AUUGN

Australian Unix
User Group Newsletter

Volume 5
Number 1

The Australian UNIX* Users Group Newsletter

Volume V Number 1

CONTENTS

* UNIX is a trademark of Bell Telephone Laboratories.

Editorial

Do you have the feeling you have missed something in the last few years? Have your doors been blowing closed more often because you cannot find those old AUUGN issues? Well, have no fear, the editor who brought you volumes two and three has returned to bring you volume five.

I would like to take this opportunity to apologise to all our regular subscribers for the problems they may have experienced in dealing with the editors of AUUGN during 1982 and 1983. Particular apologies must go to subscribers like the CSIRO, who paid for volume IV and received nothing.

Well, enough of problems, on to the good stuff!

In This Issue

You will notice the appearance of a few new departments in this volume.

Books — I have written to various publishing houses explaining that the AUUGN is read by a lot of people interested in UNIX, UNIX-like systems and the C programming language, and that I would like catalogs of UNIX/C titles and short reviews for inclusion in future issues. A few companies have already replied, sending catalogs and books. For this first segment I have compiled a list of titles and a short review of "The UNIX Programming Environment" by Kernighan and Pike. Further titles and reviews will follow. If you want to review a book, drop me a line. You may get a free book out of it.

Nets — The "UNIX/UUCP network" (for want of a better name) is now world wide. Read this section to find out how you can get on to it and how you can use it.

Netnews — A selection from my network mail and certain netnews categories.

Clippings — A gathering of press releases and clippings. If you see one, send it to me.

Survey Responses

The database maintained by the AUUG contains four item types:

address — The "address" item contains information on individual people, companies or institutions including name, address, telephone number, network mailing address, newsletter payment status, plus other flags and keys. This item may also contain multiple pointers to "site" items.

Site — One of these items is allocated to every known computer running UNIX or a similar system. Information about a "site" includes the computer's network name, machine type, memory capacity, peripheral configuration and software license status. This item contains pointers to "siteaddr" and "software" items as well as pointers to the "address" items of the "software_contact", "hardware_contact" and "other" users.

Siteaddr The "siteaddr" item contains the name, address and phone number of the computer site and pointers to "site"s at that address.

Software Software items may be flagged as available or wanted. The item contains the name and a description of the package, and pointers to the "address" items of "users" and "suppliers". Each software item also has a "type" associated with it, used to class the package into one of about 30 categories.

I have received 93 completed surveys and all of the "address" and "site" information has been entered into the database. Entry of "software" information is naturally slower, but work is continuing.

New Newsletters

Two new newsletters have appeared on the UNIX scene. The first is called "/USER - The Australian UNIX and C Journal (Plus C on Micros)", a fairly unpretentious title. It is published by Structured Language Resources and appears to be targeted at the UNIX novice. I have written a letter to the editor but have received no reply. Subscription rates are the same as AUUGN's, but issues are much smaller.

The second is published by the New Zealand UNIX User Group and is, as yet, limited to a single sheet of paper. Things should improve when they get their act together and I look forward to hearing more from them (see letter later in this issue).

Next AUUG Meeting

You should all know by now, but for the record, the next AUUG meeting will be held on February 20 and 21, 1984 on the campus of the University of Sydney. All major manufacturers of UNIX based systems will be represented providing an excellent opportunity for current and prospective users to evaluate equipment.

The first day will be devoted to general interest non technical papers and the second day will be devoted to technical presentations on UNIX applications and system development. If you wish to receive more information, register or indicate interest in tutorials, further information is reproduced later in this issue.

I have said it before and I now say it again. The AUUG must become a formal association, club, company or whatever, and it must do so soon. Consideration of such action, along with the consideration of a draft constitution is on the agenda for the next meeting. The draft constitution will be available at the February meeting and I urge you to read it and express your views at the meeting.

Contributions

No issue of AUUGN would be complete without the usual request for your contributions. If you do something, almost anything within reason, write it up and send it to me. I also have a few books to give away to the best contributor to each issue.

Books

I have used information supplied by publishers and from other sources to compile the list of titles below. Should you know of any I have not listed or have listed incorrectly, or should you want to review a title, please contact me.

Books on UNIX

1. Using the UNIX System
   Richard Gauthier
   Reston Publishing Co.(1981)

2. A User Guide to the UNIX System
   Rebecca Thomas, PhD and Jean Yates
   OSBOURNE/McGraw-Hill (1982)

3. UNIX - The Book
   M.F. Banahan and A. Rutter
   Sigma Technical Press (Wiley)(1982)

4. A UNIX Primer
   Ann Nicols Lomuto and Nico Lomuto
   Prentice-Hall (1983)

5. UNIX Primer Plus
   Mitchell Waite, Donald Martin and Stephen Prata
   Howard W. Sams and Co. Inc. (1983)

6. The UNIX System
   Stephen R. Bourne
   Addison-Wesley (1983)

7. Concurrent Euclid, UNIX and TUNIS
   R.C. Holt, University of Toronto
   Addison-Wesley (1983)

8. The UNIX Operating System
   Kaare Christian
   John Wiley and Sons, Inc (1983)

9. The UNIX Programming Environment
   Brian W. Kernighan and Rob Pike
   Prentice-Hall (Software Series)(1984)

10. Starting with UNIX
    P.J. Brown
    Addison-Wesley (1983)

11. The UNIX Programmer's Manuals
    Bell Laboratories
    CBS Educational and Professional Publishing, NY

12. Introducing the UNIX System
    Henry McGilton and Rachel Morgan
    McGraw-Hill (1983)

13. The UNIX System Book
    P.P. Silvester
    Springer-Verlag (1983 tentative)

14. Understanding UNIX, A Conceptual Guide
    James R. Groff and Paul N. Weinberg
    Que Corp (1983)

Books on C

1. The C Programming Language
   Brian W. Kernighan and Dennis M. Ritchie
   Prentice-Hall (Software Series)(1978)

2. C Notes - A Guide to the C Programming Language
   C.T. Zahn
   Yourdon Press (1979)

3. Learning to Program in C
   Thomas Plum
   Plum Hall (1983)

4. The C Puzzle Book
   Alan Feuer
   Prentice-Hall (1982)

5. The C Primer
   Les Hancock and Morris Krieger
   McGraw-Hill (1982)

6. C Programming Guide
   Jack Purdum
   Que Corp. (1983)

Related Books

1. Software Tools
   Brian W. Kernighan and P.J. Plauger
   Addison-Wesley (1976)

2. Software Tools in Pascal
   Brian W. Kernighan and P.J. Plauger
   Addison-Wesley (1981)

3. Text Processing with UNIX
   nroff, troff and eqn
   D.W. Barron and M.J. Rees
   Addison-Wesley (1983)

4. The Bell System Technical Journal Vol57 NO6 Part2
   Edition on "UNIX Time-Sharing System" July-August 1978
   American Telephone and Telegraph Company

In future issues we will present reviews of these books and others of interest to our readers. In this issue I have reprinted a review by Mike O'Dell of "The UNIX Programming Environment" (Kernighan and Pike).

A Review of
"The UNIX Programming Environment"
by Brian Kernighan and Rob Pike
Prentice-Hall Software Series

Reviewed by
Mike O'Dell
Center for Seismic Studies, Arlington, VA
mo@lbl-csam, lbl-csam!mo, seismo!mo

"The UNIX Programming Environment" is a book long overdue and badly needed. In this one place, you can find out "How to Write Good" in the Unix world. I recently saw a Letter-to-the-Editor of some trade rag effusing over how the Pick OS is better than Unix because in Pick you never have to write a program, while in Unix you always have to write a program (distinct implication was "in C"). If there ever was a rebuttal to that miscomprehension, this book is it.

This book spans an amazing range of topics: it contains a VERY good upgrade of the "Unix for Beginners" chapter, and then proceeds to demonstrate just how much work can be done on Unix, even if the C compiler has been removed from the system. The treatment of shell programming (Bourne shell) is gradual and impressive. It shows the sheer power and elegance of Unix, while at the same time, not being coy about pointing out the occasional warts. From simple one-liners to embrionic SCCS-type tools, the reader learns the skills necessary to truly exploit the powerful programming language that is the shell. After mastering the shell, the discourse moves to things which really need C, and how to write them well. Design issues, coding hints, good practices, there is a wealth of information here. Starting with "cat" (Hello, Toronto!) which does all (only?!) the right things without needing flags, to a full-fledged interpreted programming language, here is a course in software engineering using power tools, instead of hammer-and-tongs. All along the way, things are built gradually, reusing previous pieces instead of starting from a blank sheet of paper. This book should be read at the terminal, doing the exercises right along.

The reviewer has been thinking a great deal about the audience of this book. It is purest platinum for the programmer who while competent, is new to Unix and the Unix world view. Unix is a powerful environment and learning to use it effectively and harmoniously can sometimes go awry directly because the power and flexibility often make it almost as easy to do things wrong as right. Moreover, most programmers, even and especially many practicing Unix programmers, tend to think of solving problems by writing programs. This is to be encouraged; the problem, however, is people often start writing in the wrong language (frequently "C"). There are in fact many jobs which need programs to do them, but if your view of "how to get the job done" isn't altered by this text either you are already Genuinely Enlightened or you have probably missed the point. (There are indeed subtle points along the way.)

While the book is well written and an interesting philosophical discourse, there are points the reviewer differs with, and no doubt, any reader will occasionally disagree with the authors. But that too is valuable. Thoughtful introspection is good for the soul, and gets one away from the terminal as well.

Finally, I heartily commend this book to anyone wanting to Truly Know Unix; it ain't perfect, but it is enlightening. I particularly recommend it to persons responsible for influencing the direction of the system outside the BTL research group. If you ever wondered about the "intent", you will probably never see it spelled out as well anywhere else. Finally, anyone attempting to "enhance" Unix for whatever reason or application should read this book. Before you can enhance it, you have to know what is REALLY there and what is NOT broken.

Annotated Top-level Table of Contents:

1. UNIX for Beginners
   Well re-written; deals well with the issue
   of local system conventions for erase, kill, etc.

2. The File System
   What it does and doesn't do, and why that is important.

3. Using the Shell
   Intro to Metacharacters, I/O redirection, pipes, etc.

4. Filters
   grep, sed, awk and company

5. Shell Programming
   Production-quality shell scripts

6. Programming with Standard I/O
   Intro to the Standard I/O library ("C" obviously)

7. UNIX System Calls
   Low-level I/O, processes, signals, etc.

8. Program Development
   From a four-function calculator to
   an almost-Basic in 6 Easy Lessons

9. Document Preparation
   Meet -ms and troff

10. Epilog

Appendix 1: ed summary

Appendix 2: "hoc" manual page

Appendix 3: "hoc" source code listing

Nets

Here I sit, knowing the information I wish to impart, but rather awestruck by the job ahead. I feel like a needle trying to describe a haystack. Well, here goes.

First, lets talk about the "Australian UNIX Network" or AUNET. AUNET uses the "Sydney UNIX Network" (SUN) software, which in turn has its origins at the Universities of Sydney and New South Wales. AUNET may grow to be replaced by the "Australian Computer Science Network" (see clippings) but that is another story and will be covered in a later issue of AUUGN. A reasonably current list of AUNET sites is reproduced later in this issue.

How do you get your site onto AUNET? There were two questions in the recent survey I sent out relating to this subject. Most respondents said that they would forward network traffic to other sites, and those people not connected to the network expressed strong wishes to be so connected. The simplest way to get onto the net is to contact a nearby site that is already connected. Alternately, Victorian readers may contact Robert Elz at the Department of Computer Science, University of Melbourne, Parkville 3052 or phone (03) 341 5225. People in other states may contact Bob Kummerfeld or Piers Lauder at the Basser Department of Computer Science, Sydney University, Sydney 2006. Their telephone numbers are (02) 692 3766 and (02) 692 2824 (resp).

What is it likely to cost? This depends on where you are and what sort of AUNET connection you want. Most of the larger sites are connected via leased Telecom lines and this connection method provides mail/file transfer and remote login facilities. Dialup or csironet connection provides mail/file transfer only. I am willing to liaise between parties in particular geographical areas who would like to club together to lease and share a 1200 baud full duplex datel line. If you envisage making heavy use of the network then a leased line is for you. On the other hand, small out of the way sites should consider dialup, csironet or (later) AUSTPAC.

As an example, the yearly cost of a datel line from Canberra to Sydney, including modems, would be about $6320, plus $840 once only installation charges. Sharing these amounts among several sites could reduce the yearly expense to under $1000, an amount within the budgets of most institutions.

There are two other UNIX computer networks of interest to us in Australia. AUNET has links to the North American network USENET which in turn has links to EUNET in Europe. Sites on these networks exchange electronic mail and network news over mostly ad-hoc communications channels mainly using dial-up lines or leased lines. At this point you should read "The European UNIX Network", reprinted later in this issue.

Now comes the difficult part. Both USENET and EUNET use "uucp" and associated grislies as the mail/news transport medium. This means that remote login over the network is not possible and mail cannot be routed automatically to a specific user on a specific machine.

Consider machines "a", "b" and "c", and also assume that the network links "a" to "b" and "b" to "c", but not "a" to "c". On EUNET and USENET, mail from a user on machine "a" to a friend on machine "c" would have to be addressed to "a!b!c!friend". On AUNET the mail could simply be sent to

"friend:c". Now, consider that there are more than 1600 machines on EUNET/USENET with probably more than 5000 different "machine!machine" connections. What hope have we here in Australia?

Well some. I have set up a network-site/mail-connection/news-connection database on "elecvax" (at the School of EE and CS, Uni of New South Wales) which should be able to provide, at least, a set of good guesses at USENET/EUNET connection paths. Access to information contained in the database will be made available soon, through a pseudo-user on elecvax. As well as mail routing information, requests for "machine name at institution X", human contact names and addresses, and other data will be serviced. A typical database entry looks like this:

```
GROUP: usa.oh
NAME: cbosgd
ADDRESS: Mark Horton
         AT & T Bell Laboratories, Operating Systems Group
         Rm. 2C-249, 6200 E. Broad St., Columbus, OH 43213
PHONE: +1 614 860 4276
NETADDRESS: cbosg!cbosgd!mark
~NEWS: /* other sites that have news connections to this site */
  aat      astrovax burl     cbdkcl   cbneb    cbosg    cbscc    cbuxc
  decvax   felix    ihnp4    mddc     mh3bcl   mhuxl    nscs     osu-dbs
  philabs  qusavx   rlgvax   sdcrdcf  ucbvax
~MAIL: /* other sites that have mail connections to this site */
  aat      astrovax burl     cbdkcl   cbneb    cbosg    cbscc    cbuxc
  decvax   felix    ihnp4    mddc     mh3bcl   mhuxl    nscs     osu-dbs
  philabs  qusavx   rlgvax   sdcrdcf  ucbvax
NET: uucp
/* news and mail connections need not be the same */
```

The rest of the "nets" department in this issue includes some comments from Bob Kummerfeld on AUNET and the SUN software, followed by a summary of AUNET sites. Over to you, Bob.

We have always viewed the current release of SUN software as interim and distributed on the basis that people are willing to change over when the new stuff was ready. Unfortunately, in the year since the net began spreading Australia wide a lot of hosts have joined, most by getting a copy of the software from someone nearby and just hooking up. I have sent the occasional message warning people that we were going to withdraw the old system and distribute a new one soon.

The new system is a great improvement on the old. Simpler, more flexible, more powerful, easier to manage etc etc. It has a much better design at all levels. The system is layered with clean interfaces and will allow arbitrary binary messages to be sent to "handlers" on destination machines. Standard handlers will be mail and file (and network control) but we have already implemented a database enquiry handler and intend to try others. Any pair of consenting machines can have their own message types and handlers without changing the lower layers or telling anybody else. The routing and addressing is MUCH better; it will allow individual address, multicast, broadcast and explicit routing. The routing is almost optimal with minimum number of copies of messages sent.

