters of the name. We will ensure that all host names are unique in the first six characters and are chosen from a set that will not cause trouble on other machines (lower case letters, digits, and a few punctuation characters such as hyphen and underscore).

The authority issue is slightly sensitive, since there is currently no authority over the UUCP network. Such ultimate authority is, however, necessary to protect other domains and other UUCP machines against unacceptable behavior. Such behavior might involve flooding another machine with large quantities of unwanted mail or generation of traffic that fails to conform to accepted standards and breaks programs on other machines. If a host continues to cause problems after repeated warnings, the site would lose their entry in the data base and their claim to their UUCP host name. We expect use of this authority to be extremely rare. The corresponding authority has existed on the ARPANET for years, yet no site has ever been disconnected because of it.

4. Pathalias

To solve the problem of determining "good" routes to foreign machines, Steven Bellovin has written a program called "pathalias" which he has placed in the public domain. Given a list of sites and their direct neighbors, pathalias computes an "optimal" route from the local host to each other host on the network. Upon receiving a new database, the site administrator runs pathalias on the database and stores the result in a routing file. Any program can then look up the best route to any site in the routing file.

Here is a sample database for a four host network. For each host, the name of the host is given, followed by the names of all sites to which mail can be sent directly. For example, allegra can send mail directly to all three of the other hosts, but gummo can only send mail directly to allegra. In this example, all links are two-way; in the real world, most but not all links are two-way.

```
allegra  cbosgd, gummo, ucbvax
cbosgd   allegra, ucbvax
gummo    allegra
ucbvax   allegra, cbosgd
```

The output from pathalias, run on host cbosgd, is:

```
0       cbosgd                  %s
4000    allegra                 allegra!%s
4000    ucbvax                  ucbvax!%s
8000    gummo                   allegra!gummo!%s
```

The first column indicates the "cost" (this is calculated using defaults); the second column is the machine name; the third column is the mail route to the machine.

Many circumstances can cause routing through a given machine to be more desirable (or less so). Some connections may cost more than others since some links go over leased lines, local area networks, or expensive long distance lines. Some sites may not have an autodialer or may not be able to afford long distance phone calls. Such sites cannot call their neighbors on demand, but must wait to be polled. The frequency of polling varies from every hour to seldom or never, depending on the amount of traffic coming from the other site. Some links are more reliable than others.

Pathalias can find the "lowest cost" path from the local site to each other host on the net if "costs" for connections ("edges") are supplied. Rather than use the default cost of 4000, as above, we can

attach specific costs to each link. In the example below, DEMAND is 300, HOURLY is 500, DAILY/2 is 2500, and DAILY is 5000.

```
allegra  cbosgd(DEMAND), gummo(DAILY/2), ucbvax(HOURLY)
cbosgd   allegra(DEMAND), ucbvax(DEMAND)
gummo    allegra(DAILY/2)
ucbvax   allegra(HOURLY), cbosgd(DAILY)
```

The pathalias computations become:

```
0       cbosgd                          %s
300     allegra                         allegra!%s
300     ucbvax                          ucbvax!%s
2800    gummo                           allegra!gummo!%s
```

Changing the "local host" to ucbvax yields:

```
0       ucbvax                          %s
500     allegra                         allegra!%s
800     cbosgd                          allegra!cbosgd!%s
3000    gummo                           allegra!gummo!%s
```

The pathalias program can also compute paths to other networks. We believe it is a powerful enough tool to solve the routing problems -- once an accurate database is collected.

5. Data Collection

Currently, there are a few databases scattered throughout the network. Karen Summers-Horton keeps track of those Usenet sites which receive the newsgroup net.announce. Their information includes the site's name, the contact person, electronic and US Mail addresses, and hosts with which the machine exchanges news. Several sites keep track of extensive L.sys files and can call hundreds of machines. These sites have an easy time of finding routes. Rob Kolstad maintains a list of network "edges": connections between pairs of machines. The list is not of ultimate utility since it contains neither connection frequency nor direction. Even though some hosts call others on demand, it is not a good assumption that mail can flow back the other way.

Rob Kolstad (allegra!parsec!kolstad, CONVEX Computer Corp., Dallas), Scott Bradner (allegra!wjh12!sob, Harvard University), and possibly a handful of other volunteers are currently involved in an effort to collect an accurate enough database to generate pathalias style routes for any given machine. Here is a typical entry in their database:

```
=Name:              parsec
=Machine Type/OS:   VAX780/4.1cBSD
=Organization:      PARSEC/Convex Computer Corporation
=Contact Person:    Rob Kolstad
=Electronic-Addr:   parsec!kolstad
=Phone:             214-669-3700
=Postal-Address:    1819 Firman #151, Richardson, TX  75081
=Long/Lat Coors:
=Comments:
=Editor:            parsec!kolstad Tue Jan  2 09:07:00 1984
=

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
#
parsec          unmvax(DAILY/3), rice(DAILY/3), uiucdcs(DAILY/3), ctvax(DEMAND),
                rice(DAILY/3), allegra(DAILY/3), dj3b1(DEMAND), smu(DEMAND)
```

The entries are stored in compressed format (but easily expanded). They have enough information for pathalias to make optimal routes and also include enough site information to solve most problems that come up that require contacting the site.

The database currently contains entries for over 2000 sites. Typically they look like this:

```
=Name:                  acsa
=Machine Type/OS:
=Organization:
=Contact Person:
=Electronic-Addr:       acsa!root
=Phone:
=Postal-Address:
=Long/Lat Coors:
=Comments:
=Editor:
=
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
#
acsa                    inuxc(DAILY)
```

It is our goal to fill in each of the templates. We currently intend to do this by electronically mailing out our (sometimes partially completed) templates to each machine with directions for completing the form and returning it. We have several scripts and programs to maintain the database and automatically send out mail. We believe the initial data gathering effort will require from 3-9 months.

6. Data Base Maintenance

6.1. Short Term

Short term maintenance of the database will be done strictly by hand. Sites will be encouraged to mail corrections to a central collection point, at which time we will manually update the database. As this will take a considerable amount of time and effort, steps are being taken to automate the process as soon as possible.

6.2. Long Term

In the long term, it is hoped that we can distribute programs (or shell scripts) which will allow simple updating to the database by mailing well-formatted updates to an alias. While this scheme might still require human intervention in the form of "approving" updates, and dealing with improperly formatted requests, it still removes much of the effort from the update process.

We must also develop a "registry" for new sites to consult to verify uniqueness of their site name and enter their initial routing data. While this removes some of the anarchy from the network, it is felt that it is nevertheless a valuable step.

7. Data Base Distribution

Our current database is approximately eight megabytes and growing rapidly, so it is much too large to post to Usenet (even once). However, it is crucial to distribute enough data on a regular basis to allow mail routing programs to work.

One very simple option is to include the database on the USENIX tapes. This option has problems related to timeliness, but an out-of-date database is far more useful than none at all, and it would allow distribution of the entire database.

Another option is to post (on a regular basis) complete information for a small set of backbone sites to Usenet. In addition to this complete information, we would also post a list of all other hosts,

and a path from each host to a backbone site. This should cut the size of our initial distributions by a large factor.

A third option is not to post the entire list of host names. Instead, a hierarchical structure would be worked out in the spirit of ARPANET domains. Thus, an address like cbosgd.uucp might become d.osg.cb.att.uucp, indicating that within the UUCP domain, AT&T region, Columbus location, OSG project, the D machine is specified. No fixed structure such as id.proj.loc.reg.UUCP is implied, each subdomain would subdivide itself as appropriate. The Postal Service and Telephone Company have set up similar hierarchies that can route traffic without any complete lists of all possible locations.

The last two schemes have the disadvantage that they do not explicitly take advantage of the richness of connections that do exist. These problems remain to be solved.

Finally, the name servers to be set up would provide for 'on demand', electronic distribution of directory information for individual sites.

8. Summary

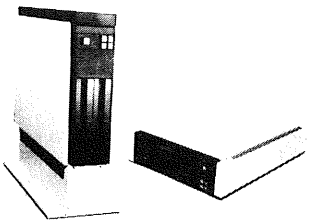
As it stands, the UUCP network is a growing, viable entity which provides very low cost network facilities to well over 2,000 machines. The problems of unreliability, lost messages, and variable propagation delay have been tolerated for many years. Because of the network's growth, its machine-to-machine routing scheme, and lack of centralized control, the requirement for an accurate network-wide map is essential. We need the cooperation of each site in order to construct and maintain a reliable database.

Clippings

The first two pages of clippings come from "Whats New in Computing", May and June, 1984. "UNIX on the Cyber 180" comes from "CSIRONET NEWS", June, 1984, number 177. The bit about UNIX on the DEC Professional 350 (some part of the foregoing has got to be a trademark of the Digital Equipment Corp) comes from "DEC Insight" May, 1984. The "Reader's Comments" come from "Electronics" May 31, 1984. Finally, the rest comes from "The Sydney Morning Herald" July 16, 1984.

32-BIT COMPUTER SYSTEM

The MDB Micro 32 is a 32-bit computer system which has a MC68000 microprocessor and Regulus, a powerful operating system which offers: user source compatibility with UNIX V6, 7 and SYSTEM III; complete support of all UNIX kernel features; multi-key B-tree ISAM and VAX/PDP-11 cross support; and a range of operating systems and command



functions not available in other UNIX systems. The MDB Micro 32 is also offered with a comprehensive multi-user integrated business system written in C including order entry, invoicing, sales analysis, accounts receivable, accounts payable, general ledger and stock control. Other commercial modules are currently being completed. The Micro 32 also allows use of MDB's line printer controllers, disc and tape controllers, 4, 8 or 16 channel multiplexers, synchronous serial interfaces and high speed DMA interface modules. The smallest configuration has 10 Mbytes of hard disc, 512 Kbytes of RAM, two 512 Kbyte floppies, four asynchronous serial ports and a parallel Centronics interface printer port. System memory can be expanded by use of dual size 512 Kbyte memory modules up to a total of 4 Mbytes. The number of users is dependent on the user environment but can be up to 64. Hard disc storage is from 10 Mbytes to 1.3 Gbytes, and a wide range of other peripherals are available.

MDB Systems (Australia) Pty Ltd.
PO Box 384,
Neutral Bay 2089

UNIX OPERATING SYSTEM

AT&T, originator of the UNIX operating system, has validated Motorola's System V/68 operating system, which is the first UNIX port developed jointly with an outside source. Developed for the M68000 family of microprocessors, the operating system is a functional equivalent to the UNIX System V product developed for minicomputer environments. AT&T and Motorola will market the product under the names UNIX System V, M68000 Version Operating System and System V/68 Operating SYstem, respectively. The System V/68 operating system is another software support option on Motorola's EXORmacs Development System and VME/10 Microcomputer System. Source code updates, maintenance, and support of the SYSTEM V/68 and UNIX System V, M68000 Version Operating systems is provided by Motorola. Motorola will also provide similar support for System V/68 object code operating on their development systems, which is available to manufacturers of M68000-based microcomputer systems at both the source and object code levels.