The new system is now installed on all basser machines. We have gateway code to interface to sites running the old system (i.e. everybody outside basser). We plan to distribute it to Robert Elz next week and hopefully have it ready for general release in April.

AUNET Site Directory

Corrections are requested and should be sent to
Bob Kummerfeld (bob:basservax)

==========

Name: agsm
Address:The Australian Graduate School of Management
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 662 0273
Machine:
VAX-780, TU78, RP07, CDC 9766 + Emulex SC21V, Emulex CS-11,
Centronics 6600 printer, UNIX 4.1BSD
Contacts:
Graeme Elsworthy (gra:agsm)
Lindsay Harris (lindsay:agsm)
Colin Webb (colin:agsm)

==========

Name: basser40
Address:Basser Department of Computer Science
University of Sydney
NSW 2006
Phone: +61 2 692 2824
Machine:
PDP 11/40, TS03, PERTEC 5Mb, AMPEX DM980 + AED 8000 controller,
ABLE DMAX 16, UNIX level 7 AUSAM
Contacts:
Bob Kummerfeld (bob:basser40)
Chris Maltby (chris:basser40)

==========

Name: basservax
Address:Basser Department of Computer Science
University of Sydney
NSW 2006
Phone: +61 2 692 2824
Machine:
VAX 11/780, RM03, TE16, KMC11, CDC 9766 via EMULEX SC21, NDK 4000 printer,
TTY40 printer, UNIX 32V + 4.1BSD + AUSAM
Contacts:
Bob Kummerfeld (bob:basservax)
Piers Lauder (piers:basservax)
Tim Long (tim1:basservax)
Chris Maltby (chris:basservax)
Doug Richardson (doug:basservax)

==========

Name: bio23
Address:School of Biological Sciences
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 662 2668
Machine:
PDP 11/23 + FPU., RL02, Tektronix 4662 plotter, UNIX level 7 AUSAM

Contacts:
John Woodard (john:bio23)
==========
Name: cadvax
Address:School of Electrical Engineering and
Computer Science
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 662 3781
Machine:
VAX 11/780, CDC 9766 via EMULEX SC21, TU45, LPA11-K,
AED colour graphics terminal, Ramtek monitor, HP 7580a plotter,
HP 7221c flat bed plotter, UNIX 32v/4.1bsd mixture + AUSAM
Contacts:
Graham Hellestrand (hell:cadvax)
Peter Maxwell (peterm:cadvax)
==========
Name: cglvax
Address:Faculty of Architecture
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone:
Machine:
Contacts:
Craig McGregor
==========
Name: chemeng
Address:Chemical Engineering
University of Sydney
NSW 2006
Phone: +61 2 692 3832
Machine:
PDP 11/60, CDC RM03 look alikes, Pertec 20MB, TU10, RX02,
PP11 (reader only!), TALLY 2000, Sanders Media 12/7, UNIX level 7 AUSAM
Contacts:
Warren Simon (warren:chemeng)
==========
Name: civil
Address:School of Civil Engineering
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 662 3034
Machine:
PDP 11/40, PERTEC disks, 2*TU10, UNIX level 6 AUSAM
Contacts:
Damian McGuckin (damianm:civil)
Weeks White (weeks:civil)
Colin Wingrove (colinw:civil)
==========
Name: comm34
Address:Faculty of Commerce
University of New South Wales
PO Box 1

```
        Kensington NSW 2033
Phone: +61 2 662 3440
Machine:
    DEC PDP 11/34, CDC 80Mb, AMPEX 80Mb, Pertec Tape drive, UNIX level 6 AUSAM
Contacts:
Jimmy Sadeli (jimmy:comm34)
David Sanchez (david:comm34)
=============================================================================
Name: comm40
Address:Faculty of Commerce
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3680
Machine:
    DEC PDP 11/40, RK05, Pertec (20 megabytes), UNIX level 6 AUSAM
Contacts:
Vincent Lawrence (vince:comm40)
=============================================================================
Name: csirovlsi
Address:C.S.I.R.O.
        Adelaide SA
Phone:
Machine:
Contacts:
Bruce Nelson (bruce:csirovlsi)
=============================================================================
Name: csu40
Address:Computing Services Unit
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3590
Machine:
    PDP-11/40, RK05J, RK05F, RX01, TU10, XY11 plotter, LV11 printer/plotter,
    Centronix 6600 printer, PC11, DP11, DR11-B, Ampex DM980 + AED 8000,
    Unix level 7 AUSAM
Contacts:
Dave Horsfall (dave:csu40)
Mike Kearney (food:csu40)
=============================================================================
Name: csuvx0
Address:Computing Services Unit
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3238
Machine:
    VAX11/780
Contacts:
=============================================================================
Name: csuvxl
Address:Computing Services Unit
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
```

Phone: +61 2 662 3238
Machine:
    VAX11/780
Contacts:
==========
Name: csuvx2
Address:Computing Services Unit
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3238
Machine:
    VAX11/780
Contacts:
==========
Name: dmsadel
Address:C.S.I.R.O
        Division of Mathematics and Statistics
        Waite Road,
        Urrbrae SA
        (Private Bag No. 2, Glen Osmond SA 5064)
Phone: +61 8 274 9364
Machine:
    PDP11/34, RK07, 2 x RL01, 2 x RK05, 800/1600 bpi mag tape, Servogor plotter,
    LA180 printer, NDK S7000T printer, sundry terminals, UNIX AUSAM level 7
Contacts:
John Field (johnf:dmsadel)
==========
Name: dmscanb
Address:C.S.I.R.O.
        Division of Mathematics and Statistics
        West Lindfield
Phone:
Machine:
Contacts:
Ron Baxter (ronb:dmscanb)
==========
Name: dmsmelb
Address:
Phone:
Machine:
Contacts:
(root:dmsmelb)
==========
Name: dmtmelb
Address:CSIRO Division of Manufacturing Technology
        175 Johnston St, Fitzroy, 3065.
        P.O. Box 71, Fitzroy, 3065.
Phone: +61 3 418 0211, +61 3 418 0225, +61 3 418 0251
Machine:
    PDP 11/23 - System III (Ausam)., 256k, 2xR102, 16 lines
Contacts:
Murray J. Jensen (mjj:dmtmelb)

```
=====================================================================
Name: dmtunison
Address:CSIRO Division of Manufacturing Technology
        175 Johnston St, Fitzroy, 3065.
        P.O. Box 71, Fitzroy, 3065.
Phone: +61 3 418 0211, +61 3 418 0225, +61 3 418 0251
Machine:
    Three IMI unisons - Version 7., 1/2 & 1meg, 20meg, 3 & 9 serial,
    1 & 3 parallel
Contacts:
Murray J. Jensen (mjj:dmtmelb)
=====================================================================
Name: dmtvlsi
Address:CSIRO Division of Manufacturing Technology
        175 Johnston St, Fitzroy, 3065.
        P.O. Box 71, Fitzroy, 3065.
Phone: +61 3 418 0211, +61 3 418 0225, +61 3 418 0251
Machine:
    DEC VAX 11/780
Contacts:
Murray J. Jensen (mjj:dmtmelb)
=====================================================================
Name: dsl
Address:Digital Systems Laboratory
        School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3781
Machine:
    PDP 11/34A, AMPEX DM9100 + MSC1100, MDB DZ, hp 2631a serial printer,
    UNIX level 7 AUSAM
Contacts:
Jeff Skebe (jeffs:dsl)
Peter Ivanov (peteri:elecvax)
=====================================================================
Name: ecstats
Address:Econ Stats
        University of Sydney
        NSW 2006
Phone:
Machine:
Contacts:
Greg Defina (greg:ecstats)
Maurice Peat (maurice:ecstats)
=====================================================================
Name: elec35
Address:School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3781
Machine:
    PDP 11/35, AMPEX DM9100 + MSC1100, UNIX level 7 AUSAM
```

Contacts:
Peter Ivanov (peteri:elecvax)
Keith Titmuss (keitht:elec35)
==========
Name: elec40
Address:School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3781
Machine:
    PDP 11/40, RK05, UNIX level 7 AUSAM
Contacts:
Peter Ivanov (peteri:elecvax)
Kevin Hill (kev:elec70a)
==========
Name: elec70a
Address:School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3781
Machine:
    PDP 11/70, CDC 9766 + EMULEX sc70, TU16, MDB DZ, LP05, Diablo 630 ECS,
    UNIX level 7 AUSAM
Contacts:
Kevin Hill (kev:elec70a)
Peter Ivanov (peteri:elecvax)
==========
Name: elec70b
Address:School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3781
Machine:
    PDP 11/70, RP04, TE16, 2 * qume micro 5, UNIX level 7 AUSAM
Contacts:
Kevin Hill (kev:elec70a)
Peter Ivanov (peteri:elecvax)
==========
Name: elecadel
Address:Room E212
        Dept. of Electrical and Electronic Engineering
        University of Adelaide
        North Terrace
        Adelaide.
Phone: +61 8 228 5893
Machine:
    PDP-11/34A, 124KW, 2*RL01, 1*RL02, 8 DZ lines, 5 DL lines
Contacts:
Michael Liebelt (mike:elecadel)

```
==========================================================================
Name: elecvax
Address:School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3781
Machine:
    VAX 11/780, RP06, TU77, Data Products B900 printer + Datasystems DLP-11,
    tektronix 4015-1, UNIX 32v/4.1bsd mixture + AUSAM
Contacts:
Kevin Hill (kev:elecvax)
Michael Rourke (michaelr:elecvax)
Peter Ivanov (peteri:elecvax)
==========================================================================
Name: food23
Address:School of Food Technology
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3418
Machine:
    LSI 11/23, Pertec D4000 20Mb, AED floppy controller,
    (8" dbl sided dbl density), Sanders Media 12/7, HP7450A (A4 plotter),
    UNIX level 7 AUSAM
Contacts:
Ronald G. Bowrey (ron:food23)
Michael S. Kearney (mike:food23)
==========================================================================
Name: graphics
Address:Basser Department of Computer Science
        University of Sydney
        NSW 2006
Phone: +61 2 692 2824
Machine:
    PDP 11/34, PERTEC 20Mb, ABLE DZ16, UNIX level 7 AUSAM
Contacts:
Bob Kummerfeld (bob:basservax)
==========================================================================
Name: mathvax
Address:School of Mathematics
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 2067
Machine:
    VAX 11/750, RM80, RM03, TS11, PERTEC T9640, UNIX 4.1BSD + AUSAM
Contacts:
Veronica Paul (veronica:mathvax)
==========================================================================
Name: mech
Address:School of Mechanical and Industrial Engineering
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
```

Phone: +61 2 662 2877
Machine:
Contacts:
David Herd (davidh:mech)
==========
Name: mechadel
Address:Adelaide
Phone:
Machine:
Contacts:
==========
Name: metro
Address:University Computing Centre
        University of Sydney
        NSW 2006
Phone: +61 2 692 3492
Machine:
    dec/vax11/750, 3Mb, r102, rm02, 4 dz11, cent6600 printer
Contacts:
Burn Alting (burn:metro)
==========
Name: mhd
Address:School of Electrical Engineering
        University of Sydney
        NSW 2006
Phone: +61 2 692 2104
Machine:
    11/34A, RL01, Priam 6650 Winchester + Emulex, Cipher 75 ips 1600/800 bpi,
    AED6200 dual density-single sided floppy, Matrox 512x512 graphical display,
    NDK-4000 printer, UNIX level 7 AUSAM
Contacts:
Roy Rankin (roy:mhd)
==========
Name: moncsbeaker
Address:Department of Computer Science
        Monash University
Phone: +61 52 541 3899, +61 52 541 3909
Machine:
    VAX 11/750, running 4.1cBSD UNIX (4.2 soon),
    1 Emulex SC750 controller (SMD), 1 CDC 9762 (RM03),
    1 Fujitzu M2284N (winchester, 180Mb, dual ported),
    1 Fujitzu M2294N (winchester, 330Mb, dual ported), 1 DEC dz11a,
    2 Able VMZ32 (4 DMF32's), 1 Ethernet box (installation: Feb 84 - link
    to Kermit), Modem: (2 lines) (052) 543 5411
Contacts:
Ken McDonell (moncskermit:kenj)
Peter Nankivell (moncsbeaker:pgn)
==========
Name: moncskermit
Address:Department of Computer Science
        Monash University
        Melbourne
Phone: +61 52 541 3899, +61 52 541 3909
Machine:
    Vax 11/780
Contacts:

Peter Herman (pmh:moncskermit)
Peter Nankivell (pgn:moncskermit)
Ken McDonell (kenj:moncskermit)

Name: moneevax
Address:Electrical Engineering Department
Monash University
Clayton VIC
Phone: +61 3 541 3475
Machine:
Vax 11/750, 2Mb memory, 2 RM03 disk drives, 1 tape drive, 32 serial lines
Contacts:
Philip Grasso (pag:moneevax)

Name: monpeme
Address:
Phone:
Machine:
Contacts:

Name: mulga
Address:Department of Computer Science
University of Melbourne
Parkville VIC 3052
Phone: +61 3 341 5225
Machine:
PE3240, CDC ´MSM 80´ disk, Ampex 40Mb, selch, pasla, 2-line commux,
8-line commux, 800 bpi tape drive, Data Set Adapter, Line Printer (TALLY),
A/D D/A converter, Interprocess switch, UNIX version 7 Berkelized
Contacts:
Robert Elz (kre:munnari)

Name: mummjeeli
Address:
Phone:
Machine:
Contacts:

Name: mundara
Address:
Phone:
Machine:
Contacts:

Name: munker
Address:
Phone:
Machine:
Contacts:

Name: munnari
Address:Department of Computer Science
University of Melbourne
Parkville VIC 3052
Phone: +61 3 341 5225
Machine:

VAX 11/780, RM03, TE16, CDC 9766 + Emulex SC21V,
UNIX Melbourne Modified 4.1a bsd
Contacts:
Paul Dunn (pad:munnari)
Peter Eden (pje:munnari)
Robert Elz (kre:munnari)
Terry Hooper (th:munnari)
==========
Name: musette
Address:
Phone:
Machine:
Contacts:
==========
Name: natmlab
Address:National Measurement Laboratory
Bradfield Road
West Lindfield
Phone: +61 2 467 6059 (ronb), +61 2 467 6058 (jenny)
Machine:
VAX11/750, 4 Mb mem, 2xRA81, RL02, CDC Keystone tape, VMZ32, 2xdz11
Contacts:
Ron Baxter (ronb:natmlab)
Jenny McRostie (jenny:natmlab)
==========
Name: nrc
Address:
Phone:
Machine:
Contacts:
==========
Name: orac
Address:SIROMATH
York St
Sydney 2000
Phone:
Machine:
Contacts:
Chris Rowles (chrisr:orac)
==========
Name: physiol
Address:Department of Physiology
University of Sydney
NSW 2006
Phone: +61 2 692 2695
Machine:
PDP 11/23, Q bus + qniverter, RK05, Pertec dual RK05, DEC dual cassette,
DEC paper tape, DEC LPS lab. peripheral system, Tektronix 4662 plotter,
Sanders Media 12/7 printer, Talos digitizing tablet, UNIX level 7 AUSAM
Contacts:
David Davey (daved:physiol)
==========
Name: psych23
Address:
Phone:
Machine:

Contacts:

Name: psych44
Address:Department of Psychology
        University of Sydney
        NSW 2006
Phone: +61 2 692 3024
Machine:
    PDP 11/44 with fpu., CDC 9762 + EMULEX SC21, PERTEC t9640 + EMULEX TC11, GT40, AR-11, NDK-4000, UNIX level 7 AUSAM
Contacts:
John Holden (johnh:psych44)

Name: research
Address:
Phone:
Machine:
Contacts:

Name: rmit
Address:Royal Melbourne Institute of Technology
        Computer Science Department
        Melbourne
Phone:
Machine:
Contacts:
Mark Ross (ROSS.RCSMR:rmit)

Name: sri
Address:Sugar Research Institute
        Nebo Road
        Mackay
        Queensland
Phone: +61 79 521511
Machine:
    VAX 11/750 3Mb FP750 2 Unibuses, UDA50 disk controller - RA81 (456Mb), RA60 (205Mb) to come, TU80 mag. tape, LP25 printer, DZ11 + 2 DMF32
Contacts:
Peter Everitt (pete:sri)

Name: srl
Address:School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 3781
Machine:
    PDP 11/34A, RL01, UNIX level 7 AUSAM
Contacts:
Peter Ivanov (peteri:elecvax)
Kevin Hill (kev:elec70a)

```
==========================================================================
Name: sucyber
Address:
Phone:
Machine:
Contacts:
==========================================================================
Name: sueeise
Address:Imaging Science Engineering
        Electrical Engineering
        University of Sydney
        NSW 2006
Phone:
Machine:
    LSI11/23
Contacts:
David Skellern (daves:sueeise)
==========================================================================
Name:   sunrise
Address:Basser Department of Computer Science
        Madsen Building room G4
        University of Sydney
        NSW 2006
Phone: +61 2 692 3766
Machine:
    SUN workstation, M68010, 1Mbyte main memory,
    80 Mbyte disk + cartridge tape backup, 1024x768 bit mapped black and
    white screen, 512x640x8 colour screen, mouse.
Contacts:
Bob Kummerfeld (bob:basservax)
==========================================================================
Name: syscon
Address:Department of Systems and Control
        School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 662 2821
Machine:
    PDP-11/34, 2 x rl01, 2 x rl02, 3 x dz11,
    1 x AD-11K analog-digital converter, 1 x AA-11K digital-analog converter,
    1 x KW-11K programmable clock, NDK 4000 and DEC LA120 printers
Contacts:
Jeff K. B. Lee (jeffl:syscon)
==========================================================================
Name: timeland
Address:TIME Office Computers (Research)
        7th Floor
        99 Mount St.
        North Sydney
Phone: +61 2 929 7977
Machine:
    8 ECS5100 Z80 main processor, 256K main memory, Z80 network processor,
    Ethernet controller, 1 ECS5800 as for ECS5100, plus, Z80 disk processor,
    32 Mbyte + 12 Mbyte Winchester drive, 2 ECS5600 as for ECS5100,
```

plus Z80 disk processor, 32 Mbyte Winchester drive,
1 Mbyte Floppy disk drive
Contacts:
Tim Wooller (timw:timeland)
Greg Kable (greg:timeland)
==========

Name: ucc
Address:University Computing Centre
University of Sydney
NSW 2006
Phone: +61 2 692 3491
Machine:
PDP11/24, DZ11, RM02, UNIX level 7 AUSAM
Contacts:
Bernard Alting (burn:ucc)
Geoff Cole (geoff:ucc)
==========

Name: uccgraphics
Address:University Computing Centre
University of Sydney
NSW 2006
Phone: +61 2 692 3491
Machine:
11/34, Florida Data 600cps printer, Versatec 2160A 18" printer/plotter,
DIGIDATA 800/1600 bpi, CDC 9762 + EMULEX SC21,
Evans & Sutherland Multi-picture system, UNIX level 7 AUSAM / RSX11-M
Contacts:
Bernard Alting (burn:ucc)
Geoff Cole (geoff:ucc)
==========

Name: unicorn
Address:Siromath Pty Ltd
Phone:
Machine:
Unicorn is a Unison of variable configuration, usually including most of a
megabyte, two winchesters (40 & 20 Mb), 2 * (3 tty + centronics) cards,
and a floppy. The variability is mainly due to the 200W power supply,
which can reliably feed about half the above collection!
Contacts:
Peter Swain (swine:orac)
Kevin Dawson (bong:orac)
==========

Name: unswpower
Address:Power Department
School of Electrical Engineering and
Computer Science
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 662 2797
Machine:
PDP 11/40, RK05, RL02, DR11b, AR11, TA11, UNIX level 7 AUSAM
Contacts:
Ted Spooner (teds:unswpower)

```
==========================================================================
Name: uowcsa
Address:Department of Computer Science
        University of Wollongong
        Wollongong NSW
Phone: +61 2 282 463
Machine:
    PE 3230, MSM300 (CDC 9766) 300 MB dual ported,
    MSM80 80 MB dual ported discs, 9trk 800bpi 45ips, Tektronix 4006,
    servogor 281 flat-bed plotter, 4 line sync link to UNIVAC 1100/60,
    UNIX level 6 (a la wollongong)
Contacts:
Ross Nealon (ross:uowcsa)
==========================================================================
Name: uowcsb
Address:Department of Computer Science
        University of Wollongong
        Wollongong NSW
Phone: +61 2 282 463
Machine:
    PE 3230, MSM300 300 MB dual ported disc,
    Pertec 10MB, UNIX level 6 (a la wollongong), Logica Cambridge ring
Contacts:
Ross Nealon (ross:uowcsa)
==========================================================================
Name: uqcs23
Address:
Phone:
Machine:
Contacts:
==========================================================================
Name: uqcs34
Address:Department of Computer Science
        University of Queensland
        St Lucia
        QLD 4067
Phone: +61 7 377 2875
Machine:
    PDP 11/34, 256Kb memory, 3 x RP02 disk drive,
    RK05f fixed cartridge drive, RK05j removable cartridge drive,
    2 x RX01 floppy drive, 800 bpi 9 track tape drive, 1 x DH11, 1 x DL11,
    dial-in modem, Sanders Technology Media 12/7 printer,
    Diablo 1620 printer, 3com Unibus ethernet controller, UNIX V7 + PWB + AUSAM
Contacts:
Tim Roper (timr:uqcs34)
==========================================================================
Name: uqcscai
Address:Department of Computer Science
        University of Queensland
        St Lucia
        QLD 4067
Phone: +61 7 377 2875
Machine:
    PDP 11/34, 128Kb memory, 2 x RL02 removable cartridge drive,
    2 x DZ11, V7 + PWB + AUSAM
Contacts:
```

Tim Roper (timr:uqcscai)
==========
Name: uqcspe
Address:Department of Computer Science
University of Queensland
St Lucia
QLD 4067
Phone: +61 7 377 2875
Machine:
Perkin-Elmer 3250, 2 Mb memory, 2 x MSM 300Mb disk drive,
800 bpi 9 track tape drive, 8 x 8 line muxs, 1 x 2 line mux,
4 x dial-in modem, 200 lpm Data Products line printer, LBP-10 laser printer,
3com etherbox, UNIX Wollongong/Melbourne/Queensland V7 + 4.1 bsd
Contacts:
Tim Roper (timr:uqcspe)
==========
Name: uqcspisa
Address:
Phone:
Machine:
Contacts:
==========
Name: uqelec750
Address:
Phone:
Machine:
Contacts:
==========
Name: uqhms
Address:The Department of Human Movement Studies
University of Queensland
St. Lucia QUEENSLAND.
Phone: +61 7 377 3929, +61 7 377 3958
Machine:
PDP - 11/34 with 256Kbytes memory, 2 by RL01 disks,
1 by 30 Mbyte Winchester disk, 2 by RX Floppy Disks, 16 User Lines,
HP 7470A plotter, HP 2468A Graphics terminal with printer,
HP Low-Frequency Spectrum Analyser, A/D Interface,
Perkin Elmer Assoc. U/V Spectrum Analyser., Unix version 7
Contacts:
Lionel Barnard (root:uqhms)
David Barnbaum (root:uqhms)
==========
Name: uqk110
Address:
Phone:
Machine:
Contacts:
==========
Name: uqmeta134
Address:Dept Mining & Metallurgical Eng.
Uni of Queensland
St. Lucia 4067
Phone: +61 7 377 2051
Machine:
PDP11/34, 128kb, fpu, RK05s, 3xRL02s, Able DMAX-16(dh like),

8xhome made KL look alikes, custom interfaces, 10 vdus, 2 serial printers,
Sanders printer, Houston plotter, GTCO digitizer.
Contacts:
Cameron Davidson (probe:uqmetal34)
==========

Name: uqpsych
Address:Department of Psychology
University of Queensland
St. Lucia QLD 4067
Phone: +61 7 377 4063
Machine:
Perkin Elmer 3210, 1Mb memory, 3x67Mb hard disk, 800 BPI Mag. Tape,
Hardware Floating Point, 4x8-line MUXs 1x2-line MUX,
2x 300 Baud Dial in modems
Contacts:
Eoin Hyden (eoin:uqpsych)
==========

Name: wacsvax
Address:Department of Computer Science
University of Western Australia
Phone: +61 9 380 2878
Machine:
VAX 11/750, RA81 480Mb Windchester, RL02 20Mb Hard Disk,
TS11 magtape (don't laugh at the afflicted),
5xDZ11 8 port serial mux (40 ports), Console, lp and lots of other junk,
RM03 on loan 'till Jan 84, RA60 200 Mb removable winchester due Jan '83,
arriving Jan '84, UNIX Berkeley 41aBsd VMUNIX
Contacts:
Glenn Huxtable (glenn:wacsvax)
==========

Name: wapsyvax
Address:Pyschology Department
University of Western Australia
Phone: +61 9 380 3270
Machine:
Vax 11/750, 2Mb memory, FPA, 1 RL02, 1 RA81, 2 Able VMZ/32, 2 DZ-11,
1 Printronix printer, 1 Servogor 281 plotter, 1 Diablo Model 630,
1 NDK Model S-7700, UNIX Berkeley 4.1c bsd (going to 4.2 bsd)
Contacts:
Tom Crawley (tomc:wapsyvax)
==========

Name: warcc
Address:Western Australian Regional Computing Center
University of Western Australia
Phone:
Machine:
This is a 'mail only' host acting as a mail gateway to the WARCC DEC10
system. The WARCC also runs several CYBER 72,73 series machines, plus a
pdp11/60 (RSTS/E) remote job entry to the CYBER.
Contacts:
==========

Name: wombat
Address:Basser Department of Computer Science
Madsen Bldg room G63
University of Sydney
Phone:

Machine:
    Burroughs B1000 (currently an 1850)
Contacts:
Paul Greenfield (paul:basservax, paul:wombat)
==========

Name: zen
Address:Room D339
        National Measurement Laboratory
        Lindfield, NSW
Phone: +61 2 467 6059
Machine:
    PDP 11/34, 256k bytes memory, cache memory, floating point processor,
    rk07, 2 x rl01, Sanders Media printer, Qume Sprint 5 printer
Contacts:
Ron Baxter (ronb:orac)
==========

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.

From piers:basservax Wed Jan 11 16:32:20 1984
To: peteri:elecvax
Subject: AUUGM

If you know of any body else at your site "planning" to give a spiel at the AUUGM, please could you ask them to let the program organiser (me) know about it in advance!

From msgs Mon Jan 16 18:21:36 1984
To: auugn
From: chris@basservax.SUN
Newsgroups: aus.general
Subject: Job Advertisement

Asst Analyst/Programmer Lincoln College - Canterbury, New Zealand

A two year (first instance) position is available in the Kellogg Farm Management Unit at Lincoln College.

Appropriate tertiary qualification and experience in the use of BASIC desired. Salary range $16,123 - $18,732.

Enquiries: Dr P.L.Nuthall (Tel Christchurch 252 811)
Conditions: G.A.Hay, Registrar, Lincoln College, Canterbury, New Zealand.

From decvax!mcvax!jim:mulga Fri Jan 13 19:41:46 1984
Subject: Hello
To: decvax!mulga!auugn:elecvax

Hi. I received your letter about the Newsletter and user database. We don't at the moment have such a database, we used to, but the last 18 months has seen the EUUG dragged kicking and screaming into being a truly European organisation, and things haven't quite got back to normal yet. However, it is in the plans for the future. We now have a full time Secretary to answer all those boring questions and deal with the paperwork. Unfortunately she is in England somewhere and has no access to a machine on the network, but we have found the money to get her one. I would be interested to know what software you have for keeping your database. I am sure there would be no problem in exchanging information in such databases.

Do you want our site information, as in the form you sent?

I am interested to see just how long this takes, decvax are really kind people to spend all that money calling you and us up.

Cheers, Jim McKie.

From michaelr Mon Jan 16 18:41:21 1984
To: DCS
Subject: make or mike?

I have written a version of Make.
It runs much faster and doesn't contain many of the annoying bugs of make.
It interprets an almost complete subset of make facilities.

Michael

From mh3bcl!ianj:mulga Thu Jan 19 15:56:44 1984
To: Peteri, Piers

> uniforum (meq) Wed Jan 18 13:48:37 1984
>
> AT&T and Digital Research announced a cooperative agreement today at UNIFORUM. A specific project will be the joint development of a UNIX System V applications library containing high quality supported portable software.
>
> Steve Johnson will visit Digital Research next week to start negotiations; he welcomes comments and suggestions about this work.
>
> Steve also reports that there are 8000 attendees at UNIFORUM. This is a four-fold increase over last summer's attendance of 2000.

From mh3bcl!ircam!adrian:mulga Thu Nov 24 15:46:53 1983
Subject: Guess what
To: ianj, mh3bcl!mulga!auugn:elecvax

the Ritchie Kernighan C book has just been translated into French.

From du:moncskermit Thu Nov 10 14:41:33 1983
To: netgurus:basservax

~s AUSAM csh bug fix

In the Ausam 'csh' there is a bug which causes the "~(user name)" feature not to work consistantly. Csh at various stages during the execution of the users commands is forced to close all the open files besides the standard files(eg. STDIN STDOUT and STDERR). When 'csh' closes the open files it fails to reset the "password file open flag - pwfl", if it had been open.

You must modify the procedures "closem" and "closech" in the program "sh.misc.c" to set the flag "pwfl" too zero at the end of each of these routines. You cannot include the "pwf.h" file in this program because of a variable name clash so just above these two routines place the line.

```
extern short pwfl;
```

I imagine that some of you will have this problem fixed but I know most don't because I have run Ausam "csh's" from Queensland, Melbourne and Sydney.

Craig Bishop.
Deakin University.

From richardg Mon Jan 23 19:15:08 1984
To: auugn
Subject: A program that CORRECTLY implements the ANSI tape standard.

I have written a program that implements the ANSI X3.27-1977
tape standard, or as much as can be implemented under UNIX.
I even read the standard, rather that whatever everyone else does.
The program implements the newest version 3, and handles everything except
multi-tape volumes, and spanned record types.
Its written in C and seems to work on both VAXes and PDPs.

Mail to richardg:elecvax, and I can immediately send the manual entry
or the program itself. The latter is network notwithstanding.

Richard Grevis, UNSW

From mh3bcl!research!wild!andrew:mulga Fri Dec 2 08:46:05 1983
Subject: Unix edition one
To: auugn

This is the first in a series of notes on the early editions of Unix.
They are random remarks on what interested me in reading the programmer's manuals for the various editions of Unix. These notes are not proprietary and will be produced at random times.

EDITION ONE: (November 1971)
---

1) the `tm' command was analogous to the `time' command. If invoked with arguments, it executed the command and gave the times used during that command. If invoked without arguments, it gave two columns of numbers: the times since boot, and the times since the last `tm' command.

The categories of times were:

"tim"	(real time)	hrs:min:secs
"ovh"	(time in sys)	ditto
"dsk"	(waiting for disk)	ditto
"idl"	(idle time)	ditto
"usr"	(user time)	
"der"	(RK disk error count !!)	

The second column was given with one decimal place in units of seconds.

2) the `time' system call returned the number of sixtieth of seconds since Jan 1, 1971. This is different because the maximum time was about 2.5 years.

3) file names were limited to 8 bytes; accordingly directory entries were 10 bytes long.

4) the permission bits were quite different:

01	write, non-owner
02	read, non-owner
04	write, owner
10	read, owner
20	executable
40	set user id on execution

5) userid - username mapping was kept in /etc/uids.
---

that's all folks! I would welcome any feedback or questions!
andrew

From timr:uqcspe Fri Nov 25 13:16:41 1983
Full-Name: Tim Roper
To: auugn:elecvax
Subject: contribution

Here follows an offering for an AUUGN, for which I admit no responsibilty.