Motorala Semiconductor Products.
250 Pacific Highway,
Crows Nest 2065

DATA ENTRY PACKAGE

iDE-32, a data entry package available on UNIX and Idris operating systems, simplifies data collelction and verification. It requires a cursor addressable screen with protected and unprotected field setting capabilities, and can be used by non-computer personnel. The field on the screen can be an integer, real, alphabetic, alphanumeric or a date (DD/MM/YY). If a field is incorrectly entered an error message is displayed and the operator can set default values. The entered data can be scanned and changed at random. iDE-32 is currently available on the M68000 processor, with a new version for NS16000, VAX and PDP 11 series to be available soon.

Interparser Systems Pty Ltd.
408 Oxford Street,
Bondi Junction 2022

16/32 BIT BASIC COMPILER

iBASIC-32, a BASIC compiler designed for 16/32 bit microprocessors, produces pseudo-code which is interpreted by a run-time system. The compiler supports three types of variables (integer, real and string) and standard BASIC statements along with while and wend statements. The function library is suited to a business environment and runs rapidly. iBASIC-32 is presently available on UNIX and Idris on the M68000 processor, with a version for NS16000, VAX and the PDP-11 series to be available soon.

Interparser Systems Pty Ltd.
408 Oxford Street,
Bondi Junction 2022

SIR/DBMS ON HP9000

SIR/DBMS software has been released for the Hewlett-Packard 9000 series of desktop computers under the HP-UX operating system. The HP9000 offers 32-bit architecture and multi-user configuration with a direct address range of 500 Mbytes. The HP-UX operating system provides a UNIX environment ported directly from AT&T's System III. SIR/DBMS is the nucleus database system of Scientific Information Retrieval's integrated 4GL software. In comparison with the IMAGE database software maintained on the HP9000, SIR/DBMS offers relational models with built-in procedures for report-writing, tabulating data and producing statistics. It also provides extensive functions for use in engineering applications.

Scientific Information Retrieval (Australasia) Pty Ltd.
Suite 15, 83 Mill Point Road,
South Perth 6151

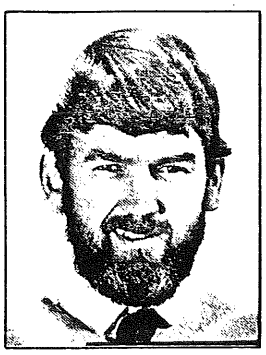
DUAL UNIX APPROACH

Pyramid's OSx operating system for the Pyramid 90X supermini includes the commands of both UNIX System V and 4.2 BSD at the user level, and incorporates all 4.2 BSD performance features plus a variety of Pyramid enhancements internally. Features unique to System V include comprehensive initialisation procedures and expanded system accounting and debugging, while unique 4.2 BSD features include virtual memory, networking, the C-shell, and the interprocess communication package. The features of the 4.2 BSD fast file system, process management and switching include large block sizes (up to 8 Kbytes) for fast data transfer, and the storage of related file blocks in clusters on the disc to reduce head movement and rotational latency, Berkeley's virtual forking mechanism and process handling. OSx also fully uses the I/O and intelligent terminal processors to offload many UNIX I/O driver functions.

Pyramid Technology Australia Pty Ltd.
PO Box 152,
St Leonards 2065

UNIX on the Cyber 180

Peter Hewestón
Controller, User Services, DCR Canberra



As noted on page 2, Control Data have recently revealed that the Cyber 180 series computers have an 8-bit architecture lurking behind the 6-bit machine we now see. The NOS/VE operating system which runs on the 8-bit machine appears to be a very promising development, with all of the features one would expect from a modern interactive operating system.

One of the more exciting announcements from Control Data is that UNIX* will be available under NOS/VE. Interest in UNIX has exploded in recent years, and it is fast becoming the standard operating system that computer vendors ignore at their peril. Most of the major computer vendors either offer UNIX, or are investigating it, and it has made particular inroads into the supermicro and minicomputer marketplace. Even the ubiquitous IBM PC now runs UNIX.

In this context, the announcement of UNIX for the Cyber 180 series becomes especially significant. For the first time, it is possible to envisage a computing facility based on desktop personal computers, supermicro- or mini-based workstations, and very large mainframes – all offering the same operating system and utilities.

The Control Data version of UNIX is known as VX/VE (because of the licencing policies of AT&T, it may not be called UNIX). However, VX/VE will be a full implementation of UNIX System V, the latest release from AT&T.

VX/VE runs as an application under NOS/VE, and shares the same file system. VX/VE files are marked with a special file type, and translation between VX/VE and NOS/VE file types is automatic in each direction. A VX/VE user is able to issue any NOS/VE command via a shell escape mechanism. Most of the standard UNIX tools are provided, including lex, yacc and nroff (for those who understand UNIX-talk). The main commands excluded are those which are inappropriate to an emulation environment, such as system administration, and commands which violate NOS/VE security, such as setuid or su.

A full implementation of the portable C compiler is part of VX/VE, and it produces code which runs under VX/VE or NOS/VE directly.

Users will be kept informed of developments at CSIRONET via *CSIRONET News*.

* Trademark of Bell Laboratories. ■

UNIX* users can now run that operating system on the award-winning Professional 350 system, creating a low-cost, personal UNIX workstation with a unique set of advantages.

PRO/VENIX requires a Professional 350 with the optional 10-Mbyte Winchester disk, makes use of the standard floating point adaptor, and 512 Kbytes of memory. The operating system supports the Professional 350 extended bit-map option; the LA50, LA100, LQP02 printers; and the DF02 and DF03 modems.

Availability

PRO/VENIX was specifically developed for Digital by VenturCom Inc. of Cambridge, MA. It will be distributed, marketed, and supported by Digital through the Digital Classified Software (DCS) program under option number QA718-C3, and is available now. Contact your Digital sales representative for more information. ❑

Knockout punch for cards

THE CLOSING of IBM Corp's last remaining punch-card facility in the United States marks the end of an era in computing.

During IBM's formative years punch card technology formed the core of the company's operations.

However, the coming of more advanced techniques of data storage spelled the end for this relatively antiquated process.

As a result of IBM's decision to get out of the punch card business, the company's biggest punch-card customer, the US Treasury Department, is preparing to change the way it processes the 550 million-plus cheques it mails out yearly.

Punch cards are thin cardboard cards used to store data which is read by detecting the position of the punched holes.

With the passing of the punch-card era, the 35 IBM 1404 punch-card cheque-writing machines that the US Treasury uses will be replaced by hardware made by Honeywell and the Control Data Corp.

Punch cards have been used for electronic tabulation since the late 1800s, when a US Census Bureau worker, Herman Hollerith, developed a mechanised tabulation system.

The system, which used holes in punch cards to denote census-related data, featured an electronic sorter that "felt" the holes in the cards and distributed them to the appropriate bins.

Being a man with a certain amount of entrepreneurial foresight, Mr Hollerith left his secure job in the public service in 1896 to start up the Tabulating Machine Company.

He later sold his company to a collective that re-named it Computing-Tabulating-Recording Corp. Not a catchy name, but that company was the forerunner of IBM.

Refined versions of Mr Hollerith's original punchcard computer formed the core of IBM's business until the 1960s.

AT&T and Olivetti have formed a joint company in Britain to promote European sales of the AT&T-developed computer operating software called Unix.

The company is the first to be formally set up under the recent co-operative agreements between AT&T and Olivetti.

Unix, developed by AT&T's Bell Laboratories, has been highly successful since its introduction in the mid-1970s.

Worldwide, some 90,000 computers from 80 manufacturers are using the software, which has the advantage that many users can access several different computing tasks at the same time.

Readers' comments

C is the worst

To the Editor: I could not disagree more with your article on the C programming language ("The power and the portability," April 19, p. 152). Why does "i+=2" convey more information than "i=i+2"? It saves two characters, but so what—it just makes the code harder to read. C has other confusing operators, too. For instance, in some cases "&" indicates "address of"; in others, "bit-wise addition"; and in still others, "bit-wise intersection and increment."

In these days of constantly plunging memory costs, why don't we stop trying to save every last bit and start going back to the user interface and the human being it serves. Let's see programs written in English or its well-known derivative, the Basic programming language. It's readable, and compilers make it run faster than interpreters can. C is without doubt the worst language that I have ever had to work with.

The whole point of writing programs is twofold. First, we want to control the computer. Second, we want to be able to read the program back in the future, when we have forgotten it or when someone else wants to read it. We should therefore try to convey as much meaning as possible in each line of code instead of boiling the lines down to the point of unintelligibility. After all, a programmer's time is more expensive than memory.

Louis Schirm IV
DSP Systems Corp.
Anaheim, Calif.

The University of Adelaide
Department of Computer Science
GPO BOX 498, ADELAIDE, SOUTH AUSTRALIA 5001 TWX: UNIVAD AA89141

K.J. Maciunas
Department of Computer Science
Phone: (08) 228-5592

June 20th ,1984

Peter Ivanov,
Editor, AUUGN.
School of EE and CS,
University of NSW.
GPO Box 1,
Kensington NSW 2033.

Dear Peter,

I write regarding the letter by Peter Braun in AUUGN Vol 5 Nr. 3. What Mr. Braun says is correct. We have *never* had UNITY here at Adelaide. I don't know where the information printed in Vol 5. Nr. 1 came from, but I can assure you, at least on behalf of this department, it is quite incorrect. I have examined UNITY, and seen a working system at the SAIT, and must agree with Mr. Braun's opinion that it is indeed a lot better than EUNICE. I send this by physical mail rather than electronic to make sure you get it, and enclose a letter I sent to Mr. Braun stating our position.

Yours Sincerely

Kevin J. Maciunas
(kevin:uacomsci)

enclosures.

KJM/kjm

The University of Adelaide
Department of Computer Science
GPO BOX 498, ADELAIDE, SOUTH AUSTRALIA 5001 TWX: UNIVAD AA89141

K.J. Maciunas
Department of Computer Science
Phone: (08) 228-5592

Mr Peter Braun,
Executive Director ATAC,
55 Lavender Street,
Milsons Point NSW.
2061.

June 20th ,1984

Dear Mr Braun,

I write regarding the letter you sent to the Australian Unix™ Users Group newsletter concerning the "experience" we have had here with UNITY. Your letter is absolutely correct, we have *not* had UNITY here (due to lack of funding). I have been unable to find out where the information printed in the earlier issue came from, no–one here has owned up. I will be sending a letter to Peter Ivanov complaining about that "information". For your own information, it is our belief that UNITY is a far superior product over EUNICE, and if we had the money, we would have certainly purchased it. I must apologise for not noticing the offending article in the earlier issue before your letter brought it to my attention, but reading about "Unix in Adelaide" is not very interesting when you are in Adelaide and know about it already! Rest assured, I will make every effort to rectify this problem.

Yours Sincerely,

Kevin J. Maciunas.

CC: Peter Ivanov.