Subject: Real Programmers don't .....

Real programmers don't write specs - users should consider themselves lucky to get any programs at all and take what they get.

Real programmers don't comment their code. If it was hard to write, it should be hard to understand.

Real programmers don't write application programs. They program right down to the bare metal. Application programming is for feebs who can't do system programming.

Real programmers don't eat quiche. In fact, real programmers don't know how to spell quiche. They eat twinkles and szechuan food.

Real programmers don't write in COBOL. COBOL is for wimpy applications programmers.

Real programmers' programs never work right the first time. But if you throw them on the machine, they can be patched into working in 'only a few' 30-hour debugging sessions.

Real programmers don't write in FORTRAN. FORTRAN is for pipe-stress freaks and crystallography weenies.

Real programmers never work 9 to 5. If any real programmers are around at 9AM, it's because they were up all night.

Real programmers don't write in BASIC. Actually, no programmers write in BASIC, after the age of 12.

Real programmers don't write in PL/1. PL/1 is for programmers who can't decide whether to write in COBOL or FORTRAN.

Real programmers don't play tennis or any other sport that requires you to change clothes. Mountain climbing is OK, and real programmers wear their climbing boots to work in case a mountain should suddenly spring up in the middle of the machine room.

Real programmers don't document. Documentation is for simps who can't read the listings or object deck.

Real programmers don't write in PASCAL or BLISS or ADA, or any of those pinko computer science languages. Strong typing is for people with weak memories.

Real Programmers don't draw flowcharts. Cavement drew flowcharts and look how much good it did them.

Real Programmers don't write in APL, unless the whole program can be written on one line.

Real Programmers don't write in LISP. Only faggot programs contain more parenthesis than actual code.

Real Programmers scorn floating-point arithmetic. The decimal point was invented for pansy bed wetters who are unable to think big.

Real Programmers don't bring brown-bag lunches. If the vending machine sells it, they eat it. If the vending machine doesn't sell it, they don't eat it. Vending machines don't sell quiche.

Real Programmers like vending-machine popcorn. Coders pop it in the microwave oven. Real programmers use the heat from the CPU. They can tell which jobs are running from the rate of popping.

Real Programmers disdain structured programming. Structured programming is for compulsive neurotics who were prematurely toilet-trained. They wear neckties and carefully line up sharp pencils on an otherwise clear desk.

Real Programmers don't believe in schedules. Planners make schedules. Managers firm up schedules. Frightened coders strive to meet schedules. Real programmers ignore schedules.

Clippings

The item at top left came from "The Australian" on 17/1/84 and "Unix Net leads the World" is from "Computerworld" 21/10/83. All the others come from "Whats New in Computing" September, October, November and December 1983.

IBM to provide Unix on personal range

From PETER SAMUEL in Washington

IBM is to offer the Unix operating system on its personal computers.

It has made a licensing agreement with the owner of Unix, AT&T Information Systems, formerly Bell Laboratories.

It had previously been thought in the industry that AT&T itself might be the first major company to introduce Unix into a $2000 personal computer.

AT&T is expected to introduce a personal computer based on its 32-bit Bellmac microprocessor and based on Unix.

It is expected to sell a family of Unix-based PCs for office use from its nation-wide chain of telephone stores.

They will have a major emphasis on networking and programming for the use of communications.

Unix enables enormous programs to be manipulated at reasonable speed as compared with CP/M, MS-DOS and other well-known operating systems.

IBM's version of Unix will be called Personal Computer Interactive Executive.

It was customised for IBM by Interactive Systems Corp, a small software house in California, and will be on sale at all IBM stores and IBM dealers in the US at $US900 ($938) from April.

The company is believed likely to stress in its advertising of the product its advantage in communications and in multi-tasking.

Also, Unix has advantages in handling "windowing" programs, in which separate documents are shown on the screeen at the same time.

Unix has been around for some years but has mostly been used in $10,000-plus computers.

XENIX FOR TRS-80 MODEL 16

The XENIX multi-user operating system developed by Microsoft is to be the standard operating system on Tandy Electronics' TRS-80 Model 16 microcomputer. XENIX gives the Model 16 full multi-user, multi-task abilities without any degradation in performace. Up to three users may operate their different applications programmes or share the same programmes and data, such as accounting, inventory control, word processing or electronic filing. They also share the same peripherals attached to the Model 16. The several multi-user applications software packages for XENIX equipped Model 16s include a full complement of interactive Australian modified accounting packages and a high capcity inventory control system. Tandy also offers Microsoft's Multiplan, an advanced spreadsheet programme for planning and modelling, in a multi-user version. XENIX will operate on any TRS-80 Model 16 equipped with 256 Kbytes of memory and a hard disc and on similarly equipped Model 11 or Model 12 microcomputers that have a Model 16 upgrade kit installed. All applications software currently offered by Tandy for the Model 16 an also be moved to the XENIX system. The XENIX operating system includes many sophisticated features including tree-structured directories, device independent I/O, chaining of programme input and output, and foreground and background programme execution. In addition to the basic operating system supplied with all Model 16s, a XENIX developed system including the C language for programmers and a version of Microsoft BASIC developed for use with XENIX on computers using the 68000 microprocessor are offered. A floppy disc based version of XENIX for single user operation will also be available shortly.

Tandy Electronics.
91 Kurrajong Avenue,
Mt Druitt 2770

POWERFUL EXPANDABLE SYSTEM

The System 6400 is offered as a powerful, economical system designed for sophisticated computing applications. It comes standard with 256 Kbytes of main memory and a nine hex backplane that allows easy expansion. Configurations are available with floppy discs, 28 megabyte Winchester hard discs, ¼in streaming tape drives and a range of optional workstation components. It can be expanded to large database applications while maintaining basic system architecture, thus protecting hardware and software investments. It can be used as a powerful stand-alone system or as part of a large multi-user network. System software choices include ASCII Commercial Software packages, standard Digital software systems, MUMPS and UNITY. The System 6400 is providing dependable performance in a wide variety of commercial, educational, industrial, scientific and public utilities applications. The System 6400 will run the following operating systems and languages: CP/M80, UNIX, RT-11/TSX+, RSX-11M, RSTS-E, or MUMPS; Basic, Dibol/DBL, Cobol-Plus, Fortran-IV, Fortran-77, C, Macro, and many others.

ASCII Computer Suppliers,
Suite 2/172 Liverpool Road,
Enfield 2136

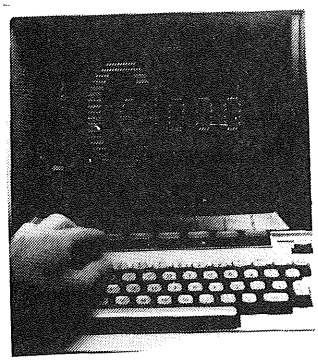
UNIX SYSTEM V FOR 286

Intel, in association with Western Electric, have developed a port of the UNIX System V operating system for the iAPX 286 microprocessor, thus providing movement towards standardisation within the industry. The 286, with high performance and features such as on-chip memory management, also offers Microsoft's XENIX operating system.

Intel Australia Pty Ltd.
Level 6,
200 Pacific Highway,
Crows Nest 2065

C COMPILERS FOR HP 64000

Hewlett-Packard's new C compilers for the HP 64000 logic-development system support the 8086, 8088, 68000, Z8001, Z8002, 6800/6802 and 6809 microprocessors, complementing



the already available Pascal compilers and the microprocessor-specific assembly languages. Assembly-level programming is best suited to the sections of a programme that require attention to details of microprocessor operation and other system devices. Pascal, by contrast, provides a very structured environment for flexible and thorough definition and manipulation of data structures and algorithms. C allows closer interaction with the processor than Pascal, while retaining advantages of high-level language structure, readability and ease of maintenance, as in Pascal. The relocatable object code modules from assembly language or the compilation of C and Pascal languages can be brought together easily into an executable programme using an HP 64000 system linker. The operating environment provided by the system-hardware-emulation capabilities of the HP 64000 then provides debug and verification of the resulting programme. The C compiler also passes key symbolic information through the linker into the emulator/analyser to expedite debugging, for example, information source-code line numbers may be used as trace points or software break points during emulation.

Hewlett-Packard Australia Ltd.
31 Joseph Street,
Blackburn 3130
Insert Q442 on Information Feedback Card

PRIME C COMPILER

The Prime C Compiler is an additional programmer productivity tool to provide fast compilation speeds, interactive debugger support, good syntactic error recovery, modern flow control and data structures, as well as a rich set of operators and data types. Other key features include separate compilation, data sharing and data initialisation. The new product fully implements the C language, making it easy to use and more effective for many tasks than specialised languages such as FORTRAN or COBOL. The Prime C Compiler provides extensive UNIX-like library calls plus access to Prime's standard library calls. Combining the interactive capabilities of the PRIMOS operating system with Prime's Source Level Debugger, programmers using C can create, edit, compile, execute and test programmes online with a great reduction in test and debugging time. Prime's C Compiler also utilises EMACS, an interactive full-screen display editor which allows the user to compile source code without leaving the editor, disrupting the screen or leaving the routine being debugged. Prime's Command Procedure Language (CPL) allows sequences of operating system commands and CPL directives to be stored in a command procedure file that can be executed by specifying the file name. The C Compiler can be used to produce transportable code, which can easily be moved from machine to machine with minimal modification, and has compilation speeds of up to 5,000 source lines per minute, producing fast and highly efficient object code.

Prime Computer of Australia.
15 Blue Street,
North Sydney 2060
Insert Q444 on Information Feedback Card

UNIX WITH C FOR ERG

The UniPlus+ operating system frtm UniSof Corporation provides for the ERG 68-696 CPU the full power of Bell Laboratories' UNIX System III operating system with Berkeley enhancements. The multi-user/multi-tasking capability utilises the Motorola 68451 memory management unit and provides a flexible, expandable, operating system for programme development, text preparation, and genera office tasks. Uniwplus+ is derived from the rece tly released Bell Labs' UNIX System III. Standard features include: System III utilities (over 150 programmes); source code control system (SCCS); communicative software (UUCP, cu, etc); "C" compiler; 68000 assembler (as). The "C" compiler; 68000 assembler and linker/loader are bundled with the operating system. The operating system includes a full UNIX kernel implementation plus many Berkeley 4.1 enhancements. Features include: C shell (csh); visual editor (vi); a pagination command for CRTs (more); programme and documentation locator (whereis); terminal independent library (termcap). Minimum hardware requirements to fully support UniPlus+ are 384 Kbytes of RAM, 5 MB hard disc, 68451 MMU, and a console serial port. UniSoft intends to support demand paging and the advanced 32-bit architecture along with the System V upgrade. UniSoft has over 50 ports of the UniPlus+ to the Motorola 68000/68010.

SI Microcomputers Pty Ltd.
23 Berry Street,
North Sydney. 2060.
Insert S642 on Information Feedback Card

IDRIS OPERATING SYSTEM

IDRIS/R68K, a multi-user/multi-tasking operating system from Whitesmiths, is pin for pin compatible with version 6 of UNIX and comes with a C compiler which is fully compatible with version 7 of UNIX. Edition 2.2 features: UNIX-compatible libraries which include most of the functions from the standard UNIX libraries; UNIX III system calls in line with moves to adopt UNIX as a standard; more Ada-style library support than the exception handling of release 2.1; more optimisations added to the code generators (the MC68000 code generator now makes extensive use of the single-bit manipulation instructions); improved system interfaces (the CP/M interface has been altered for CP/M 2.2 features and, except for a small header file, the same interface can be used for CP/M-80, CP/M-86, CP/M-68K and PCDOS); and more software tools (including a spelling checker, facility for compressing stored files, automatic reformatting to multi-column display, post-processor for the runtime profiling package and a programme which shows executable commands at any time). All versions of IDRIS support the same file system and library formats, regardless of host computer word size or byte order. The resident is divided into a host-independent kernel plus a small processor dependent memory collector and a collection of bus-dependent device handlers. The minimum requirements to fully implement IDRIS are 192 Kbytes of RAM, 5 Mbytes of hard disc, clock interrupt and a console serial port. An optional memory mapped version of IDRIS is also available.

SI Microcomputers Pty Ltd.
23 Berry Street,
North Sydney 2060
Insert S480 on Information Feedback Card

Unix net leads world

By David Noble

SYDNEY — The developers of the Australian Computer Science Network, a Unix network claimed to be the most sophisticated of its type in the world, intend to make the service available to commercial users interested in computer science research.

Development of the network began in 1975 when Sydney University and the University of NSW started work on Unix research. While the initial membership included only those universities, the network now supports 72 host computers at seven universities around Australia and CSIRO's VLSI facility.

The network uses Datel, Dialup and Csironet to send messages around Australia but is expected to move on to Austpac when a line is installed. Users are currently able to use the system for inter-machine mail, file transfer and real-time access.

The Unix network is achieved via an applications software package written in C which runs on all 72 participating computers under Unix.

According to Piers Lauder, a programmer working on the network through a two-year $A90,000 grant from Telecom, the aim is now to allow commercial users access to the network for computer science research for a nominal fee.

The Australian Computer Science Network is based at the Basser Department of Computer Science, Sydney University, Sydney, NSW 2006. Tel: (02) 692 2222.

Symposium on UNIX

[Advance Notice]

James Conran Pty Ltd, who have arranged two excellent symposia in Sydney during the past 15 months (on Local Area Networks and Databases), are now arranging a symposium on UNIX for the first week (3rd, 4th & 5th) of April.

Several aspects of the arrangements, including the location, are still fluid but will be finalised very shortly. The list of overseas invited speakers who have already accepted includes Ken Thompson and Brian Kernighan of Bell Laboratories, Ted Dolotta of Interactive Systems, Santa Monica, and Sam Leffler of Lucasfilm. More will be announced shortly.

This will be an important and unique opportunity for anyone in Australia wishing to evaluate UNIX, or to meet many of its principal designers and users. (As this will be a first class symposium at a first class venue, it will not come cheap; so start saving now!)

Further information will be published in due course or may be obtained by phoning Malcolm Cardis on (02) 922-6833.

PN234: The Australasian Unix Users' Group Meeting, August 1983

Doug Richardson, Sydney University

This meeting was held at the Flinders University Staff Club on Monday, August 29th 1983 at 10am. The meeting was chaired by John Field of CSIRO. The next meeting will be in organised by Digital Electronics in Sydney in the final two weeks of February 1984.

Overview of Unix in Adelaide

John Field recounted the equipment at the Division of Maths and Stats in Adelaide (dmsadel): PDP-11/34, RK07, 2xRL01, 2xRK05, NDK7000T, LA180, magtape and plotter. It runs AUSAM level 7 used for data manipulation (unix, glim, fortran, pascal), word processing (nroff, neqn, tbl), and communications (to VAXes in Canberra and Sydney and the CYBER 76 at DCR).

Rod Curtin, Adelaide Uni Elec Eng (elecadel): PDP-11/34, 2xRL01, RL02 + intel MDS + Z80 micro kits + access to CYBER and VAX. SUN connection to:

Adelaide Uni Mech Eng (mechadel): PDP-11/34, 2xRK05, 30Mb Winch, RX02, 800bpi tape + CSIRONET + Lab Network (3xLSI-11 <RT-11>, 3xAIM 65, Z80). Robotics have a Puma, Graphics have a MATROX 8086. Access to CYBER 176 and VAX 780.

Adelaide Uni Computer Centre: VAX 780 with VMS. Bad experience with EUNICE - found it expensive to run, slow, files incompatible with VMS. UNITY not much better. Also experience with DEC C compiler (see later session).

Trevor Norman, Flinders Uni Earth Sciences: PDP-11/23, 4xRK06, tape. Catch and process environmental data using lots of strange data logging devices. Use UNIX to massage data before processing on a Prime at Flinders Comp Centre, also for word processing (NDK7700, Qume, Oki).

John Bennett, Defence Research Centre, Salisbury: 3 groups but only described his (Computers for Design and Manufacture): in process of getting UNIX on a UNISON. Have 5 PDP-11s used for workstations or RJE for an IBM.

Chris Price, SOLA (Scientific Optical Labs Australia): Doing on-line order entry system. Using WICAT 150 and UNIX so can change hardware to suit changing times and places (eg hardware in Brazil may be different). Sees weaknesses in C - no index sequential files, no commercial sort package (unix sort is unsatisfactory). Using C anyway. Need 4th generation language.

Robyn Lamacraft spoke about CSIRO Manufacturing Technology: UNISON (1Mb memory, 20Mb disk, floppy) with SIROUNIX used for data manipulation by metal forming group, should be faster when get 80Mb disk. Would like to use PROLOG for expert system.

Robert Elz: 1983 Unicom Meeting

Unicom meeting combines several UNIX groups: Usenix, Software Tools, etc. It attracts about 1500 people, a big equipment exhibition in a big hotel and runs for about 4 days of parallel sessions starting at 8:45am (ugh!). Too much for one person to cover, these are a quick cross section.

Keynote by Mike Lesk: about the psychology of program design using Library and Telephone systems at Bell Labs as paradigms, conclusions:

- users prefer key words to menus (menus should fade out after two weeks)
- less documentation is better
- want less of time consuming error messages

AT&T presentation: usual graphic showing growth of UNIX licence numbers in all categories followed by announcement of new products followed by ads for same. The products:

- S (nice maths stats and graphics package) cost < $1000
- UNIX Writer's workbench (useful goodies for pro writers) cost ~ $1000
- UNIX Instructional Workbench (glorified 'learn', but sell binaries only so cannot write your own lessons) cost ~ $2000 per cpu.

Many Presentations: UNIX on the ---- machine. There are 16032 implementations that do efficient memory management. Most new UNIX implementations are by commercial enterprises. Most talks were at the user level. There is little kernel hacking (except at Bell and Berkeley).

Korn Shell: has everything (except windows), is smaller and faster BUT is only inside Bell Labs and they wont let it out.

Berkeley and Bell versions of UNIX are converging - each are adopting features of the other.

cat -v considered harmful: An entertaining talk to the whole session, by Rob Pike, warning against creeping featurism.

DEC Hardware Preview: releases for the coming year:

- The Venus (nee 11/790?): 4 times the power of the VAX 11/780.
- a dual processor VAX done Perdue style, called the Superstar
- 3 new micro VAXes that emulate some of the (COBOL) VAX instructions:

  seahorse 1 is a VAX on a chip
  seahorse 2 is a 3 chip set
  microVAX

Piers Lauder: Design of ACSnet

The Australian Computer Science Network (ACSnet) will be a wide-area network written in C for UNIX environments, available to anyone with a UNIX licence. The work is funded by a Telecom Research Grant for 2 years. Piers Lauder at Sydney University is doing the design and development, Robert Elz at Melbourne Uni is developing the low level support for AUSTPAC/X.25. The grant also pays for 2x9600 baud X.25 host connections to Austpac at Sydney and Melbourne, and it covers the Austpac charges.

ACSnet is a message transfer network, made up of nodes (destinations) connected together by virtual circuits (links). Messages are transferred reliably from any node to any other node(s). There is a negative acknowledgement if a message could not be delivered. It is a 3 level hierarchical network providing node-to-node virtual circuits, end-to-end message routing service, and end-to-end protocol.

Flexible addressing scheme allows several forms: broadcast (to every node), single (to a particular node), multi-cast (to a set of nodes) and explicit (via a nominated route). The media used may be RS-232, Dial-up, CSIRONET, Austpac, or Ethernet. Gateways will be available for SUN and UUCP (USENET).

Messages arriving at destination nodes are passed to a handler. Basic handlers are provided for mail transfer and file transfer. These or other handlers may be user written to provide special services, eg print servers, RJE servers, names servers. The end to end protocols used by "mailer" and "filer" allow multiple files in a message, multiple users at multiple destinations, data integrity and delivery acknowledgement.

As far as possible network management is automatic. A command file controls the stats. Access permissions are controlled by a per-site configurable routine that sets flags if the user is allowed to use addressing modes (broadcast, multicast, explicit), file transfer and/or virtual circuits, non-standard handlers, etc. Statistics are kept on bytes received/sent and passed to/from all other nodes. These allow per-node charges to be made.

Messages use trailing headers (ie the header information follows the data) so that the length of the header is easily changed. The data contains the end to end protocols; the header contains the source and destination addresses, handler, environment, time-to-die, travel time, route, CRC, etc. all as null terminated ascii strings.

A state file at each node contains info on all foreign nodes and links (Node types are foreign, workstation and terminal). For each link there is info on state, cost, speed and gateway handlers. On any change of state, (signalled by a broadcast state-change message), the state file is updated and compiled into a routing table of shortest paths, including a link table for the extended reverse path forwarding algorithm. Shortest path info is kept for both "cheapest" and "quickest" routes.

ACSnet will be released to beta test sites sometime in October. General release is scheduled for January. Support for Austpac will follow asap. The resident daemons will be smaller and faster than those in the current SUN. It will be compatible with SUN.

Chris Rowles: Towards 4.2 BSD Unix

SIROMATH supports 4.1 BSD quite happily; it is the only version of Unix that would meet the needs of their users. 'cat -v' has its devotees! SIROMATH has had its problems; in 1982 the staff grew from 4 to 18 practically overnight, the users are widely distributed and have conflicting needs.

Differences between System III and 4.1 BSD: System III is a swapping based system that is much faster at executing processes. It handles more users than 4.1 BSD, but each process must be smaller than the physical memory

available. This is the system used at Basser and at UNSW. It currently runs only on VAX 11/780 but could easily be moved to other VAXes.

4.1 BSD is a Virtual Memory system; processes may be larger than the available physical memory. It has good support for local area networks ('sockets') that is not available in system III (but should be in System VI). The 4.1c file system is more efficient - it has variable cluster size, a split swap region and uses bigger blocks.

A VAX 11/750 will support about 15-20 users under 4.1c (only about 10 could be handled under 4.1a). Berkeley will probably cease development after releasing 4.2, their good people seem to have left.

Both systems have common system calls at the source level, the kernels are functionally similar. Programs running under System III can be run on 4.1 BSD after re-compilation, but the reverse is not necessarily true. It is easy to migrate between systems, though a standardised system call interface is needed (there is a draft proposed ISO standard available). The file systems of System III and 4.1c are incompatible.

Ron Baxter: Yet Another Way of talking to remote machines

A new function was added to awk ('easy' to do) that reads and writes to special files (devices). In reading, it returns a line at a time, or times out and returns an error indicator. In writing, it sends a line at a time, or times out and returns an error indicator.

Awk, augmented by this function, can be used for writing programs to control (eg) the automatic sending of data over a network. (ugate, utalk, con, etc are hard to use from shell scripts because of the tricky failure condition handling.)

An example program showed how the method works. It is not easy to use, but does the job better than shell/sift scripts.

Lunch

Business Meeting

Peter Ivanov will return as editor of AUUGN. There is $1500 in the kitty. The next meeting will be organised by Digital Electronics in Sydney in the last two weeks of February. The meeting after that will be in Melbourne in July or August.

Ken McDonell, Monash: The VMS C Compiler

This compiler is distributed by DEC. It costs $6500 per cpu. It is used at Monash to put Unix utilities (ed, awk, grep, pi, px) on the Computer Centre VAX running VMS. The manual, unlike most DEC documentation, was 1 volume (not 5) and was readable.

Moving applications from Unix to VMS requires a choice:

- EUNICE did not work acceptably. Very expensive to run.

- Homegrown Unix within VMS not efficient either.

- VAX-11 C expensive, but supported by DEC

VAX-11 C has serious problems, but is usable; here are some hints:

- forget releases prior to 1.2

- "extern int x" is identical to "int x"!

- structure element naming has problems, some cases give no errors but produce bad code.

- The linker does not distinguish case, so global names cannot rely on this (as does about 30% of Berkeley code).

- structure-valued functions must have arguments (a dummy perhaps), else the optimiser arranges to overwrite the pointer to space reserved on the stack.

- constants associated with case labels in a switch statement must fit into 16 bits. System code sometimes relies on 32 bits here (eg status of RMS call).

- non-standard include file processing. Cannot use files with period in name (eg .h files!)

- structs are byte aligned, not word aligned. Sometimes fools programs that do not use sizeof.

- asm(".....") wont work

- some useful routines are omitted from the run-time libraries eg: asctime, assert, crypt, fork (but vfork is included), index, ioctl(!), link (disastrous), libcurses, libdbm, qsort, re_comp, re_exec, rindex, stat, fstat (argh!), system, ttyname, unlink (but there is a remove), etc. The effect is that terminal oriented programs (editors in particular) are difficult (or nearly impossible) to port.

- Problems with files: the compiler does not accept standard unix path names. Seeking only works on stream files, and then is flakey - best rewind and do a sequential read. Concurrent i/o ok if all are reading, else forget it. VMS does not support links, so forget them.

- Command line arguments: meta characters and i/o re-direction do not work (eg "fred*" is passed through as "fred*" !) but the guys at Monash have a fix for these.

- Exception handling problems eg: px cannot trap a divide by zero.

The compiler is probably as good as it is going to get, unless we apply pressure on DEC to improve C under VMS, or perhaps use VAX-11 C to bootstrap a real C compiler then re-do the run-time environment. BUT both solutions are severely constrained by long-term decisions within VMS, DCL, LINKER, DISK ACP, RMS, etc.

Ron Baxter: S on Unix

S is a language and system for data analysis developed by Bell's stats group. It is quite neat and powerful, but you need a good chunk of a VAX to run it. Versions for PDP 11/70 and 11/44 exist and are perhaps adequate.

It is good for doing all sorts of data analysis eg time series, linear regression, plotting, etc.

Experimental copies of S were distributed a year or two ago, under non-disclosure licences. Now you have to buy S for a thousand dollars or so.

S is written in RATFOR, probably to allow access to the extensive FORTRAN libraries of mathematical routines. It maintains many small files and runs many simultaneous processes; it is expensive to use. It assumes that each user has plenty of CPU and their own multi-pen plotter.

It is easy to use. The influence of APL is clear. You can define your own data structures. There is a powerful macro facility. Users can define their own functions, written in RATFOR/FORTRAN or C, but this requires a study of the documentation and some practice.

The interactive graphics is particularly good. All sorts of plots are easy to produce.

Tim Long: Quick Benchmarks of the Machines on Display

A simple cpu-bound benchmark was run on each of the machines on display. The benchmark was 'echo 99k2vp8opq | /bin/time dc > /dev/null'. It uses dc (the desk calculator) to calculate the square root of 2 to 99 decimal places, and to 'print' the result in decimal and then in octal. The results are in fact never printed, being piped to /dev/null. The user time is all that is compared.

This benchmark has been applied to a large number of machines. It has (up until now) been useful because most manufacturers have not optimised dc, so the results are not likely to have been distorted by attempts to optimise for benchmarks.

The results were:

D.E. UNITY (DE-68K)	11.3 sec
Plexus P/25	14.1 sec
NCR Tower	21.3 sec
Wicat 150WS	27.3 sec
Unison	32.6 sec

By comparison, a VAX 11/780 clocks about 5 to 6 secs, VAX750 9 sec, PDP11s range from 27 secs (11/23) to 6 secs (11/70), PDP 11/34s range from 12 to 19 secs depending upon the presence of a cache. Perkin-Elmer range from 12.5 secs (32/10) to 7.9 secs (32/40).

Peter Swain: Unison

The UNISON is a much-maligned 68000 based machine designed at the University of Melbourne, manufactured in Australia, running a version of Unix developed by SIROMATH. Its performance is miserable at the moment, but the next release of the software will show a marked improvement.

The main hardware innovation is that memory management is done on the memory boards, not in the cpu. This feature should give the UNISON a performance plus when running virtual memory jobs.

Have had problems with the power supply, but these have been solved.

Couldn't do overlapped seek. This has been solved.

The C compiler is now producing better code. The optimiser now uses shorts for suitable multiplies.

The new disks are faster.

Geoff Cole: Architectural quirks of PDP11s

In the process of bringing up Unix on a range of PDP-11s (34, 44, and a 24) we have become very aware of the variety of differences among PDP-11s. Unix had to be modified to run on each of our machines, a task that would have been easier if the original code had not assumed that the presence of one feature implied the presence of another.

Memory management: the original PDP-11s were limited to a 16-bit (64Kb) address space. MiniUix runs on such machines. A memory management unit (MMU) may be added to most PDP-11s (23 up) to allow the use of more physical memory. The MMU maps the 16-bit virtual address space onto physical memory, using 8 relocation registers. Two sets of relocation registers exist - one for user programs (USER mode) and one for the system (KERNEL mode). Some PDP-11s (44, 45, 70) have a third mode - SUPERVISOR mode, with a third set of relocation registers that Unix (and most DEC operating systems) ignore.

I & D space mapping: To allow bigger programs, some PDP-11s (44, 45, 70) map the virtual addresses of code ('instructions') and data to separate physical addresses. This rules out the unsavoury practice of instruction modification and allows up to 64Kb of text (code) and 64Kb of data. Level 6 Unix will run on any PDP-11; Level 7 Unix (in standard form) requires a separate I&D space machine.

BIGUNIX is version of Level 7 Unix that has been hacked so that by juggling the memory management registers at run time it can address all the relevant system code, though user processes are still restricted to 64Kb.

Floating Point Emulation: If your PDP-11 does not have a floating point unit, execution of a FP instruction results in an illegal instruction trap. UNIX detects this, and uses a section of the user program to emulate the FP hardware. (This is the function of the -f flag to cc. It causes the loading of the floating point interpreter code.) Unfortunately this method does not work on a machine with separate I&D space. The hardware wont let a user mode process read the instruction to be emulated unless you (like UNSW) choose to modify the hardware slightly. Otherwise you must put the emulation code into

kernel mode, or (better) simply buy the Floating Point hardware.

Busses: Most PDP-11s (23 up) run on a UNIBUS that provides 18 address lines (ie a maximum of 256Kb of physical memory). Some later PDP-11s (24, 44, 70) have an Extended UNIBUS with 22 bit addresses (a 4Mb memory limit). Most Unibus devices only know about 18-bit addresses, so the Extended Unibus requires its own set of relocation registers. The Level 7 bootstrap did not reset these, assuming that if you had an Extended Unibus then you had an 11/70 and that you would bootstrap from the MASSBUS on such a machine. (The MASSBUS is only used on 11/70s, it has 22 bit addressing). Some smaller PDP-11s (23, LSI) use a QBUS.

Mapped Buffers: To allow a larger number of buffers, the MAPPED_BUFFERS scheme allocates one relocation register to access buffers. This allows more buffers, anywhere in memory, BUT Unix needed to be modified to allow for the option of Unibus Mapping that is required by an Extended Unibus.

We are working towards one set of sources that can be readily adapted for any machine. The key to this is to not assume that one hardware feature implies another.

Tim Long: How to get a fast file system without changes to it

Berkely 4.1c has a much faster file system, done by increasing the block size and grouping files in 'clusters'. The following improvements follow a different path, and could be patched into 4.1c to further enhance its performance.

Unix filesystems tend to become messy as time goes by. As blocks are used and then freed, the groups of contiguous free blocks tend to become smaller and more separated. This leads to much random seeking when writing a new file (eg during a sync). Reading then suffers the same fate.

An improvement is to sort the free list, so that the seeks sweep the disk in only one direction. Experience showed that it was more efficient to sort just a part of the free list rather than the whole list. The sort is done just before a block of the free list is written to disk. This results in groups of 50 blocks or so, each group sorted for a sweep down the disk. (An improvement is to have alternate groups sweep up). This simple change makes a dramatic improvement in file system performance, and may be implement to advantage even on small machines (eg PDP 11s). When a new file is to be written to disk, the free blocks are allocated in a 'tidy' manner, and disk head movement is in sweeps up and down the disk, reducing the overall seek time.