KJM/kjm

Netnews

I have reproduced below some of my network mail and a few "netnews" articles that I thought may be of interest to Australian UNIX users. I have deleted some of the less meaningful data generated by various mailers and news programs. No responsibility is taken for the accuracy (or lack thereof) of anything below.

From mike:food23 Thu Jun 28 10:35:02 1984
To: auugn:elecvax kevh:elec70a
Subject: Submision to netmail ?

> From peterg:csb44 Wed Jun 27 10:34:21 1984
> From pavlovs Wed Jun 27 09:15:08 1984
>
> To: peterg
> Subject: "I was a teenage body-snatcher..."
>
> It was a dark and stormy last Wednesday evening when I
> sat down at my greenly glowing terminal and typed in the
> fateful words "con mathvax".
>
> At first all seemed for the best in this best of all
> possible worlds as it made its usual stammering reply.
> Suddenly, on my screen, there appeared the symbol ">"!
> I sat stunned until realization forced its way into my
> brain and I came to accept that I was in an editor - but
> someone else's editor in someone else's account in ....
> My mind reeled and I swooned to the floor.
>
> Afterwards, when the forensic boys had taken their photographs,
> the guys from homicide asked their usual questions and the poor
> buggers in blue cleaned up the mess I sat in the D.A.'s office
> as he read me the statement of one Peter Lovibond (peterl:mathvax).
> "I was at Sydney University and had conned on to mathvax and was
> editing a file when all of a sudden I was logged off...I didn't know
> what was happening...I was so scared."
>
> Is this a regular bug in the con software/hardware?
> Can it be fixed since it is obviously a dangerous little fella?
> If not how can we protect our families and ourselves?
>
> Frightened (a.k.a. pavlovs)
>
> P.S. Could this be the work of communists? Does that
> Coombe fellow have a terminal?
>
> Very frightened (a.k.a. pavlovs)

From kre@munnari.OZ Thu Jul 5 23:44:38 1984
Date: Thu, 5-Jul-84 23:44:38 AEST
Newsgroups: aus.general
Subject: Complete list of USENET groups
Organization: Comp Sci, Melbourne Uni, Australia

I have been asked to post a complete list of USENET groups, so you have some idea what could be available. The following was taken from net.news.group (one of the groups that we do not get) last month, and indicates the groups existing at that time. (I have abbreviated the following header)

From: alb@alice.UUCP (Adam L. Buchsbaum)
Newsgroups: net.news.group
Subject: List of Active Newsgroups
Date: Fri, 15-Jun-84 03:21:41 PDT
Organization: AT&T Bell Laboratories, Murray Hill

The following is a list of currently active USENET newsgroups as of 15 June, 1984. There are three basic subcategories of netwide newsgroups: net.all, fa.all, and mod.all. net.all consists of USENET bulletin board newsgroups that are circulated around the entire net; fa.all is a set of groups that are gatewayed to USENET from the ARPANET. The fa.all groups consist mainly of digests, though there are some bulletin boards. Some of the net.all and fa.all groups are gatewayed between the networks, i.e. items submitted from the ARPA side to the digest are split up and submitted to the USENET group, while articles submitted on the USENET side are bundled up and submitted to the digest. To post to fa.all groups, send mail back along the return address found in the From line of the articles. mod.all groups are moderated. They can only be posted to by mailing to the moderator (provided after the description of the group on the list). There are other subcategories, but they are local, local to institutions, local to geographic regions, etc. and are not listed here. Please report any corrections, additions, etc. to this list to me.

Adam L. Buchsbaum
research!alb

Newsgroup	Description
net.abortion	All sorts of discussions on abortion.
net.adm.site	Automatic maintenance of the USENET directory. Currently experimental.
net.ai	Artificial intelligence.
net.analog	Analog design developments, ideas, and components.
net.announce	General announcements of interest to all. Moderated (Mark Horton -- cbosgd!announce)
net.announce.newusers	Subgroup for new users. Monthly postings, etc. Moderated (Mark Horton -- cbosgd!announce)
net.arch	Computer architecture.
net.astro	Astronomy.
net.astro.expert	Subgroup for experts in astronomy.
net.audio	High fidelity audio.
net.auto	Automobiles and automotive products and laws.

net.aviation	Aviation rules, means, and methods.
net.bicycle	Bicycles and related products and laws.
net.bio	Biology and related sciences.
net.books	Books of all genres, shapes, and sizes.
net.bugs	General bug reports and fixes.
net.bugs.2bsd	Subgroup for UNIX* version 2BSD related bugs.
net.bugs.4bsd	Subgroup for UNIX version 4BSD related bugs.
net.bugs.usg	Subgroup for USG (System III, V, etc.) bugs.
net.bugs.uucp	Subgroup for UUCP related bugs.
net.bugs.v7	Subgroup for UNIX V7 related bugs.
net.chess	Chess and computer chess.
net.cog-eng	Cognitive engineering.
net.college	College, college activities, campus life, etc.
net.columbia	The space shuttle and the STS program.
net.comics	The funnies, old and new.
net.consumers	Consumer interests, product reviews, etc.
net.cooks	Food, cooking, cookbooks, and recipes.
net.crypt	Different methods of data en/decryption.
net.cse	Computer science education.
net.cycle	Motorcycles and related products and laws.
net.dcom	Data communications hardware and software.
net.decus	DEC* User's Society newsgroup.
net.emacs	EMACS editors of different flavors.
net.eunice	The SRI Eunice system.
net.flame	For flaming on any topic.
net.followup	Followups to articles in net.general.
net.games	Games and computer games.
net.games.emp	Subgroup for Empire.
net.games.frp	Subgroup for Fantasy Role Playing games.
net.games.go	Subgroup for Go.
net.games.pbm	Subgroup for Play by Mail games.
net.games.rogue	Subgroup for Rogue.
net.games.trivia	Subgroup for trivia.
net.games.video	Subgroup for video games.
net.garden	Gardening, methods and results.
net.general	Important and timely announcements of interest to all.
net.graphics	Computer graphics, art, and animation.
net.ham-radio	Amateur Radio practices, contests, events, rules, etc.
net.info-terms	All sorts of terminals.
net.invest	Investments and the handling of money.
net.jobs	Job announcements, requests, etc.
net.jokes	Jokes and the like. May be slightly offensive.
net.jokes.d	Subgroup for discussions on the content of submissions to net.jokes.
net.kids	Children, their behavior and activities.
net.lan	Local area network hardware and software.
net.lang	Different computer languages.
net.lang.ada	Subgroup for Ada*.
net.lang.apl	Subgroup for APL.
net.lang.c	Subgroup for C.
net.lang.f77	Subgroup for FORTRAN.
net.lang.forth	Subgroup for Forth.
net.lang.lisp	Subgroup for LISP.
net.lang.mod2	Subgroup for Modula-2.
net.lang.pascal	Subgroup for Pascal.
net.lang.prolog	Subgroup for PROLOG.

net.lang.st80	Subgroup for Smalltalk 80.
net.legal	Legalities and the ethics of law.
net.lsi	Large scale integrated circuits.
net.mag	Magazine summaries, tables of contents, etc.
net.mail	Proposed new mail/network standards.
net.mail.headers	Subgroup for the ARPA header-people list.
net.mail.msggroup	Subgroup for the ARPA MsgGroup list.
net.math	Mathematical discussions and puzzles.
net.math.stat	Subgroup for statistics.
net.med	Medicine and its related products and regulations.
net.micro	Micro computers of all kinds.
net.micro.16k	Subgroup for 16k's.
net.micro.432	Subgroup for 432's.
net.micro.6809	Subgroup for 6809's.
net.micro.68k	Subgroup for 68k's.
net.micro.apple	Subgroup for Apple's.
net.micro.atari	Subgroup for Atari's.
net.micro.cbm	Subgroup for Commodore's.
net.micro.cpm	Subgroup for the CP/M operating system.
net.micro.hp	Subgroup for Hewlett/Packard's.
net.micro.pc	Subgroup for IBM personal computers.
net.micro.trs-80	Subgroup for TRS-80's.
net.micro.zx	Subgroup for zx's.
net.misc	Miscellaneous discussions too short lived for their own groups.
net.motss	Issues pertaining to homosexuality.
net.movies	Reviews and discussions of movies.
net.movies.sw	Subgroup for the Star Wars saga(s).
net.music	Music lovers' group.
net.music.classical	Subgroup for classical music.
net.net-people	Announcements, requests, pointers, etc. concerning people on the net.
net.news	Discussions of USENET itself.
net.news.adm	Subgroup for news administrators.
net.news.b	Subgroup for B news software.
net.news.config	Subgroup for posting of computer down times and network interruptions.
net.news.group	Subgroup for discussions and lists of newsgroups.
net.news.map	Subgroup for maps.
net.news.newsite	Subgroup for new site announcements.
net.news.sa	Subgroup for system administrators.
net.nlang	Natural languages, cultures, heritages, etc.
net.nlang.celts	Subgroup for Celtics.
net.nlang.greek	Subgroup for Greeks.
net.notes	Notesfile software from the University of Illinois.
net.origins	Evolution versus creationism (hot).
net.periphs	Peripheral devices.
net.pets	Pets, pet care, and household animals in general.
net.philosophy	Philosophical discussions.
net.physics	Physical laws, properties, etc.
net.poems	For the posting of poems.
net.politics	Political discussions. Could get hot.
net.puzzle	Puzzles, problems, and quizzes.
net.railroad	Real and model train fans' newsgroup.
net.rec	Recreational/participant sports.

```
        net.rec.birds           Subgroup for bird watching.
        net.rec.boat            Subgroup for boating.
        net.rec.bridge          Subgroup for bridge.
        net.rec.caves           Subgroup for caving.
        net.rec.coins           Subgroup for coin collecting.
        net.rec.disc            Subgroup for disc activities.
        net.rec.nude            Subgroup for naturalist/nudist activities.
        net.rec.photo           Subgroup for photography.
        net.rec.scuba           Subgroup for SCUBA diving.
        net.rec.ski             Subgroup for skiing.
        net.rec.skydive         Subgroup for skydiving.
        net.rec.wood            Subgroup for woodworking.
net.religion            Religious, ethical, and moral implications of actions.
        net.religion.jewish     Subgroup for Judaism.
net.research            Research and computer research.
net.roots               Genealogical matters.
net.rumor               For the posting of rumors.
net.sf-lovers           Science fiction lovers' newsgroup.
net.singles             Newsgroup for single people, their activities, etc.
net.social              Like net.singles, but for everyone.
net.sources             For the posting of software packages.
net.space               Space, space programs, space related research, etc.
net.sport               Spectator sports.
        net.sport.baseball      Subgroup for baseball.
        net.sport.football      Subgroup for football.
        net.sport.hockey        Subgroup for hockey.
        net.sport.hoops         Subgroup for basketball.
net.startrek            Star Trek, the TV show and the movies.
net.std                 All sorts of standards.
net.suicide             Suicide, its causes and effects (!).
net.taxes               Tax laws and advice.
net.test                For testing of network software.  Very boring.
net.text                Text processing.
net.travel              Traveling all over the world.
net.tv                  The boob tube, its history, and past and current shows.
        net.tv.drwho            Subgroup for Dr. Who.
        net.tv.soaps            Subgroup for soap operas.
net.unix                UNIX neophytes group.
net.unix-wizards        Discussions, bug reports, and fixes on and
                        for UNIX.  Not for the weak of heart.
net.usenix              USENIX Association events and announcements.
net.usoft               Universal (public domain) software packages.
net.veg                 Vegetarians.
net.video               Video and video components.
net.vvs                 The Vortex Video System for digitized video images.
net.wanted              Requests for things that are needed, e.g. device
                        drivers, information, etc.
net.wines               Wines and spirits.
net.wobegon             "The Prairie Home Companion" radio show.
net.women               Women's rights, discrimination, etc.
        net.women.only          Postings by women only (read by all).
net.works               Assorted workstations.
        net.works.apollo        Subgroup for Apollo's.

fa.arms-d               Arms discussion digest.
fa.arpa-bboard          ARPANET bulletin board.
```

fa.bitgraph	The BBN bitgraph terminal.
fa.digest-p	Digest-people digest.
fa.editor-p	Editor-people digest.
fa.energy	Energy programs, conservation, etc.
fa.human-nets	Computer aided communications digest.
fa.info-terms	All sorts of terminals.
fa.info-vax	DEC's VAX* line of computers.
fa.info-vlsi	Very large scale integrated circuits.
fa.laser-lovers	Laser printers, hardware and software.
fa.poli-sci	Politics and/versus science.
fa.railroad	Real and model train fans' newsgroup.
fa.sf-lovers	Science fiction lovers' digest.
fa.tcp-ip	TCP and IP network protocols.
fa.telecom	Telecommunications digest.
fa.teletext	Teletext digest.
mod.ber	Summaries of discussions from other groups. Brian Redman -- harpo!ber

---

* UNIX is a Trademark of AT&T Bell Laboratories.
* DEC is a Trademark of the Digital Equipment Corporation.
* VAX is a Trademark of the Digital Equipment Corporation.
* Ada is a Trademark of the Ada Joint Program Office of the Department of Defense.

I repeat again, if you think that we should be getting any of these, please let me know, however, I doubt if there will be any additions in the near future. Knowledge of what is wanted is still needed though, as the opportunity may arise at short notice.

Robert Elz kre:munnari

From ksh@cbosgd.UUCP Mon Jul 2 11:35:10 1984
Date: Mon, 2-Jul-84 11:35:10 AEST
Newsgroups: net.announce
Subject: Summary of June 1984 Salt Lake City Usenet BOF
Organization: AT&T Bell Laboratories, Columbus

On Thursday night of the Usenix Conference, a Birds-of-a-Feather concerning Usenet took place. Many issues were discussed and voted on. We'll summarize those issues here, and tell you which group the follow-up discussions should take place in.

First of all, the Usenet maps were discussed. These are the maps that Karen posts to net.news.map on the first of every month. After the first of July, the 'Mail' line in the maps will go away (this decision was made before the conference). This will be replaced by the UUCP map, that will be posted (hopefully) before the first of July. Since most people use these maps to route mail, it was felt that posting them quarterly instead of every month would be enough (changes only would be posted once a month). Please mail comments on this change to cbosgd!ksh, or discuss in net.news.map. I should also mention that due to my time constraints, the maps will continue to be posted once a month through the summer. By then, I should have the time to devote to getting software written to post changes only.

Second, the idea of including the date a site joined the net (in the Usenet maps) was vetoed. The idea of including the date the entry was last edited was approved. Comments to cbosgd!ksh.

The idea of removing net.general and/or net.followup. No one was opposed, but many felt that the idea should be put before the net so that those who were unable to go to Usenix could be heard. Discussion of this topic is taking place in net.news.group.

A committee to decide on new newsgroups had been proposed to the net. Discussion made it clear that almost nobody was in favor of this idea. Comments to cbosgd!mark.

Renaming some nontechnical net.all groups as pers.all. This idea was proposed two years ago and rejected, but the increased levels of traffic suggested it might be worthwhile to raise again. However, there was no support for the idea. It was instead felt that if sites wanted to cut down on the number of newsgroups, they should be able to choose for themselves which ones they want or don't want. Comments to cbosgd!mark.

Karen Summers-Horton and Mark Horton

From donn@utah-cs.UUCP Tue Jul 10 10:16:59 1984
Date: Tue, 10-Jul-84 10:16:59 AEST
Newsgroups: net.lang.f77,net.unix-wizards,net.bugs.4bsd,net.news.group
Subject: A proposal for a 4.2 BSD f77 mailing list
Organization: CS Dept., University of Utah

At the Unix Fortran birds-of-a-feather meeting that was held at the recent Salt Lake City Usenix conference, it was proposed that a mailing list for 4.2 BSD f77 sufferers be created. The idea was that this mailing list would tie together the bug reports and fixes for the 4.2 BSD f77 compiler that have been appearing in the Usenet groups net.bugs.4bsd, net.lang.f77 and net.unix-wizards (of which only unix-wizards goes to ARPAnauts). The medium of a mailing list was felt to be superior to any attempt to use the existing Usenet news software by an overwhelming plurality of the audience at the meeting. I volunteered (gulp!) to organize the list, and moderate it if that was felt to be necessary.

Before embarking on the creation of yet another mailing list, I thought I would canvass the readership of the original three groups (plus Usenet news administrators) and see what you thought about the following issues:

> Is a 4.2 BSD f77 mailing list necessary? Would a (possibly moderated) Usenet news group be acceptable to you? Should there be a Usenet group that is gatewayed to a list?
>
> Would you subscribe to the mailing list if it were created? What if it were only available as a Usenet news group? If both forms were available, which would you subscribe to, if any?
>
> Would you contribute bug reports to a list or news group? Would you contribute bug fixes to it?
>
> Should the list or group be moderated for quality control? How do you feel about the current level of quality in 4.2 BSD f77 bug reports and fixes? How long would you be willing to wait for a new fix to be approved?
>
> If you feel that the list needs to be moderated in order to ensure some sort of quality control, would you volunteer your site to run experimental versions of the compiler and libraries on production software?

If you feel up to organizing your thoughts on these questions, mail your response to me at one of the following addresses and I will summarize to the net when the dust settles down in a week or two.

Donn Seeley
Computer Science Dept.
Merrill Engineering Building
University of Utah
Salt Lake City, UT 84112
(801) 581-5668

donn@utah-cs.ARPA
decvax!harpo!utah-cs!donn
hplabs!utah-cs!donn
arizona!utah-cs!donn

From donn@utah-cs.UUCP Sat Jul 28 12:53:34 1984
Date: Sat, 28-Jul-84 12:53:34 AEST
Newsgroups: net.lang.f77,net.unix-wizards
Subject: Announcing '4bsd-f77', a way to coordinate 4.2 BSD f77 bug reports
Organization: CS Dept., University of Utah

Based on the responses to the questionnaire which was sent out nearly three weeks ago, I have decided to create a quasi-formal system for handling 4.2 BSD f77 bugs over the network.

Since a number of respondents felt strongly that a newsgroup was the appropriate medium and a nearly equal number wanted a mailing list, I will create a mailing list called '4bsd-f77' and gateway it into the existing Usenet newsgroup net.lang.f77. The mailing list part of the arrangement will be moderated -- this means that I will look at bug reports coming in to the 4bsd-f77 address and check to make sure that the reports describe a behavior of the compiler that can be duplicated with my software and don't represent a bug that is already known. Messages that are fed into net.lang.f77 will be marked as having come through 4bsd-f77, so if you want some sort of moderation in the newsgroup you can just check the subject line for the list identification. I should note here that I have no intention (or ability) to interfere with the normal traffic in net.lang.f77, so if you want to post a '4bsd-f77' article, you need to mail it to the 4bsd-f77 address just as though you were on the mailing list.

There will be a smaller mailing list 'f77-fixers' for people who want to coordinate making fixes to the compiler. At the moment the kind of subscribers I would like on this list are hackers who are willing to seriously spend a fair amount of time either processing bug reports or testing proposed bug fixes. When someone on this list develops a fix for a bug and it passes a certain amount of testing, it will go out on 4bsd-f77 and net.lang.f77. If a fix comes in on 4bsd-f77, it will be forwarded to f77-fixers for testing before it goes out on 4bsd-f77.

Does all that sound reasonable?

Anyway, here are some ways to get to '4bsd-f77':

4bsd-f77@utah-cs.arpa	decvax!utah-cs!4bsd-f77
harpo!utah-cs!4bsd-f77	hplabs!utah-cs!4bsd-f77
nbires!utah-cs!4bsd-f77	arizona!utah-cs!4bsd-f77

The list will not be active for a while, but I will take solicitations at the 4bsd-f77 address if you want to get on the distribution list when I get it working. If possible, try to give me a mail alias like parties in your area; this cuts down on mail traffic (and administrative duties!).

If you know how to get a mailing-list-Usenet gateway working and perhaps even have some software to share, I'd really like to hear from you -- send me mail at my normal address (donn@utah-cs.arpa).

Thanks to everyone who responded,
Donn Seeley donn@utah-cs.arpa (801) 581-5668 decvax!utah-cs!donn

From esa@kvvax4.UUCP Thu Jun 21 00:50:49 1984
Date: Thu, 21-Jun-84 00:50:49 AEST
Newsgroups: net.unix-wizards
Subject: Re: Generic mc68000 port
Organization: Kongsberg Vaapenfabrikk, CTG, P.O Box 25, N-3601 Kongsberg, Norway

[]
Some weeks ago I posted an article, in which I asked for any
info about a rumor of AT&T preparing the announcement of
some sort of "Generic MC68000 Unix Port".
Here's the promised summary of the answers I got.

> I don't think they have a M68000 sV.2 version yet. The version they
> have is sV.1 (or whatever number the first version has). I think
> it will be some time before they have a 2.0 version for the 68000,
> a time frame I have heared is 1Q85.
>
> Thinking of all the problems we have had to find out who we can get
> the sV/68 source from and what licences we need, the actual port
> seems to be a minor, simple, technical problem.
>
> The UNIX* System V/68000 is a System V release 1 version of the UNIX
> operating system. It runs on an EXORMACS box (68000-based).
> THe source code can be ordered from AT&T (as with any of our UNIX system
> products) and the media can be 9-track tape or EXORMACS-readable
> winchester disks. I am not sure what media Motorola intends to support.
>
> If you want specific details (e.g. price, available dates, licensing
> arrangements, etc) you should contact the sales office in N. Carolina, USA.
> The number is 1-800-828-UNIX.
>
> Motorola is also advertising the product with their specific enhancements.
> You should contact them for their product details.
>
> The port is indeed delayed. If Motorola can come up with a binary version
> that can be reconfigured for different I/O configurations WITHOUT having
> sources for the kernal etc. it will probably be out this fall. The problem
> is not insoluble as DEC has done just that with their ULTRIX binary
> licences. I would expect the price to still be in the 10K$ range but
> that's just a guess.