For machines with a large buffer pool (eg our VAX has 512 buffers), it is possible to delete a file before its blocks have been written to disk. The standard disk driver remains unaware of this, and writes the blocks to disk anyway. We have a fix for this bug.

Unix notices when a process is doing sequential reads and does 1 block read-aheads. It is normal for processes to consume blocks at a quicker rate, so a larger read-ahead is advantageous. Of course, it is pointless reading so far ahead that the buffer pool fills to the point that the full buffers are needed for other things. So our new disk driver doubles the read-ahead until a sensible limit is reached.

The limit is found by date stamping all buffers. The current lifetime of a buffer is then easily found. The read-ahead is doubled if the buffers being processed are older than one quarter of the lifetime, else the read-ahead is halved.

The result is that (eg) file to file copies are much quicker; typically a copy is now cpu bound.

The final improvement to the disk driver takes advantage of the memory mapping facilities available on the VAX, using scatter/gather to do contiguous i/o on the disk. Several blocks may be picked up in a single read from contiguous locations on the same cylinder. The memory mapping hardware is used to route these blocks to the appropriate buffers, or to 'junk' unwanted intervening blocks. This approach reduces the number of disk operations and hence the associated interrupt-time overheads. A 'tactics' routine is also used to sense the current position within the cylinder and to thus initiate i/o on the nearest possible desired block, minimising the rotational latency.

The oiginal Unix file system had a throughput of about 40Kb/sec. These simple improvements bring the throughput up to about 150Kb/sec. The radical changes in 4.1c bsd allow throughput to reach 300Kb/sec, but at a high cost in cpu time.

Panel Discussion (Piers Lauder, Chris Rowles, Robert Elz)

History of Unix: Started in Bell Labs on PDP 11/40. Early versions called Level 1, 2, 3 etc. Level 5 leaked out, Level 6 was the first widespread release, Level 7 was the last from the research section of Bell Labs. All of these only ran on PDP 11s. A VAX version of Level 7, called Unix 32/V was the starting point for Berkeley and several other educational institutions.

Development of Unix was then taken over by a new group within Bell, and their first release was System III. The latest release is System V (System IV never saw public release). System VI is expected to be released in about a year; it should provide virtual memory and a some other Berkeley type features (eg sockets).

Development of Unix at Berkeley was spurred by an ARPA grant to provide a distributable VAX Unix. This development went on separately from that at Bell Labs; users wanting the Berkeley version must first buy a licence from Western Electric (they own Unix).

Commercial Shortcomings of Unix: No good COBOL, or Database, or FORTRAN. There is no file locking, much less record locking. No Index sequential files. This may all change; '/usr/group' in the USA has proposals for many improvements, there is a demand for such commercial stuff and several groups in the U.S. are working on it. Real-time is NOT usually a problem - Unix is usually fast enough to meet most real-time needs.

Command Naming: The arcane command names of Unix are NOT usually a problem in practice. The user must learn SOME set of command names, the actual names make little difference to the learning time unless they correspond to a set that is already known. But what such set is best? All manufacturers have different terminology. In any case it is easy to change the names of commands in Unix.

AUSAM and MUSH: UNSW and Sydney University collaborated to add features to Unix that were needed in a student environment, viz limits on each user's use of memory, disk, connect time, etc. Basser also uses a 'share scheduler' to dynamically alter response times so users get a pre-determined 'share' of the machine at peak loading. This is part of the secret to gracefully handling more than 80 simultaneous users on a VAX 11/780. The resulting Australian Unix Share Accounting Method (AUSAM) was implemented on Berkeley Unix by Robert Elz. He called it MUSH (Melbourne University SHare).

There is no distributed file system available from Bell or Berkeley, but SUN, Plexus and Microsoft are doing a joint project.

DEC maintenance will support foreign equipment; they are doing it at several universities. Do not let DEC salesmen frighten you.

There is a big demand for Unix hackers in the U.S.

EUUG
European UNIX† Systems User Group

Newsletter Vol 3 No 1
Spring 1983

† UNIX is a Trademark of Bell Laboratories.

The European UNIX Network

Mathematisch Centrum, Amsterdam

Recent years has seen the rise in popularity of the UNIX[1] operating system in universities, private companies and research organisations. The reasons for this are many, one primary reason is the large user community and the ease of interchange of information between them. This is accomplished by an 'ad-hoc' communications network of UNIX systems mainly using dial-up lines[2] which provide *electronic mail*[3] and *network news*[4] facilities. The network is called **USENET** in North America, and **EUNET** is the European extension.

Electronic mail and news

Electronic mail is the ability to post a message to another user, possibly on another computer, in a simple manner. There are facilities for editing messages, sending 'carbon-copies' to other recipients, etc. A user is normally informed when mail has arrived for him.

Network news is a bulletin board shared among many computer systems around the United States, Canada, and now, Europe. This is useful in a number of ways. Someone wishing to announce a new program or product can reach a wide audience. A user can ask "Does anyone have an *x*?" and will usually receive several responses within a few days. Bug reports and their fixes can be made quickly available without the usual overhead of mass mailings. Programs are freely exchanged. Discussions involving many people at different locations can take place without having to get everyone together. The news network has provisions which define the spread of news, and which groups of news are subscribed to. The software has a controlling mechanism for handling the database.

The structure of the network

All things cost money, and networking is no exception. The success of the UNIX network is partly due to it being relatively cheap to join, provided high performance is not sought. Sites, and their financial commitments, can be split up into roughly three groups, *backbone sites*, *secondary feeders*, and *terminal sites*.

A backbone site is one that bends over backwards to make delivery of mail/news as reliable and fast as possible, so it can feed mail/news to smaller sites in the same general area. Backbone sites have a great responsibility and investment in keeping the network running. The hardware required includes modems, auto-diallers, possible expensive connections to other networks, and a considerable amount of machine resource, i.e. computing cycles, disk space, etc. Manpower is needed here, probably at least one person, full time. A backbone site is the channel for all long-distance communication, and the transmission of mail/news can be lengthy and expensive here.

Feeder sites are similar to backbone sites, but only have the responsibility of passing mail/news traffic on to local sites "downstream". The investment here covers hardware in order to be called by or to call other sites, and temporary storage of data which is to be forwarded. The amount of manpower should be low (one man-month per year) as most of the work is done by the backbone site.

Terminal sites involve a minimal investment of money and manpower. Such sites are at the "end of the line", and are not involved in passing traffic on. There is very little manpower investment in a terminal site connection, apart from installation and routine maintenance. The hardware cost is that of a connection for the feeder site.

The extent of the network

At present, the network spans over 1600 sites all over the world with electronic mail, and about 800 of those also participate in the network news. In North America, almost every research institute has a connection to the network. This is the largest such network in the world.

The **Mathematisch Centrum** in Amsterdam started EUNET in early 1982, and is the backbone site for Europe. It regularly calls, by means of dial-up, two sites in North America to exchange

mail/news, then feeds this out to other European sites. Typically, 1M bytes per month are transferred across the Atlantic for mail/news, about one third mail and two thirds news. The amount of news forwarded to Europe is 20% of that which is available in the United States.

In early 1983, one year after its inception, EUNET encompassed around 30 sites; there are over 300 UNIX sites in Europe, so there is still plenty of room for growth.

A map of sites which are formally connected to the network news is included. The map dates from the end of 1982.

Connections to other networks

The UNIX network touches other networks at various points, ARPANET[5] and CSNET[6] in North America, SERC-Net and RCO-Net in Great Britain, and the Australian Computer Science Network[7] for example. Mail, and in some cases news, can be transferred across some of these boundaries. Since the network is a *logical* network which sits on top of physical networks, there is no need for the computer systems to be using UNIX at all, provided the physical networks allow the transfer of logical messages between networks, and each computer system understands the format of the logical messages. This is the case with ARPANET for example.

There are many physical ways in which the mail/news can be delivered, including dial-up lines, X25 networks, ARPANET, private networks, etc. The most popular is by means of dial-up lines, as this is cheap, easy to install, and requires no special hardware or communications media.

Costs

Installation of the network software, like it's running, is almost automatic, and should take one or two hours on a lightly-loaded system. Ideally a site has it's own auto-dialler (cost Hfl 2500.-) to phone other sites. In this way, the site pays automatically for their connection costs. A 300 baud modem costs approximately Hfl 1500.-, but requires the site to be polled.

Backbone sites, such as the Mathematisch Centrum, need more of everything. The only part of this which is currently being passed on to the other sites is their percentage of the local and long-haul communications costs; in the future it may be unavoidable to share the other recurrent costs with all sites connected.

Facilities not provided

The layering of the network, as well as the software, does not allow remote login via a path to a certain site. Of course, local-area networks may provide this, but it is not part of the mail/news network, both facilities are merely using the same transport medium.

Short-term expectations

In it's early stages, EUNET can be expected to follow the same pattern as the North American USENET, spontaneous growth mainly using dial-up lines. However, other computer networks and transport services are now becoming available, and use of these facilities must be made if the network is to remain cost effective as the volume of data increases. This is important as there is no central organisation funding the network, the users do so directly.

At present, two thirds of the European sites are in The Netherlands, where using the public telephone system is relatively cheap. There are two sites in Great Britain which are called up from the Mathematisch Centrum, and these feed other sites by means of private networks, as the public telephone system is expensive there. The public X25 service in Great Britain is relatively cheap, and this will probably play an increasing role as a transport medium there and in the rest of Europe, especially once the international X25 networks become available (end of 1983 in The Netherlands), and the amount of data increases. Some work will need to be done interfacing EUNET sites over these networks.

Future developments

Standards are now available for network protocols, hardware interfaces, and even for the format of electronic mail messages. Many networks, the UNIX network is no exception, have grown up when no such standards applied. There will soon be pressure from various factions to conform.

The present network addressing scheme is a full pathname. An attractive alternative to this would be to use an addressing scheme similar to the surface post, addresses with domains. This change is already under development, and it is interesting to note that this will provide the network with a distributed nameserver, not a central one.

A consequence of the present addressing scheme when news is spread to every subscriber is much duplication of data. If data transport becomes cheaper and faster, a network-wide database may be possible and manageable.

There is investment in existing network hardware and software. Due to

- abovementioned standards
- hardware developments
- growth of local-area networks
- cheaper connection to X25 networks
- technical backup (solving hardware and software problems)
- growth of the network.

Changes will be required in the financial structure of the network. The backbone sites are most vulnerable here, as they are committed to providing a service, the cost of much of which is not passed on to the other sites.

Conclusions

The experience of the last few years with the UNIX network has shown that the national and international communications infrastructure can be considerably improved at reasonable costs. The benefits for the research community are obvious. It is expected, however, that with the growth of the user community of the network, the diversity of this community will also increase. Currently the majority of the UNIX sites in Europe can be found in universities and research institutes, only 30% of the European sites are commercial. whereas in the United States, around 90% of UNIX sites are commercial. Of the European UNIX sites which have access to the network, only 20% are commercial, in the United States, 60% of UNIX sites with access to the network are commercial. Industry is becoming more and more interested in gaining access to the vast amount of information and expertise as embodied by the UNIX network, and the number of users from industry will rapidly increase in the years to come. In this way the UNIX network will play a major role in bridging the gap between researchers in universities and researchers in industry.

1. D. M. Ritchie and K. Thompson, "The UNIX Time-Sharing System," *Comm. Assoc. Comput. Mach.* **17**(7), pp. 365-375 (July 1974).
2. D. A. Nowitz and M. E. Lesk, "A Dial-Up Network of UNIX Systems," UNIX Programmer's Manual, Section **2**, Bell Labs, Murray Hill, New Jersey (August 1978).
3. K. Shoens, "Mail Reference Manual," UNIX Programmer's Manual, Virtual VAX-11 Version, Section **2c**, University of California, Berkeley (November 1980).
4. M. R. Horton, *How to Read the Network News.*
5. J. M. McQuillan and D. C. Walden, "The ARPA Network Design Decisions," *Computer Networks* **1**, pp. 243-289 (August 1977).
6. C. Barney, "CSNET Unites Computer Scientists," *Electronics* (October 20, 1982).
7. R. J. Kummerfeld and P. R. Lauder, "The Sydney UNIX Network," *The Australian Computer Journal* **13**(2), pp. 52-57 (May 1981).

EUUG

European UNIX† Systems User Group

Newsletter Vol 3 No 2
Summer 1983

† UNIX is a Trademark of Bell Laboratories.

Cookbook for setting up a National UNIX systems Users Group

Teus Hagen, EUUG

ABSTRACT

This paper is intended to help you set up a National UNIX systems Users Group. It can not be used as a manual or a set of rules of how the EUUG wants you to do it. If you have your own ideas do not hesitate to use them!
Omissions are due to the shorthand in writing this paper.

6. Why a Users Group

Groups are formed to join together the users of UNIX systems, to join the problems, to join the joy. Of course the group can be used to extend the joy to the vendors for equipment on which UNIX is used. Vendors provide you with the machines and peripherals, and you are their market. It is worthwhile to split the effort in fighting the UNIX beast.

7. What can be organised

National meetings or conferences, UNIX workshops, UNIX introductions, special interest groups, national UNIX network, advisements, publications, exchange of experience and stimulating the UNIX market.

8. Why national

You cannot expect that the UK will understand the problems you have with your national keyboard. You cannot expect that people from Holland will be around all the time to boot your system. But you cannot do it without the other users in Europe! And of course, the EUUG needs you!

9. Why have cover from the EUUG

Some of the needs can be fulfilled better in a union of national users groups, such as international conferences (you have not the money to invite all these people, you have not the time to do it, those people have not the time and money to travel all this time through all these countries), EUUG newsletter, networking, UNIX information service.

The EUUG can (and should) provide you with catalogues as such as available software, UNIX systems, with introductory information and memberlist, because there is always somebody somewhere in Europe with the same trouble as you.

10. EUUG arrangement

Every country should have their own National UNIX Systems Group (NUUG). If needed the country can split that in regional groups as well (RUUG). The RUUG is responsible towards to NUUG, the NUUGs are joined to the EUUG. In order to provide you with the following services, the EUUG needs some money:

- secretary costs (the EUUG has now a professional staff)
- printing and postage costs for the newsletter
- printing and postage cost for EUUG publications (catalog etc.)
- organising the EUUG conferences.

However the EUUG can provide startup support for new national groups as well. (The EUUG should be able to arrange subsidies via the EEC for her work). The EUUG should get enough money from out of the national groups to keep the EUUG running (and not the other way around).

11. Finance for the EUUG

The NUUG should provide enough finances to the EUUG to fulfill her role. Nowhere should be any profit. A yearly arrangement about the money for the EUUG is preferred.

12. NUUG responsibilities

So far the role of the NUUG towards to the EUUG is fixed and explained. The NUUG has pretty much her own constitution, but it should reflect the constitutions of other NUUG's via the EUUG. Preferably the membership arrangements should be roughly the same and no rule should fight other NUUG's or the EUUG.

13. NUUG members

Basically there should be five categories for memberships:

Installation membership (commercial and Academic)
Individual Membership
Associate Membership
Honorary Membership
Vendor Membership

Membership is based on a UNIX Systems license, proof of which must be given to the board (NUUG or EUUG). For AT&T license holders there should be no problem.

14. What about the name

The name: National UNIX Systems Users Group with some prefixing is just about perfect. AT&T will fight you if you have no "Systems" in it. National is a must if you want to play the game.

15. How to arrange a group

Have a national meeting. Do some advertising in national computing papers, but do not think you need to reach everybody, big meetings are a heavy struggle. Get at the meeting a list of people who are interested in joining. And try to get some people who are willing to form a committee. Keep in mind that the same people will be in the national executive board. You need someone for the finance, a secretary and of course a chairman. Arrange a special meeting with some vendors to get them organised as well.

16. What service can you give

Organise UNIX workshops (in some time schedule) to exchange information, to find a backbone site which can be hooked up to the network and arrange a meeting with a representative of the EUUG.