I'm still interested in any additional info, so if you have
anything worth adding, please, mail me or post it to the news.
There were some others, too, who were interested in the subject
and had found it awfully painful to get any information about
it.

AT&T sales people, Motorola people? Are you listening?
--

Esa K Viitala {decvax,philabs}!mcvax!kvport!kvvax4!esa
A/S Kongsberg Vaapenfabrikk,CTG4,P.O.Box 25,N-3601 Kongsberg,Norway
tel. 473-739644 tlx. 71491 vaapn n

From HFISCHER@USC-ECLB.ARPA Fri Jun 22 02:28:34 1984
Date: Fri, 22-Jun-84 02:28:34 AEST
Newsgroups: net.unix-wizards
Subject: PC/IX and Xenix Reviews

From: Herm Fischer <HFISCHER@USC-ECLB.ARPA>

In preparing for Ada environments at my company, I am attempting to employ distributed Unix systems to offload the ever-increasing demand for computing resources on our central computers. There are lots of PC's and XT's around, but they are mostly used for stand alone work, with the bulk of our engineering acitivities concentrated on hosts. These hosts are loading up to alarming proportions, and a way to offload host jobs would certainly improve response time. Ada tools will further increase our demands for these resources.

I have anxiously awaited (since January) for my own copies of PC Unix to begin to attempt to port tools and tasks from the hosts, and to determine how well they perform. (Readers in March remember my astounding observations of speed for Prolog benchmarks.)

I have now had enough time on two different PC Unix implementations, IBM's PC/IX, and Santa Cruz's Xenix, to comment on what a PC (XT) is like with Unix. The PC is no longer a toy! It's Unix performance is sufficient, when compared to loaded hosts, to make it's use worthwhile. It's ability to integrate, through Unix communications facilities like UUCP, UUX, and mailers, with networked Unix hosts, is really superb! But, don't expect miracles (yet).

If you don't want to read on, I can summarize by saying that both PC/IX, and Xenix will eventually be outstanding products. IBM's has more polish (and fewer bugs) now, and the potential for a really neat multi-windowed forms-driven slick user interface. Xenix's is much more Berkeley-ish, with tools familiar to the most die-hard VI and c-shell enthusiasts. Each system is faster than the other at some things and aggravating at others. Xenix's support crew definately wins out on customer responsiveness (sorry, IBM), by a mile, because "no" and "we will consider it" are not in their vocabulary. Xenix is, however, substantially more expensive than IBM, due basically to a warranty which makes you pay for updates and future releases (free from IBM for two years). Furthermore their license agreement is so restrictive (compared to IBM's) that my next net-note may be from the county jail.

Unpacking the Boxes and Installing Unix

Both systems come in cartons which seem to weigh as much as a case of wine. Both products come with excellent installation instructions, and lots and lots of floppies. Xenix takes over an elapsed hour to install and PC/IX installs in half the time (I've done each several times). You are in for a tremendous reading assignment if you do not know Unix setup and UUCP setup before you open the box.

Manuals

Both products basically word-processed the Bell documentation for System III Unix. The differences are that IBM probably could afford to invest more in adding helpful sentences here and there. I hate to say, but you need IBM's documentation in some places and Xenix's documentation in others, for unless you are a Unix expert, each system preserves holes in Bell documentation in different areas.

Xenix's typeface is harder to read than IBM's. But IBM let the printing out to a printer whose offset ink smudges under sweaty or greasy fingers. Nobody is perfect...

UUCP Communications

Bringing up UUCP is usually reserved for a cult of Unix experts who reserve the right to never divulge how the logon files work. IBM's documentation in this area fills in more holes than Xenix's, but it sure takes lots and lots of (and even more lots of) recursive reading and patience.

IBM includes autodialer (ACU) code for Hayes, Ventel, and DEC modems, and it works exactly as described (you just have a heck of a time becomming a UUCP logon cult member). I am using a Qubie modem, and the Hayes code drives it well. As of last weekend I still did not succeed with Xenix's UUCP (which only supports Hayes dialers).

IBM documents how to write your own dialer code (with Rixon as the example) and the Xenix folks promise to do likewise soon.

KERMIT

Since I have not yet wrung the last setup error out of Xenix UUCP, I have been using Kermit when using Xenix. Kermit.c, as distributed, with the FIONREAD stuff disabled, works right on the first attempt. However, Kermit.c will not work right under PC/IX. I spent several hours converting the Berkeley stty and gtty calls to the newer IOCTLs, and it still gets hung. With UUCP under PC/IX, I have been too satisfied to finish debugging Kermit yet on it.

Is IBM Trying to Announce Unix for the 370???

The connect program (which both systems have), operates similarly to Kermit, in that once speaking to a remote site, you must enter a special escape sequence to get back to the local machine. For PC/IX, the strange escaping sequence, control VM (actually ^Vu^M) leads me to think somebody slipped their tongue on a product whose letters start with VM and end with UNIX. I always loved rumours...

Performance

Either system, even on as small as a 256K properly configured machine, is better (at simple things) than anybody's medium loaded VAX.

PC/IX has a snappier shell response than Xenix. Xenix's C code runs faster than PC/IX's. Basically, PC/IX is two- to four- fold faster than Xenix at shell execution (loading things, getting the editor up, etc.) and editing, for an identical configuration. (This is without making any tasks "sticky" in the swap areas.) But once in execution, the "C" code generated by Xenix is faster. (For example, the UNSW Prolog interpreter runs a standard test at 192 logical inferences per second on Xenix, while only getting 182 per second on PC/IX.)

Limitations

PC/IX has a 64K size restriction on programs. Xenix doesn't limit code size (medium model).

PC/IX limits data structures in "C" to 4K bytes. Xenix doesn't. But both limit data segment size to 64K.

The PC/IX restrictions make no sense, especially if one plans to support Ada compilers and programs. The silly 4K restriction is also a problem in porting existing "C" code from PDP/11's and from Xenix systems.

Porting an Application

I ported a very complex specialized "menu generating editor" program for a DoD terminal product, from the PDP 11 to PC/IX. (It ran into a compiler bug on Xenix.) (I also ran into a PC/IX compiler bug, but it was an easy one to work around.)

Since this application only runs on PC/IX, this paragraph only pertains to PC/IX. The "C" routines which write on the CRT, in the printf family, are slower than molasses (140 to 150 characters per second on the console screen). I replaced them with routines to do direct screen memory writes (about ten times faster). These routines, "ibmcur" and "ibmprt" will be submitted to the INFO-IBMPC lending library of public domain software in a separate message. (These routines are most interesting, because they show the casual hacker how to embed assembler code in his "C" programs, and how to access memory outside of the 64K allocated data segment.)

(Perhaps the slow speed of screen writing is due the the enormous flexibility the PC/IX console handler has. An Interactive employee called the handler a "brain-damaged" ANSI emulation. I disagree; you can set colors, erase in fields, selectively scroll, and even run most vt100 code. But to do this within the vertical retrace period of the color display slows writing speed down!)

Editors

PC/IX is Interactive-ish in flavor, with a distant cousin of the Rand Editor; Xenix supports a version of the VI editor.

PX/IX has a quarterplane style editor, with full keyboard integration, online help, pop-up menu's, pop-up windows, fill-in forms (structured files), multiple windows, and more to come. (They tell me at Interactive that a product soon to be available on the VAX, "ten-plus", may be distributed by IBM for the PC. I have seen this product and eagerly await it. I saw a developer editing his "C" code, popping up a menu and selecting compile, and after a pause, seeing the error messages in pop-up boxes pointing to the respective offending syntactical construct. It sure makes Unix more user friendly, even if the editor now becomes a pseudo shell. I'd love to have this for Ada!)

The Xenix VI is honest VI. A mode-sensitive non-quarterplane editor, but well loved by many users. (I am partial to EMACS. Sorry, Xenix.)

The Xenix VI editor does not work with my (Logitech) mouse; it presently only uses standard VI commands ("hjkl" for cursor movement). The PC/IX editor supports the mouse, though it crawls slower on screen then under my hand.

If you want windows and menus under Xenix, it has v-shell; however, this program crashed the system several times for me. V-shell is basically the same as R.F. Starr's "UTIL" freeware program under PCDOS.

Impressing Your Friends with Dial-Up Ports

Both Xenix and PC/IX allow you to enable your modems on the asynchronous ports and have up to two dial-up remote users. I leave my system on at night in the office, and call it from a terminal at home. Unless your remote users try to do "makes" or otherwise pig-out the feeble 8088 CPU, response time is not much worse than with a loaded VAX. PC/IX even distributes accounting software, with instructions on how to charge for usage like a big host.

A **MAJOR** PC/IX annoyance is that the editor, with all of its wonderous user friendliness, is not runnable remotely. When I go home and call up my PC with PC/IX, I must grovel and use the yucky ED editor. All (with no exception yet found) other PC/IX programs seem to work properly at remote terminals. I know that this editor is written for VAXes and 11's to work with regular remote CRT's. (The PC/IX version obviously does direct console screen-writes, and will not work remotely; even if IBM were to have to charge extra to have the editor work remotely, it would be worth it.) VI works with all terminals, local and remote, on Xenix.

I do not recommend planning to have multiple online users doing any CPU-intensive work; both systems are really only as fast as the 8088.

Minimal Systems

Both systems run decently on a "properly configured" 256K machine. But they degrade differently. PC/IX seems to run at the same speed (as a large memory configuration), when executing a program or the editor, and just slow down when loading programs, starting print spool activity, and the like. If you let cron do its "sync" every minute (as distributed) the 256K program will stop while the swapping occurs (2 seconds or so). That is disconcerting while editing, but easily remedied. Xenix, unless you disable the remote ports, will crawl in 256K. (15 minutes to do a 1 1/2 minute compile, for example.) With ports disabled, it's performance is nearly the same as with more memory; and, when swapping activity occurs, Xenix is more graceful, becomming sluggish at the VI cursor but not stopping you entirely as PC/IX does.

Jailtime

The license agreement for Xenix restricts Unix use to a single machine. An expensive product, and you cannot even use it at home in the evening and upstairs at work when your secretary is fancy-fonting under PC-DOS. You cannot even loan it to a friend for him to try on his PC on the weekend. IBM wording is far more lenient; you can only use it on one machine at a given time, and they do not even ask you to specify the serial number.

Conclusion

Unix for the PC is here. Two honest Bell System III ports have been reviewed, and both perform surprisingly well. Idiosyncrasies are different for each system. The smaller company is more responsive; the larger has more user friendly software. For every strength you will find a peeve. Time will straighten things out. Competition is wonderful (especially for software products).

From wunder@wdll.UUCP Wed Jun 27 04:25:39 1984
Date: Wed, 27-Jun-84 04:25:39 AEST
Newsgroups: net.unix-wizards
Subject: Re: VMS/UNIX mail transfer - (nf)

#R:elecvax:-24600:wdll:17100002:000:2296
wdll!wunder Jun 20 10:26:00 1984

For transfers between Unix, VAX/VMS, RSX, etc. I reccomend KERMIT.
It is a reasonably fast, and stable (doesn't hang). Recent versions
have a server mode, so that you can call up and do business with your
KERMIT, just like uucp (well, almost). We get KERMIT file transfers
at ~6000 bits/sec over a 9600 baud line, compared to FTP/TCP/IP/DDCMP
file transfers at ~8000 bits/sec over 9600. For more info about KERMIT,
get in touch with Frank da Cruz at Columbia University computer center.

It is worth bringing up KERMIT. It is good stuff, and available for
anything from an Apple II to a CRAY X-MP.

Here's the most recent scoop on getting KERMIT:

> Subject: New Kermits, How To Get Them
> To: Info-Kermit
>
> This issue of the Info-Kermit digest contains announcements of several new
> KERMITs. For the benefit of those who are new to the Info-Kermit list,
> and to refresh the memories of those who aren't, here's how to get the
> KERMIT files:
>
> ARPANET/Internet
> Use FTP, connect to host COLUMBIA-20, login as user ANONYMOUS with any
> non-null password, and GET (or MULTIPLE GET) the files you're interested
> in. Anonymous FTP access to COLUMBIA-20 is allowed only after 6:00pm and
> before 6:00am.
>
> CCNET (DECNET hosts at Columbia, CMU, CWRU, NYU, Stevens, and (soon) Vassar):
> Use NFT to COPY the desired files from CU20B::KER:. If you're on a VAX,
> just use the COPY command. Specify /USER:ANONYMOUS. If CU20B does not
> grant you anonymous access, ask your system manager to get the files.
>
> BITNET:
> On an IBM VM/CMS system, type the command SMSG RSCS MSG CUVMA KERMSRV HELP
> to get instructions for how to use the Kermit server at host CUVMA. There
> is usually a delay between the announcement of a new version of Kermit and
> its availability on BITNET, because each file has to be painfully screened
> and possibly processed to fit the requirements of our RJE link.
>
> Other networks like USENET, MAILNET, etc, or if you're not on a network:
>
> Send mail to Info-Kermit-Request@COLUMBIA-20, or to:
>
> KERMIT Distribution
> Columbia University Center for Computing Activities
> 612 West 115th Street
> New York, N.Y. 10025

From mats@dual.UUCP Thu Jun 21 15:49:54 1984
Date: Thu, 21-Jun-84 15:49:54 AEST
Newsgroups: net.unix-wizards
Subject: Re: Generic mc68000 port
Organization: Dual Systems, Berkeley, CA

My information is gathered from numerous conversations with AT&T people and Motorola people, especially Bill Lowery. However, the story changes almost weekly, so this may not be the latest version.

Motorola did a port to the 68000 of AT&T UNIX System V, Release 1. This port was *NOT* intended to be an enhancement of the standard AT&T UNIX System (unlike what many porting houses try to provide: UniSoft, MicroSoft, HCR, etc). It was specifically intended to be an extremely vanilla port. Prior to beginning the port, Motorola was handed a document describing acceptance criteria for AT&T. They did the port, and in due time had it validated and accepted by AT&T. Most of the rumors flying around pertain to the gray area of the time between the port being up and running and AT&T validating it.

Motrola will send you source code for the port NOW for $2000 (plus media cost). The catch is that you must have an AT&T System V source license with AT&T and it *MUST* be specified as a 68000 Version (*NOT* VAX version) source license. Those of you with a VAX source license will need to have it converted or purchase an `additional CPU' license, specified as 68000 Version. Until validation, this license was not available, therefore Motorola was unable to ship the product.

Some clarification is still necessary with respect to AT&T's position on processor-type-specific licenses, a new idea for most people. Such things as conversion details (switching from Vax to 68000, for example) are still not completely straight. It is rumored to be easy and cheap (like free). I have not gone through the procedure, so I don't know for sure.

System V, release 2 is due to be completed by Motrola 3Q or 4Q of '84. Presumably, the same shipping restrictions will apply - no valiadtion from AT&T, no product shipped. Validation consists, apparently, of checking each line of code with the original, so there are no substantial rewrites, except in very processor-specific stuff (kernel, compilers, PS, debuggers). Do not expect Sys V, R 2 before '85, in my humble opinion.

Mats Wichmann
Dual Systems Corp.
...{ucbvax,amd70,ihnp4,cbosgd,decwrl,fortune}!dual!mats

P.S. - apparently, there was a session at the recent Usenix show, where Motorola may have discussed some of these details. I did not attend, so I don't know. If someone has info from there that conflicts with or is an addition to what I have mentioned, please post comments.

From: tim@ucf-cs.UUCP (Tim Curry)
Date: Wed, 13-Jun-84 05:50:04 AEST
Newsgroups: net.jobs,net.usenix
Subject: UNIX on the SEL 32
Organization: University of Central Florida

The Gould-CSD division located in Ft. Lauderdale Florida is actively developing a Berkeley 4.2bsd port of UNIX* for their SEL 32 high performance 32 bit computer. They are interested in hiring several people to work on this project in all areas of UNIX software. For more details, you can contact Wayne McLaren at (305) 797-5924 or stop by the Gould booth at USENIX in Salt Lake City.

*UNIX is a trademark of Bell Laboratories.
--
Tim Curry
USENET: decvax!ucf-cs!tim ARPANET: tim.ucf-cs@csnet-relay

From steveg@hammer.UUCP Thu Jul 19 16:24:47 1984
Date: Thu, 19-Jul-84 16:24:47 AEST
Newsgroups: net.usenix
Subject: Re: when is next usenix confrence?
Organization: Tektronix, Wilsonville OR

The next summer USENIX conference is scheduled for Portland Oregon. June 12-14, 1985 (tutorials on the 11th).

This is a usenix and software tools only conference (no significant /usr/group participation tho they will probably have a booth).

We're still trying to settle on most of the details. We wanted to wait to see how some of the new stuff that they tried in Salt Lake worked out (did you turn in the survey form from your registration packet?)

Vendors: There will be exhibit space (approx the same size as SLC). Contact Chuck Bierly of Industrial Presentations West (303-696-6100) to reserve space [this is the same crew that did SLC exhibits].

Steve Glaser
General Chairman
USENIX/Portland 85

tektronix!steveg (UUCP)
steveg.tektronix@csnet-relay.csnet (ARPANET)

Draft 23/7/84
Proposed Rules and By-Laws for
the Australian Unix systems User Group

PREAMBLE

(a) **Sources.**
This document has been compiled by John Lions, with inputs, assistance and advice from the Interim Executive Committee consisting of: Chris Campbell, Phil Chadwick, Geoff Cole, Tom Crawley, Robert Elz, John Field, Kevin Hill, David Horsfall, Peter Ivanov, Chris Maltby, Ross Nealon, John O'Brien, Doug Richardson, and Tim Roper.

Like most draft constitutions, it owes much to other similar documents. In this case, the ones that came to hand were the earlier draft for an AUUG constitution compiled by Chris Maltby and Ron Baxter, and the constitutions of the USENIX Association, the European UNIX systems User Group, the Australian Computer Society, the Australian Association of University Staff, and the Roseville Park Tennis Club.

(b) **General Considerations.**
This constitution is needed because the informal association that has existed up until now is no longer suitable for handling significant sums of money, organising large meetings, etc.
Because the technical meetings can now draw in excess of two hundred participants, and may grow much larger if overseas experience can be used as a guide, advance commitments for renting conference halls, arranging catering, etc. involving substantial financial guarantees are now needed.
At the same time, the informality of the present arrangements in so far as the average member is concerned should be preserved.

(c) **Postal Ballots.**
It is very difficult and expensive to gather the membership of an association such as AUUG together in one place. Therefore the normal procedures used by associations for deciding important issues are unattractive; it is certainly the case that we don't want to spend valuable meeting time when people are together performing the mechanics of elections and voting on various issues.

Therefore it is important that all major issues, including the election of office-bearers should be resolved by postal ballot, in spite of the cost in postage and stationery, and in the delays that must follow.

(d) **General Meetings.**
On the other hand, there should be opportunities for members to be informed of the association's affairs first hand, to query the office-bearers on matters of concern, and to initiate discussions on matters that may subsequently be resolved by postal ballot.

It seems highly desirable that if the association continues to hold two major technical meetings per year, the Annual General Meeting should be held in conjunction with one of these (with formal reports distributed beforehand; new office-bearers already elected; so relax!). An Ordinary General Meeting should be held in conjunction with the other technical meeting.

It is however hard to frame the Rules to prescribe this without binding the association too tightly in other respects (as USENIX has already found).

(e) **Management Committee Meetings.**
The Management Committee should be able to conduct most of its business informally, so the need to convene full meetings face-to-face should not be too great (such meetings can be very expensive in terms of air fares and accommodation).
It is envisaged that the two required meetings will be held on the day before the two Technical Meetings, at the same location of course.

(f) **Membership Arrangements.**
The proposed clauses regarding commencement and termination of membership imply essentially that membership only changes on the first day of each month. This should simplify the life of the Secretary, and allow accurate lists of voting members to be compiled when needed.

(g) **Classes of Members.**
This is one point on which there has been little agreement. There is no clear understanding as to what kinds of members we want to be. The draft constitution envisages four classes of members: Student, Ordinary, Institutional and Honorary Life Members. No one will be eligible for the latter class for at least five years.
Other similar associations have found it appropriate to have Institutional members or the like: it is proposed that the class be established, but that no Institutional members be accepted until the Management Committee has clarified appropriate conditions etc. There is no a priori reason why an individual should not be able to pay the subscriptions and receive the benefits of an Institutional member if he so chooses.
Student members are proposed as receiving the Newsletter, and having no voting rights (hence saving administrative costs). Their membership fee will be set at the marginal cost of producing and distributing the newsletter.
This leaves ordinary members, who, as we all know, is us.

(h) **An Incorporated Association.**
The rules that follow have been drafted in so far as it has been possible with the view that the AUUG would become incorporated. However this point has not been reached yet ... certainly some legal advice is going to be required along the way ... something to keep the new Management Committee on its toes.

Draft 23/7/84
Proposed Rules and By-Laws for
the Australian Unix systems User Group

I. RULES

(1) The association shall be known as the **Australian Unix systems User Group**, abbreviated hereinafter to **AUUG**.
[UNIX is a trademark of A.T. & T. Bell Laboratories.]

(2) **Office of the Association.**
The office of the AUUG shall be at Room 343E, School of Electrical Engineering, University of New South Wales, Kensington, New South Wales, or at such other place as shall from time to time be determined by the Management Committee.

(3) **Definitions.**
In these rules, unless otherwise stated:
``he´´, ``him´´ and ``his´´ shall also be construed to mean ``she´´, ``her´´ and ``her´´ respectively;
``Financial year´´ means the period from 1 June to 31 May.
``By-Laws´´ shall refer to the By-Laws of the AUUG unless otherwise expressly stated.
``mail´´ shall imply the transmission of information in written or printed form, first-class pre-paid, via the general post or public or private courier service.
``unfinancial member´´ shall mean any member whose most recent term of membership has expired and who has not yet paid the subscription for the next twelve month period.
``voting member´´ shall mean any member entitled to cast a vote.