17. Official status

Up until now you are personally responsible for your acts. Your country should have arrangements ready for you to provide you with a legal cover, what you probably need for that is a constitution. Included here is some shorthand for what is in the constitution of the NLUUG (National UNIX Systems Users Group, Netherlands).

18. Questions

If you still have questions, please do not hesitate to call the EUUG secretary.

19. NLUUG constitutions

19.1. Name, date of constitution and location

19.1.1. Goal of the NUUG

The goal of the NLUUG is:

- exchange of information about UNIX
- distribution of software and documentation, as far as is allowed by licenses
- advice on hardware and software for UNIX
- preparation of the UNIX market.

19.1.2. Means:

- yearly (closed) meeting for members
- yearly (open) conference for members, vendors and UNIX interests
- UNIX networking
- EUUG membership
- communication arrangements with vendors and members.

19.2. Time schedule

For ever, starting on the 1st of January ending on the 31st December.

19.3. Membership

Five categories: main membership is installation member based on UNIX System license.

19.3.1. How to start as member

Written request to the board. The board decides with respect to the general membership meeting.

19.3.2. How to end as member

Being legally not a person (the company is dead), by written request for ending, by a decision of the general membership meeting, or by injury of the NUUG or EUUG.

19.4. Finance

Income:

- Yearly payment, fee yearly approved by the general membership meeting.
- Other ways like gifts and other benefits.

19.5. Executive board

Minimum of three persons, voted by the general membership meeting. The board decides who is doing what. The general membership meeting can dismiss an executive board member. Members of the executive board can leave the board after a minimum of 2 months. Yearly there will be one executive board member for (re)-election.

19.6. Responsibility

19.6.1. Representation

The chairman and secretary are allowed to represent the group legally. The executive board can appoint a plenipotentiary.

Compatibility Quiz

Mike O'Carroll

Microsystems Unit, Dept. of Electrical Engineering
The University, Leeds LS2 9JT, England

Herewith a super new quiz for all you 'DEC compatible' fans! Answers, on a postcard please, to the appropriate manufacturer. (N.B. to help those of you who have never gone to another supplier, our answers are included below.)

I. QUESTIONS

Q1. What do you understand by the term 'DEC compatible'?

Q2. What do you understand by the phrase '100%'?

Q3. '[The controller gives] an emulation of all functional features of the DEC RJM02 .... subsystem'. Explain.

Q5. 'Fully supported in this country!' [my exclamation mark]. What is being supported here?

Q6. Are DEC 100% compatible? Think about it.

II. ANSWERS

A1. Certain features, such as the code number of the product, general register layout, etc. are not totally dissimilar from those encountered in the DEC version.

A2. Pick a number in the range from 0% - 90%. We award 90% to our current Emulex controllers which can do most things in a reasonably compatible manner (apart from minor things like multi-sector transfers). Dilog get a slightly lower mark for managing multi-sector transfers, but locking up completely if anything else tries an NPR at the same time. DSD do slightly better now, having fixed the problem whereby the extended address counter went 00, 10, 01, 11, in 'ascending' order.

Xylogics got 5% for an LSI-11 controller which managed to execute the bootstrap, but fell over as soon as it tried to write anything. A Spectralogics controller got 1% as it managed to read in the bootstrap, but failed to execute it. [If you find the last two hard to believe, ask the engineers who came to 'get it working in 10 minutes, squire'.]

A3. '... the maintenance mode features of the DEC RM02 controller is [sic] only partially emulated' - from the same manual as question. Presumably, 'maintenance mode' features are not 'functional' features. See also earlier correspondence in this august publication.

A4. 'The diagnostics marked with an asterisk require certain patches'. See question.

A5. The manufacture of telephone answering machines.

A6. I've given enough away already. Work this one out for yourself.

In order to stave off possible legal action, the author would like to point out that these comments (naturally) relate to our personal experiences with products made by the companies named. However, the points raised above have never been answered by those concerned (see question on support), so here's your chance folks! Incidentally, talking of legal action, here is a supplementary question:

Q7. What, if anything, does the Trades Description Act have to say about terms like 'compatible'?

Extract from 'UKC User's Bulletin No. 123', March 1983

Mike Bayliss

UNIX

1. Mail and News - the Continuing Saga

It seems only reasonable to explain what is happening to the mail and news systems on UNIX, since they have been changing frequently in the last few months.

In the beginning was UNIX, and then there were lots of them, and they talked to each other using something called uucp(1). And then somebody said 'wouldn't it be nice if all our nice mail programs could send mail over uucp'. So they did, but it was badly documented, and nobody else talked like that.

Also in the beginning there was ARPA, and all the ARPA machines talked to each other, and they started sending each other mail. It was documented and was called RFC#733, and was incomprehensible, but it was a *standard.*

Shortly after the beginning there was SERCNET and there was PSS, and all their machines talked and sent mail to each other. Then the JNT spake, saying 'let SERCNET and PSS understand each other, we need a *standard.* And JNT looked at RFC#733 and said 'we will use RFC#733 but it is not good enough for us, we will add things to it, and call it The Grey Book' and SERCNET and PSS talked to each other.

In the meantime there was UKCNET, (although it did not have such a posh name in those days). And all the machines on UKCNET talked (and talked and talked) and somebody said 'let there be *standards*'. And there were, two of them, the EMAS *standard* and the UNIX *standard* and they were incompatible.

Now EMAS was big and powerful and a mainframe and it talked EMAS *standard.* But UNIX was only a small mini-computer and it only talked UUCP *standard.* So the word from on high was 'UNIX will change'. And then EMAS spoke and said 'I am a JNT *standard* mailer, and will only talk to JNT mailers'. So UNIX (and all the other little UNICIES) got together and said 'we must do something' and they did. To talk to EMAS they turned all their mail into JNT mail and to talk to each other they put a JNT mail header in front of UUCP mail. And it sort of worked, but there was a multitude of 'core' files and lost messages.

And then a phone rang and said 'Hello I am Amsterdam, and I want to talk to your UNIX, and so do lots of other people in America'. UNIX said 'that is nice, I am lonely and I like to talk to other UNICIES, do you talk UUCP?' And Amsterdam said 'Yes, talk to me'. But then Amsterdam said 'you cannot call yourself UNIX, there are over 2,000 of us'. So UNIX became UKC, but only when it talked to the World, when it talked to UKCNET it as still UNIX, and UNIX became schizoid (and so did it's programmers).

At the same time, EMAS was told 'you cannot be EMAS, there are two of us, you must change your name'. So EMAS became UKC, but only when it talked to PSS.

But then the World realised there was life beyond UKC, and PSS realised there was life beyond UKC, and they wanted to talk to each other.

But PSS did not want to know about two UKCs and neither did the World. So EMAS and UNIX both said 'I will be UKC', and they fought, and the floor was covered in core dumps and rejected mail.

Then UNIX said 'I am small and flexible, and will make all our machines look like one machine called UKC' and the silence from EMAS was deafening.

But everybody was happy, for in the fullness of time we would be one site and lots of people would talk to each other through us, and UKC would become well known. However, still the UNIX mailer produced core dumps and it was NOT good, and the world could not talk to PSS.

And then the World shook, for ARPA said 'RFC#733 is bad, you will use RFC#822' and the

people who write UNIX mailers said 'we want a *standard*, let us use RFC#822', but they did not produce the code. And then the JNT said 'We will change the Grey Book, but we will not change it', and JNT used RFC#822, but changed parts of it, and said to SERCNET and PSS 'you will change (please)'.

And the UNIX mailer said 'HELP!!!!!' and 'I'm confused'.

So now to the serious part, with apologies for any libel of JNT, ARPA or EMAS.

ARPA now runs RFC#822 mailers, and SERCNET/PSS is changing to new JNT. EMAS is running old JNT but plans to change to new JNT. UNIX runs UUCP, but plans to switch to RFC#822. This is, however, a lot of code, which we intend to obtain from other sites. By the time you read this bulletin, UNIX will be running 'bridging software', (simple, crude and nasty) which converts UUCP and JNT mail formats.

Within a few months UNIX (and COMET, REGI, ROGER) will switch to RFC#822, and EMAS will switch to new JNT. The end result will be better, more *standard* mailers with some nice new features.

How does this affect the news system?

The first version of news we ran here was news2.8, which is being changed to news2.9 at the moment, to clear various bugs and features. In two months we will change to news2.10 which is a drastic overhaul of news2.9 to provide compatibility with RFC#822.

What do all these letters stand for?

UUCP Unix to UniX CoPy
ARPA Advanced Research Projects Agency (American)
SERCNET Science and Engineering Research Council NETwork
PSS Something or other to do with the Post Office Network
JNT Joint Network Team
RFC Request For Comments (seriously!)
UNIX is, of course, a Trademark of Bell Telephone Laboratories Inc.
UNICIES is a plural of UNIX
EMAS might be the trademark of somebody or other.

Second Distribution of Berkeley PDP-11* Software for UNIX**
Release 2.9
(revised June 1983)

A new release of the UNIX system with many enhancements is available from the Computer Science Division of the University of California at Berkeley. It is a complete V7 UNIX system, including the kernel, all standard utilities, and additional Berkeley products. The kernel will run on any PDP-11 with memory management hardware and at least 192K bytes of memory, including the 11/23, 11/24, 11/34, 11/34A, 11/40, 11/44, 11/45, 11/55, and 11/70. It supports most common disks (RK05/06/07, RL01/02, RM02/03/05, RP03/04/05/06, and emulations of these) and tapes (TM02/03, TM11, and TS11). With only a few exceptions (pcc and INGRES), all of the programs in the release will also run on all of the supported machines. The major kernel changes since the 2.8BSD distribution are:

- Process control, a mechanism for stopping and restarting jobs in foreground or background, and the new reliable signal mechanism that supports it. This is nearly identical to the process control facility of 4.1BSD VMUNIX (the Berkeley VAX UNIX system).
- Vfork, a more efficient version of fork.
- Automatic reboots, after crashes or on demand.
- Automatic detection of hardware configuration at boot time, with most of the configuration-dependent addresses and vectors in a single ASCII file.
- Much easier kernel configuration process, with most parameters in one machine description file.
- There are numerous efficiency changes. System overhead has markedly decreased in a number of areas: floating point traps (90% decrease) overlay switches (45% decrease), and system calls (22% decrease).
- There have been many bug fixes. The system is now far more robust.

Other features of the kernel, which were also in the 2.8BSD release, include hashing buffers and inodes, moving buffers and clists out of kernel data space, and the 1K block filesystem. The system supports kernel overlays, allowing it to run on nonseparate I/D machines. It also supports user overlays, so that ex version 3 can be run, even on nonseparate machines!

The Berkeley tty driver is included; it correctly handles erase and kill characters on crt and printing terminals, including correctly backspacing over tabs and control characters.

The enhanced Berkeley implementation of the TCP/IP network facility is included.

Changes to the kernel are conditionally compiled with mnemonic names, making it easy to turn on and off features you decide you do or do not want. This kernel contains contributions from Berkeley's Computer Systems Research Group, the U.S. Geological Survey system, DEC's UNIX Engineering Group, and Tektronix (to mention only a few).

This package also includes the instructional Pascal system, the editor ex, the INGRES database management system, and other software (some of which is described below). Source code, binaries and machine readable versions of all documentation are included. The distribution is provided on two 9-track 800BPI magnetic tapes, one of which is bootable and contains the standalone utilities required to bring up a root filesystem and the kernel. The remainder of the sources, documentation and binaries are in tar format, blocked by a factor of 20 (10240 byte records). Tapes written at 1600BPI are available, as are tapes intended for use on the DEC TS-11 tape drive. We will supply the magnetic tape(s) on which the software will be written. Distributions of the software on disk media are not available. Tp and cpio formats are also not available.

---

*DEC, PDP, and VAX are trademarks of Digital Equipment Corporation. **UNIX is a trademark of Bell Laboratories.

;login:

The USENIX Association Newsletter

Volume 8 Number 5 **November 1983**

CONTENTS

The deadline for submissions for the January issue of *;login:* is December 16

Japan UNIX Users Group Formed

The Japan UNIX Society (JUS) has been formed to support the exchange of technical information among UNIX users in Japan. It plans to hold technical meetings once a month, symposium twice a year, and to publish a newsletter twice a year. JUS may be reached at:

Japanese UNIX Society
c/o Japan Software Development
2-8-10 Toranomon
Minato-ku, Toyko Japan
03-503-4981

USENIX Conference Proceedings Available

San Diego Conference

Copies of the proceedings of the San Diego UNICOM conference are still available from the Software Tools Users Group. They are over 350 pages long and include all papers presented by the speakers as well as reports on many of the presentations.

The price is $25 per copy, plus $10 per copy for overseas postage. Send your check or money order made out to "Software Tools Users Group" to:

STUG
1259 El Camino Real, #242
Menlo Park, CA 94025

Toronto Conference

Copies of the Toronto Proceedings are available for purchase from the USENIX office. The price is $30 per copy, plus $15 per copy for overseas postage. Payment **must** accompany your order.

;login:

The USENIX Association Newsletter

Volume 8 Number 6 **December 1983**

CONTENTS

The deadline for submissions for the March issue of *;login:* is February 15

The Evolution of the Berkeley UNIX Project

[Received from the Computer Systems Research Group, U.C. Berkeley]

The distribution of Berkeley UNIX 4.2BSD to licensed installations began on September 30, 1983, and is proceeding quite smoothly. In the meantime, the Computer Systems Research Group of the University of California at Berkeley (CSRG) has started working on four new research projects, in addition to further tuning and refining the facilities of 4.2BSD. As of August 1, CSRG has been headed by Mike Karels, who previously worked with the Berkeley PDP-11 distribution. He is working under the guidance of Prof. Domenico Ferrari while Prof. Robert S. Fabry is on sabbatical this year. Plans have been prepared for the next three years in each of the four areas. The projects will be supported by a new three-year contract from the Defense Advanced Research Projects Agency.

The first project will study various issues related to the design of mechanisms for distributed access of files and other resources in a network of single-user workstations running under Berkeley UNIX. The issues include the performance effects of the amount and use of local storage present in each workstation, the design of flow control policies to prevent saturation of such remote facilities as file servers, and the tradeoffs between autonomy and transparency in accessing distributed objects.

Determining the cost-performance impact of several decisions to be made when designing a distributed name server is the main goal of the second project. Various lookup algorithms, local caching, and cache validation policies will be investigated. A Berkeley UNIX-based Name Domain server for the DARPA Internet is being developed to run experiments and to provide parameters for the modeling part of the study.

The third project is concerned with evaluating various metrics and policies for automatic load balancing in distributed Berkeley UNIX systems. All the configurations being considered are based on a local-area network, and include single-user workstations as well as multiple-user interactive machines. The goal of this effort is to design and implement a viable load balancing scheme that can be tuned to the characteristics of different environments and host configurations.

The researchers involved in the fourth project are designing a distributed measurement instrument and a distributed program debugger. The two problems are being attacked together because of their common need for remote process control functions. The measurement instrument should, among other capabilities, allow the experimenters to create and sustain an artificial load on a distributed system, to control remote software monitors, to capture inter-process communications, and to keep the clocks of the various hosts on a local-area network in as full synchronization as possible. The debugger should allow programmers to control the state of their distributed applications, and verify the correctness of their assumptions about synchronization and event ordering.

The addition to 4.2BSD of sub-network routing, allowing logical and physical networks (or external and internal addressing) to be different, and the revision of the terminal line disciplines to make them more consistent with the network interface, thereby improving remote terminal facilities, are, among the other projects being considered, the most likely to be undertaken in the near future.

At this time, there are no specific plans for future releases of 4BSD; as the system evolves, the additions will be incorporated into a new distribution when appropriate.