(4) **Aims.**
The aims for which the AUUG is established are to promote knowledge and understanding of the UNIX system, and of similar or related computer systems.

For the furtherance of these aims and to achieve its purposes, the AUUG may carry out any or all of the following activities: conduct general meetings, conferences, discussion groups, panels, lectures and other types of meeting; prepare and distribute a newsletter and other publications; collect software and distribute said software to its members for their use; verify licenses of members for the purposes of administering the services of the AUUG; subscribe to or cooperate with or affiliate with or amalgamate with other associations formed elsewhere with similar aims; accumulate assets; and establish and promote other activities consistent with its purpose for the benefit of its members.

(5) **Membership.**
Membership in the AUUG shall be open to all individuals or organisations who subscribe to the aims of the association, and who agree to be bound by its rules and regulations.

(6) **Application for Membership.**
An application for membership shall be in writing on the form approved by the Management Committee and shall provide such information as shall from time to time be prescribed by the Management Committee.

(7) **Commencement of Membership.**
Membership shall become current on the first day of the month following the date on which a valid membership application accompanied by payment of the appropriate entrance fee plus annual membership fee is received by the Secretary, and shall continue for twelve months from that date.

(8) **Renewal of Membership.**
Upon completion of the initial membership period and any subsequent periods, membership may be renewed for a further period of twelve months by payment of an additional annual subscription to the Treasurer.

(9) **Rights of Members.**
Each member shall be entitled to attend all meetings of the AUUG, including meetings of the Management Committee, provided any prescribed attendance fee is paid.
Each member shall receive a copy of the association's newsletter.
Each votings member shall receive notice in writing of all ballots and copies in writing of the annual reports of the Secretary, Treasurer and Auditor.

(10) **Obligations of Members.**
Each member shall abide by the Rules and By-Laws of the AUUG as they may from time to time appear. Each member shall respect licensing obligations.
Each member shall inform the Secretary of changes to his postal address.

(11) **Termination of Membership.**
A member may resign his membership at any time by giving notice in writing to the Secretary. No member who resigns shall have any claim for a refund of subscriptions paid.
A member who has been unfinancial for more than two calendar months shall be deemed to have resigned his membership, and shall no longer be entitled to any privileges enjoyed by members.
Former members who have resigned will be entitled to rejoin the AUUG on the same basis as new members joining the AUUG.

(12) **Amount of Guarantee.**
Each member of the AUUG undertakes to contribute to the assets of the AUUG in the event of its being wound up while he is a member or within one year after he ceases to be a member for payment of the debts and liabilities of the AUUG contracted before he ceases to be a member, and for the costs, charges and expenses of winding up and for the adjustment of the rights of contributories among themselves, such amount as may be required but not exceeding fifty dollars.

(13) **Expulsion of Members.**
Upon receipt of a petition so requesting from twenty or more members, or half the membership, whichever is less, the Management Committee may call upon any member to explain any alleged misconduct, and the Management Committee shall have power to suspend or expel any member

who in its opinion has either been guilty of misconduct or has acted prejudicially to the interests of the AUUG or who has wilfully infringed any of the Rules or By-Laws of the AUUG.

(14) **Annual General Meeting.**
The Annual General Meeting shall be held within the second half of each calendar year. The time and place of each Annual General Meeting shall be determined at the preceding Annual General Meeting but either the time or place or both may be changed by the Management Committee if it proves impossible or highly inconvenient to meet at the place previously selected at the time previously selected.

(15) **Ordinary General Meetings.**
A general meeting of the AUUG may be called by the Management Committee in conjunction with any Technical meeting or conference which a quarter or more of the voting members are expected to attend. The business that may be conducted at such a meeting shall be as prescribed in the By-Laws.

(16) **Extraordinary General Meetings.**
An Extraordinary General meeting shall be called by the Secretary after receipt of a petition so requesting by at least twenty voting members, or half the membership, whichever is less. The Secretary must appoint a date for the meeting no later than three calendar months after receipt of the petition, and the business of the meeting must be confined to matters described in the petition and recorded in the written agenda sent to all members by mail at least four weeks before the date set for the meeting.

(17) **Voting Rights.**
All Ordinary, Institutional and Honorary Life Members whose membership is current shall be entitled to one vote.
Any voting member may award his proxy to another voting member for the period of a single General meeting providing he so notifies the Secretary in writing at least 24 hours before the appointed time of commencement of the meeting.
The quorum for a general meeting shall be six members personally present and entitled to vote.

(18) **Office-bearers.**
The Office-bearers of the AUUG shall be: the **President**; the **Secretary**; the **Treasurer**; the **Returning Officer**; the **Assistant Returning Officer**; and the **Auditor.**

(19) **Management Committee.**
The management and control of the business and general affairs of the AUUG shall be vested in a Management Committee of seven members, namely: the President; the Secretary; the Treasurer; and four General members.

(20) **Elections.**
The election of Office-bearers and General members of the Management Committee shall be by postal ballot, under conditions defined by the By-Laws.
The term of office for all Office-bearers and Committee members except the Auditor shall be for one year, from July 1 to June 30.

(21) **Auditor.**
The Auditor shall take office after the end of the Annual General Meeting following his election and shall hold office until the end of the Annual General meeting following.
If at any time the position of Auditor becomes vacant, the Management Committee shall at its earliest opportunity appoint as auditor a certified public accountant who is not a member of the AUUG, and he shall hold office until the next annual general meeting of the AUUG.
At least once in each financial year the Auditor shall examine the accounts and financial records of the AUUG. The Auditor shall certify as to the correctness of the accounts of the AUUG and shall report thereon in writing to the Secretary and to the members at the Annual General Meeting.

(22) **Vacancies on the Management Committee.**
The position of any Committee member shall be vacated if the member fails to attend any Management Committee meeting without furnishing a satisfactory explanation as to the cause of his absence, and if the Management Committee resolves that his office be vacated.

If at any time any of the principal office bearers (President or Secretary or Treasurer) be unable to continue in office for any reason, then the Management Committee shall appoint one of their number to the vacant office.
Should a vacancy occur among the other office bearers but excluding the Auditor, or among the General members of the Management Committee, then the Management Committee shall appoint an ordinary member of the AUUG to fill the vacancy.
The Management Committee shall make the approval of such appointments an order of business for the next General Meeting of the AUUG if any such meeting will be held before the end of the association's financial year.

(23) **Management Committee Meetings.**
The Management Committee shall meet formally at least twice per year.
Notification of time, place and agenda for each meeting shall be made in writing to each member of the Committee by the Secretary at least four weeks in advance. All members of the AUUG are entitled to be present at such meetings, and may speak when invited by the Chairman, but only members of the Management Committee may vote. The quorum for such meeting shall be four. Resolutions of the committee shall require a simple majority of the members present and voting. The chairman shall have a casting vote in the event of a tie.

(24) **Distribution of Income.**
The property of the AUUG whencesoever derived shall be applied solely towards the objects of the AUUG as set out in these Rules, and no portion thereof shall be paid or transferred directly or indirectly by way of dividend to any member of the AUUG at any time.
Notwithstanding the above the AUUG may compensate the reasonable expenses actually incurred by any member in the conduct of the business of the AUUG under the direction of the Management Committee.

(25) **Chapters.**
Ten or more members of the AUUG may petition the Management Committee to form a **chapter** of the AUUG.
General rules for the organisation, operation, obligations and privileges of chapters shall be as resolved by the Management

Committee or the membership as a whole from time to time.
Each chapter shall appoint a chapter committee consisting of at least a Chapter Chairman and a Secretary/Treasurer.
The chapter committee may convene meetings consistent with the aims of the AUUG, but may not enter into any financial commitments on behalf of or in the name of the AUUG except with the written approval of the Management Committee.

(26) **Affiliation or Amalgamation with other organisations.**
The Management Committee may at any time seek or discuss the possibility of affiliation or amalgamation with any other organisation whose aims are similar to or compatible with those of the AUUG. No agreement for affiliation or amalgamation may be finalised until the matter has received the assent of two-thirds of the members voting in a postal ballot.

(27) **Dissolution of the AUUG.**
Upon receipt of a petition requesting the dissolution of the AUUG from twenty or more members, or half the membership, whichever is less, the Secretary shall within one month put the question to the membership by ballot.
If two-thirds of the members voting agree, the AUUG shall be dissolved. If upon the dissolution of the AUUG there remains after satisfaction of all its debts and liabilities any property whatsoever, the same shall not be paid to or distributed among the members or Chapters if any, but shall be given or transferred to some public educational institution, or other institution to be determined at or before the time of dissolution by resolution of the membership.

(28) **Changes to the Rules and By-Laws.**
Changes to these Rules and By-Laws may be initiated at the request of a General meeting, or by the Management Committee. All proposed changes must be approved by a two-thirds majority of the votes received in a postal ballot of the members before having effect.

(29) **Interpretation of the Constitution.**
If any doubt arises as to the proper construction or meaning of any clauses in these Rules or By-Laws, the decision of the Management Committee thereon shall be final and conclusive provided such decision be reduced to writing and recorded in the minutes of a meeting of the Management Committee.

Draft 23/7/84

Proposed Rules and By-Laws for
the Australian Unix systems User Group

II. BY-LAWS.

(30) **Classes of Membership.**
There shall be four classes of members: Ordinary members, Institutional members, Student members and Honorary Life members.

(31) **Ordinary Members.**
Any person is eligible to become an Ordinary Member.

(32) **Institutional Members.**
Any person or organisation is eligible to become an Institutional Member.

(33) **Student Members.**
Any full-time student is eligible to become a Student Member.

(34) **Honorary Life Members.**
Any person who is an ordinary member of at least five years standing and who has rendered special services to the AUUG may be elected as an Honorary Life member.

(35) **Membership Subscriptions and Fees.**
The Management Committee shall determine before the commencement of each financial year a scale of fees for entrance to the AUUG, for annual subscriptions and for the attendance at meetings, for each class of members to be applied during that financial year.

(36) **Chairman of Meetings.**
At all General meetings of the AUUG and at all meetings of the Management Committee except where otherwise provided, the Chair shall be taken by the President, or in his absence, by a member elected by the meeting.

(37) **Duties of the Secretary.**
The Secretary shall keep or cause to be kept a register of members setting forth the names and addresses in full of all members of the AUUG.
The Secretary shall furnish to the Returning Officer a complete list of all voting members whenever this is required for the conduct of a ballot.
The Secretary shall keep or cause to be kept full and correct minutes of all resolutions and proceedings at General meetings and Management Committee meetings of the AUUG.
The Secretary shall conduct correspondence on behalf of the AUUG.
The Secretary shall, during his last month of office, prepare a written report on the state of the affairs of the AUUG for distribution to the membership.

(38) **Duties of the Treasurer.**
The Treasurer shall keep or cause to be kept correct accounts and books and records showing the financial affairs of the AUUG.
The Treasurer shall notify the President and Secretary in writing of the usual location of said accounts, books and records whenever this location is changed.
The Treasurer shall receive all fees and subscriptions and all other monies on account of the AUUG and provide receipts for the same. The Treasurer shall deposit all monies received into a bank account maintained by the AUUG.
The Treasurer shall receive accounts for payment for services rendered to the AUUG, and as directed by the Management Committee arrange for payment from the AUUG's account.
The Treasurer shall, during his last month of office, prepare or cause to be prepared a written report on the financial affairs of the AUUG for approval by the Auditor and subsequent distribution to the membership.

(39) **Execution of Contracts.**
The Management Committee, except as otherwise provided in these Rules and By-Laws, may prospectively or retroactively authorise any office-bearer or member of the AUUG to enter into any contract or execute and satisfy any instrument, and any such authority may be general or confined to specific instances, except that any contract whose dollar value exceeds an amount predetermined by the Management Committee must be specifically authorised by the Management Committee.

(40) **Disbursements.**
Signing officers for the AUUG's accounts shall be the President, the Secretary, the Treasurer and one other Management Committee member chosen by the Management Committee.
All cheques, drafts, and other orders for payment of money out of the funds of the AUUG, if for less than a limit established by the Management Committee, may be signed by only one Signing officer.
For other amounts, each such instrument must be signed by at least two Signing Officers.

(41) **Conduct of General Meetings.**
Written notice of the time and place for each meeting and its agenda must be mailed to each voting member of the AUUG at least four weeks before the date of the meeting.
Business conducted at such meetings shall be confined to matters included in the written agenda, reports from office-bearers, and resolutions instructing the Management Committee to conduct a formal ballot of the membership on matters of substance. Such resolutions shall not be binding on the Management Committee unless the meeting was attended by at least twenty voting members, or half the membership, whichever is less, and the resolution was supported by at least two-thirds of the members voting.

(42) **Voting.**
All voting by the members with respect to the election of Office-bearers and Committee Members, with respect to the election of Honorary Life Members, with respect to changes to these Rules and By-Laws, and all other substantive matters shall be conducted by postal ballot.
Every voting member of record as of the date of entry of a ballot into the mails shall be entitled to one vote.

On all questions to be submitted to a ballot, the Secretary shall designate a date for the ballot to be placed in the mails, and the due date shall be four weeks after that date. The Returning Officer shall nominate the address to which voters shall post ballot papers. A ballot will not be counted if it is received after the due date or if the ballot paper does not comply with the instructions printed on it. The ballots will be received by the Returning Officer, and counted by him and the Assistant Returning Officer. The Returning Officer will report the result of the ballot in writing to the Secretary no later than two weeks after the due date.

(43) **Conduct of Elections.**
Elections are held annually for all positions of Office-bearer and Management Committee member.
Nominations for each position will be received by the Secretary up until the first day of May each year. Each nomination must be in writing, must name the position or positions sought, must be signed by at least three voting members, and must be countersigned by the nominated member who must be a financial voting member of the AUUG.
The Management Committee shall ensure that at least one valid nomination is obtained for each position. Where only one valid nomination is received for a particular position by the close of nominations, the nominee shall be declared elected forthwith, and no ballot for that position will be held.
Within one week after the first day of May, the Secretary shall advise the Returning Officer of a date for the ballot for all contested positions no later than the fifteenth day of May, shall advise him of all valid nominations received, and shall provide him with a list of voting members.
While any ordinary member may be nominated to more than one office or position, no person shall be elected to more than one position.
Ballots shall be determined in the following order: for President, for Secretary, for Treasurer, for General Committee Member, for Returning Officers, and for Auditor.

(44) **Election of Honorary Life Members.**
If before the first day of May the Secretary receives a petition from at least twenty voting members requesting the election of a member of the AUUG to the position of Honorary Life Member, then he shall arrange a ballot of the membership on this question to be conducted in conjunction with the annual election of office bearers.

Australian UNIX* systems User Group
(AUUG)

Current Subscriber Membership Application

I, _____ wish to become a founding ordinary member of the Australian UNIX systems User Group and agree to be bound by the rules of the association, as adopted or amended by the meeting to be held August 27 and 28, 1984, especially with respect to non-disclosure of confidential and restricted licensed information. I understand that, as a current subscriber to the Australian UNIX systems User Group Newsletter, I do not have to pay any membership dues for the remainder of 1984 and that, should I wish to remain a member of the association after this time, I will have to pay appropriate membership dues after January 1, 1985.

Signed _____ Date _____

==========

Name _____

Mailing address for AUUG information _____

_____

_____

Telephone number (including area code) _____

UNIX Network address _____

Name of licensed institution _____

Relevant licenses _____

==========

Office use only

* UNIX is a trademark of AT&T Bell Laboratories

Australian UNIX* systems User Group
(AUUG)

Membership Application

I, _____ do hereby apply for ordinary/student** membership of the Australian UNIX systems User Group and do agree to abide by the rules of the association especially with respect to non-disclosure of confidential and restricted licensed information. I understand that the membership fee entitles me to receive the Australian UNIX systems User Group Newsletter and I enclose payment of $_____ herewith.

Signed _____ Date _____

==========

Name _____

Mailing address for AUUG information _____

_____

_____

Telephone number (including area code) _____

UNIX Network address _____

Name of licensed institution _____

Relevant licenses _____

==========

Student Member Certification

I certify that _____ is a full-time student at _____

Expected date of graduation _____

Faculty signature _____ Date _____

==========

Office use only

* UNIX is a trademark of AT&T Bell Laboratories
** delete one

Australian UNIX* systems User Group Newsletter
(AUUGN)

Subscription Application

I wish to subscribe to the Australian UNIX systems User Group Newsletter and enclose payment of $_____ herewith for the items indicated below.

Signed _____ Date _____

==========

☐	One years subscription (6 issues) available on microfiche or paper	$30.00
☐	Back issues of Volume 1 (6 issues) available only on microfiche	$24.00
☐	Back issues of Volume 2 (6 issues) available only on microfiche	$24.00
☐	Back issues of Volume 3 (6 issues) available only on microfiche	$24.00
☐	Back issues of Volume 4 (6 issues) available on microfiche, some paper copies	$24.00
☐	Back issues of Volume 5 (6 issues) available on microfiche or paper	$24.00
☐	Subscribers outside Australia must add an extra to cover surface mail costs	$10.00
☐	Subscribers outside Australia must add an extra to cover air mail costs	$30.00

Name _____

Mailing address _____

_____

_____

Telephone number (including area code) _____

UNIX Network address _____

* UNIX is a trademark of AT&T Bell Laboratories

AUSTRALIAN UNIX-SYSTEMS USERS' GROUP WINTER MEETING 1984

You are invited to attend the 1984 winter meeting of the Australian Unix-systems Users' Group to be held in the Department of Computer Science, at the University of Melbourne. Cost will be $30 for advance registrants, $50 on site.

Dates, Hours, and Location

The meeting will be held on Monday August 27, 1984, and Tuesday August 28, 1984. On Monday, it will commence at 10:30 am (with registration commencing at 9:30) and adjourn at approximately 6 pm. On Tuesday the meeting will resume at 9 am and conclude at 4 pm.

The meeting will be held in Theatre A on the ground floor of the Richard Berry Building, which is located at the eastern side of the University (Swanston Street side). See the attached map for more details. Registration will be on the ground floor of that building, follow the signs when you arrive.

Exhibition

There will be an exhibition of Unix related computer hardware (and software) held in conjunction with the meeting. It will also be sited in the Richard Berry Building, and will be open from 10am until 5pm on Monday, and 9am until 3:30 pm on Tuesday.

Conference Dinner.

On Monday night (August 27) there will be a dinner for conference attendees. This will be held at the Comedy Cafe 177 Brunswick Street, Fitzroy (a short bus trip from the Uni). It will commence at 7:30 pm and conclude when we are thrown out. Places at this dinner are limited, acceptance will be strictly on a first come basis. No reservations will be available at the conference itself, you must book a place using the attached form. Guest tickets will be available, but limited to one guest per registrant. Cost for the dinner is $30, which includes meal, entertainment, and wines. The Comedy Cafe has a BYO liquor licence, so if your tastes run to stronger beverages, or exotic wines, you are welcome to bring your own (but there will be no discounts, sorry).

Meals and Refreshments.

Coffee, fruit juice, (etc) will be available in the Richard Berry Building throughout the meeting. There are a number of excellent restaurants and also a number of cheap restaurants (the sets are intersecting) in Carlton, only three minutes from the Uni. Lunches will not be organized, so you will be free to sample these restaurants. We will provide some guidance, and assistance with reservations where needed. More information will be available at the meeting.

Accommodation.

We have reserved rooms in the University Colleges for meeting attendees. Again, places are limited, so be sure to reserve yours early, using the attached form. There are also a number of excellent hotels and motels within walking distance of the University. Hotels in the city would be about 20 to 30 minutes walk, but the tram service from the city to the University is quite fast.

Transport

The University is located just north of the city centre, and transport from the city is available on trams. The best service to use is that in Swanston St. Catch a number 1 tram going north (destination board should show "East Coburg") or a number 15 north ("Moreland"), or anything which claims to be going to the university. Cost is $0.60.

From the airport, you can take a taxi, at a cost of about $15, or catch a coach outside the terminal for $2.40 (I think). The coach service can drop you at various city locations (including major hotels), I recommend alighting at the Ansett city terminal building, from which you can easily catch a tram to the university. Phone numbers for taxi services include 62 0331 (Combined Services), 345 3455 (Silver Top), and 347 5511 (Astoria).

Driving is not recommended, parking space in the vicinity of the university is extremely limited.

For more information, contact

Robert Elz,
Dept of Computer Science,
University of Melbourne,
Melbourne, Vic., Australia.
Phone: (03) 341 5225, International +61 3 341 5225

or

Prue Downie
(same address)
Phone: (03) 341 5232

* Unix is a trademark of AT&T Bell Laboratories.

If you will be attending the meeting, and desire to register in advance (and obtain the consequent saving of $20) return this form, with full payment, to reach us by August 10 to:

Australian Unix Users Group Meeting,
Department of Computer Science,
University of Melbourne,
Gratten St,
Parkville,
Victoria. 3052

Make cheques payable to "Australian Unix Users' Group".

Please photo-copy this form as necessary. Please type or print clearly!

Name: _____

Affiliation: _____

Address: _____

_____

_____

Phone: _____

Net address (if available): _____

Meeting Registration (pre-registration fee $30) $30.00

Conference Dinner (conference registrant, $30) ..... _____

(guest, $30, max of 1 guest) ..... _____

Accommodation ($25.00 per night in Uni College) Sun Aug 26 ..... _____

(Single rooms) Mon Aug 27 ..... _____

(Bed & Breakfast) Tue Aug 28 ..... _____

Total: ..... _____

If you have requested accommodation, we will return details of accommodation reserved as soon as possible. Refunds of amounts paid for unavailable accommodation or for unavailable places at the conference dinner will be made soon after the conference. Due to manpower limitations, refunds will not be available at or before the conference.