Schedule of USENIX Technical Sessions at UniForum

Wednesday, January 18

1:30– 3:00 **Networks – Aspects of Networking under UNIX**
Chair: Thomas Ferrin, University of California at San Francisco

1:30– 1:55 Driver-Based Protocol Implementations
Greg Chesson, Silicon Graphics

1:55– 2:20 The Excelan TCP/IP Protocol Package
Bruce Borden, Silicon Graphics, Inc.
Bill Northlich, Northlich Computer Systems & Software, Inc.

2:20– 2:40 Worknet: A Xenix-Based Merged Filesystem Network
Alan Greenspan, Altos Computer Systems

2:40– 3:00 CSNET Grows Up
Michael T. O'Brien, The Rand Corporation
Daniel B. Long, Bolt Beranek and Newman, Inc.

3:00– 3:30 BREAK

3:30– 5:00 **Distributed UNIX – Multiprocessing under UNIX**
Chair: Alan Nemeth, Prime Computer Inc.

3:30– 3:50 Software Administration over Computer Networks – The exptools Experience
Joseph L. Steffan, Bell Laboratories

3:50– 4:10 A Distributed File System for UNIX
Matthew S. Hecht, John R. Levine & Justin Walker,
Interactive Systems Corp.

4:10– 4:30 Multiprocessor Debugging on a Shared Memory System
Chet Britten & Paul Chen, Metheus Corporation

4:30– 5:00 Panel Discussion – Multiprocessing Issues

5:00– 7:00 DINNER

7:00– 8:30 C Language Evolution & Standardization
L. Rosler, Bell Laboratories

Thursday, January 19

8:30–10:00 **Compilers and Languages**
New Languages and Developments in Familiar Languages
Chair: Louis Salkind, New York University

8:30– 9:00 The Evolution of the Portable C Compiler
S. C. Johnson, AT&T Bell Laboratories

9:00– 9:20 How to Feel Better about Knowing It is Written in C
Alan R. Feuer, Catalytix Corporation

9:20– 9:40 An Implementation of The B Programming Language
Lambert Meertens & Steven Pemberton,
Centrum voor Wiskunde en Informatica

9:40–10:00 Object-oriented Programming in C Language
Brad J. Cox, Productivity Products, Inc.

10:00–10:30 BREAK

10:30–12:00 **Over Under Sideways Down, Backwards Forwards Square & Round UNIX Directions**
Chair: Brian E. Redman, Central Services Organization
Bell Operating Companies

10:30–10:45 Behind Every Binary License is the UNIX Heritage
B. E. Redman, P. E. Parseghian, Bell Laboratories

10:45–11:00 How did UNIX Get Here?
A Sketchy History of the Politics of UNIX Development
Andrew Tannenbaum, MASSCOMP

11:00–11:15 A Proposed Syntax Standard for UNIX System Commands
Kathleen Hemenway & Helen Armitage, Bell Laboratories

11:15–11:30 Decision Avoidance; UNIX from DEC, Finally
Armando Stettner, Digital Equipment Corporation

11:30–12:00 Panel Discussion

12:00– 1:30 LUNCH

1:30– 3:00 **Applications**
Chair: Reidar Bornholdt, Columbia University

1:30– 1:45 MLE (Multi-Lingual Editor)
Scott Bradner, Harvard University

1:45– 2:05 MPS: A UNIX-Based Microcomputer Message Switching System
T. Scott Pyne & Joseph S. D. Yao, Hadron Inc.

2:05– 2:20 Taming The Beast (An RSX Emulator for UNIX)
Daniel R. Strick, University of Pittsburgh

2:20– 2:40 A UNIX-Based Color Graphics Workstation
Rex McDowell, Metheus Corporation

2:40– 3:00 User-Level Window Managers for UNIX
Robert J. K. Jacob, Naval Research Laboratory

3:00– 3:30 BREAK

3:30– 5:00 **Implementations I**
Chair: Michael O'Dell, Group L

3:30– 3:50 The UNIX Paging System
Keith Kelleman, Bell Laboratories

3:50– 4:05 Providing a Job Control Facility in UNIX System V
W. P. Weber Jr. & L. S. Weisbrot, Bell Laboratories

4:05– 4:20 Loadable Virtual Disk Device Driver and Server
Thomas A. Alborough, Creare R&D

4:20– 4:45 OSx: Towards a Single UNIX System for Superminis
Ross A. Bott, Pyramid Technology Inc.

4:45– 5:00 New ½ Inch Tape Options and Trade-Offs for 4BSD on DEC VAX Processors
Robert J. Kridle, mt xinu, inc.

5:00– 7:00 Meeting of the USENIX Board with the Members

Friday, January 20

8:30–10:00 **Implementations II**
Chair: Joseph S. D. Yao, Hadron, Inc.

8:30– 8:50 A Layered Implementation of the UNIX Kernel on the HP9000 Series 500 Computer
Jeff Lindberg, Hewlett-Packard

8:50– 9:05 Porting Xenix to the Unmapped 8086
John Bruno Hare & Dean Thomas, The Santa Cruz Operation, Inc.

9:05– 9:25 Writing Device Drivers for Xenix Systems
Jean McNamara, Paresh Vaish & Richard N. Bryant, Intel Corporation

9:25– 9:45 Transparent Implementation of Shared Libraries
Curtis Downing, Bunker Ramo Information Systems
Frank Farance, Systems Theory Design Corporation

9:45–10:00 UNIX File Systems Optimization on Microcomputer Systems
David Robboy, Intel Corporation

10:00–10:30 BREAK

10:30–12:00 **Communications – UNIX Talks to Itself and Others**
Chair: Mark Horton, Bell Laboratories

10:30–10:45 A UNIX to VAX/VMS Communications Link
Clifford N. Cary, Creare R&D

10:45–11:00 Loving UUCP, or How I Spent My Summer Vacation
P. Honeyman, D. A. Nowitz, B. E. Redman, Bell Laboratories

11:00–11:15 New Implementations of UUCP
D. A. Nowitz, P. Honeyman, B. E. Redman, Bell Laboratories

11:15–11:30 Mapping the UUCP Network
Rob Kolstad & Karen Summers-Horton, Bell Laboratories

11:30–12:00 UUCP-USENET Panel – UUCP/USENET issues

12:00– 1:30 LUNCH

1:30– 3:00 **Graphics Under UNIX**
Chair: Noel Kropf, Columbia University

1:30– 2:00 UNIX as a Development Base for High Performance Graphics Applications
Thomas Ferrin, University of California at San Francisco

2:00– 2:20 DAPS and GSDL: A Procedural Approach to Graphics
Christos Tountas, Graphics Information Systems Technology Inc.

2:20– 2:40 The Design of a Dedicated Graphics Subsystem for a UNIX Machine
Jack Burness, MASSCOMP

2:40– 3:00 Image Processing Work Station
Dale Hensley, TRW

3:00– 3:30 BREAK

3:30– 5:00 **Real-Time Under UNIX**
Chair: Joseph Germann, Sky Computers

3:30– 3:50 Life with UNIX in Real-Time
Steven Polyak, Contel Information Systems

3:50– 4:10 Real-Time Extensions to the UNIX Operating System
Bryon Look & Gary Ho, Hewlett-Packard

4:10– 4:30 UNIX Optimization – UNIX/68000/SKYFFP
Joseph Germann, Sky Computers

4:30– 4:45 Integrating a Peripheral Floating-Point Processor into UNIX
Eryk Vershen, UniSoft Systems

Abstracts for USENIX Technical Sessions at UniForum

Networking

1:30–3:00 Wednesday

Driver-Based Protocol Implementations

Greg Chesson

Silicon Graphics
630 Clyde Court
Mountain View, CA 94043
(415) 960-1980

Network implementations under UNIX tend to add layers, primitives, system calls and other software gadgets to the operating system. The result is seldom, possibly never, pleasing. Consider the problem of providing incoming virtual terminal terminations in the operating system. A common technique involves the use of pseudo-typewriters, extra processes, and lots of data movement with no particular performance or complexity advantages. Dennis Ritchie's elegant implementation of kernel coroutines has good, but not spectacular performance, and some complexity. It compares favorably with other approaches, especially pseudo-typewriters.

However, one might argue that all of these software techniques will soon be rendered obsolete by new-generation communications interfaces that provide on-board transport protocols. If we postulate a network device that is well suited to both character and block i/o by virtue of its firmware, the corresponding device driver might look like a combination of a dz/dh and a simple disk driver, e.g. the ancient rf. The main attraction here is that the network would no longer be an alien creation, but would more closely resemble the i/o model that the system prefers.

Since these network devices are not quite yet available, it is interesting as well as useful to investigate software techniques that work with existing devices. The approach used at Silicon Graphics consists of adding a protocol module to an otherwise raw device driver. The resulting driver looks like a character device to the typewriter portion of the system and looks like a block device to block portion of the system. The system "sees" reliable data streams and is not aware that a protocol implementation is lurking in the driver. This means that virtual terminals hook into the operating system at the same place as local terminals and have identical modes and character-processing semantics. The block device interface is equally well situated for representing remote files.

An implementation of Xerox's Sequenced Packet Protocol has been tested in various operating systems: Version 7, System V, 4.1, 4.2, and VMS. In each of these cases the protocol part changes very little. The real differences are in the device driver interfaces.

The presentation will focus on the architecture and implementation, and as time permits tuning and debugging issues.

The Excelan TCP/IP Protocol Package

Bruce Borden

Silicon Graphics, Inc.
630 Clyde Court
Mountain View, CA 94043
(415) 960-1980

Bill Northlich

Northlich Computer System & Software
21 Newell Court
Walnut Creek, CA 94595
(415) 938-9713

The Berkeley/BBN IP/TCP internetwork protocol code was ported to an intelligent front-end Ethernet board. The EXOS/101 Ethernet Front-End Processor from Excelan is a Multibus DMA board with an 8MHz 8088 and 128KBytes of ram memory. By porting the protocol code onto the front-end, the host overhead was reduced drastically, and the kernel-resident protocol software was reduced to a small device driver. The implementation mimics the Berkeley socket interface utilizing pseudo-devices and ioctl calls, thus allowing most of the existing network user code to run without modification.

The prom-resident operating system on the EXOS/101 board provides support for multiple processes and communication via message queues. The initial implementation maintains one process per TCP connection, which had to be rewritten to properly handle multiple forks reading and writing the same connection. The current implementation maintains one process which knows how to save its state when "sleep" is called, and to resume a blocked incarnation when "wakeup" is issued. Bothe of these approaches were possible without any changes to the Berkeley code, only to the interface and support routines.

The host to front-end protocol was designed to place most of the burden on the front-end, and to be completely host operating system independent.

This talk will describe the port to the front-end, and will examine the desirability of this approach for future network protocol implementations.

Worknet: A Xenix based merged filesystem network

Alan Greenspan

Altos Computer Systems
2641 Orchard Park Way
San Jose, CA 95134
(408) 946 - 6700
ucbvax!microso!altos86!alan

Worknet is a merged filesystem network of Altos microcomputers running Xenix. All machines on the network are logically connected by a "super-root" directory which is used to generate pathnames for accessing files network wide. This approach allows the sharing of disk resources and peripherals such as tape drives and printers while maintaining a user interface which is essentially transparent. Running processes on remote CPUs is supported and includes the ability to pipe together processes running on different machines. Support for diskless workstations is provided by including a network pseudo-disk

driver for pipes and swapping. Protections are extended across the network at login time by verifying passwords and special considerations are given to "setuid" programs.

The implementation involves changes only to the Xenix kernel and the addition of user mode fileserver processes. This allows applications to access the network without re-compilation or re-linking. At the transport level both Ethernet and RS-422 are supported with protocols compatible with the ISO proposed standard.

CSNET Grows Up

Michael T. O'Brien

The Rand Corporation
1700 Main St.
Santa Monica, CA 90406
Net: obrien@rand-unix

Daniel B. Long

Bolt Beranek and Newman Inc.
10 Moulton St.
Cambridge, MA 02238
Net: long@bbn-unix

The Computer Science Research network (CSNET) is undergoing a transition from a development phase to a member-supported utility. In this paper, we describe the many changes that will result from this transition. Work is proceeding in several areas. Among these are:

New Network Software. Several CSNET members are in the process of becoming CSNET/Telenet sites using the CSNET-developed IP-to-X.25 interface. This software/hardware facility allows a site to connect to the Telenet public packet-switch network, and to use it to communicate with other hosts on both Arpanet and Telenet using the DOD IP/TCP protocols encapsulated in X.25 packets. This development gives member sites who use it the ability to treat Telenet as a type of public Arpanet.

New Mail Software. CSNET is committed to tracking Berkeley UNIX release 4.2BSD and the BBN TCP/IP implementation. Transporting and testing the software will be completed in early 1984. Phonenet software will be available for V7, 4.1BSD, 4.2BSD, and (using member-contributed software) VMS and other systems with Pascal. CSNET/Telenet and Nameserver software will be available for 4.2BSD only. A major rewrite of MMDF, the CSNET standard mail transport system, fully supports address standard RFC 822 and has operational and performance improvements in several other areas. In addition, member-contributed facilities exist for connecting MMDF to UUCP and managing mailing lists.

Mail Gateways. Negotiations are under way to connect CSNET with several European and domestic networks. There are already several affiliate members in Canada in the process of connecting to CSNET. An organization in Sweden is soon to become a gateway to the Swedish University Network using the CSNET/Telenet facility. In addition, a CSNET/BITNET gateway will be operational by the end of 1983.

Distributed UNIX

3:30—5:00 Wednesday

Software Administration over Computer Networks
The Exptools Experience

Joseph L. Steffen

Bell Laboratories
Naperville, Illinois 60566
(312) 979-5381
harpo!ihnp4!ihu1h!steffen

This abstract describes a working method for large-scale, distributed administration of software over computer networks. It is being used for the experimental tool package (exptools) at Bell Labs, which is a collection of the latest programming and text processing tools supported by the authors or other individuals rather than an official organization. It contains more than 60 tools maintained by over 20 people on more than 100 machines of four types connected by two different networks. There is an overall coordinator who controls the installation of new tools to insure a uniform tool set across all machines; but the individual tool supporters actually maintain the tools, that is, fix bugs and install new releases.

The novel features of *exptools* administration are:

The division of software administration between overall coordination and individual tool support.

The use of existing computer network capabilities (file transfer and remote command execution) to install and update tools.

The latter avoids manually logging into each machine, which has several benefits:

A tool can be updated on all machines in hours instead of days or weeks.

Adding more machines does not significantly increase administrative effort.

Tools can be supported by people or organizations outside the computer center because the coordinator has final control.

Work done by the coordinator is minimized, which allows one person to administer scores of machines.

Tool administration can be separated from machine administration, i.e., the local system administrator does not install or update tools.

A Distributed File System for UNIX

Matthew S. Hecht
John R. Levine
Justin C. Walker

INTERACTIVE Systems Corporation
8689 Grovemont Circle
Gaithersburg, Maryland 20877
(202) 789-1155
allegra!ima!matthew

and

441 Stuart Street
Boston, Massachusetts 02116

We describe a design for achieving user-level transparent access to remote files in a local area network of UNIX systems. This design features (1) a *remote mount* system call that allows the mounting of a remote directory, and (2) a specialized *remote function call* mechanism that allows one UNIX kernel to call functions in another. The talk focuses on design problems that arise and the solutions that we chose.

The problem we solve here is how to make access to local and remote UNIX files indistinguishable to users; that is, to users the syntax and semantics of local and remote file access are identical. For example, the user of a command like

cp~~x~~y

where x and y are arbitrary pathnames, can be oblivious to the location of files x and y (one or both may be local or remote) since the underlying UNIX system calls do the right things in either case.

Transparent access to remote files in a local area network of UNIX systems provides new opportunities for file sharing. Independent UNIX systems can share files, diskless workstations can obtain file service from another UNIX system, and workstations with low capacity disks can share databases. Transparent access also allows files to be relocated without breaking programs.

Our solution features a remote mount system call, and makes a cut at the i-node function interface. This work contains a novel mix of implementation ideas, yielding a simple, clean, and practical solution. Instead of assigning a remote file-server process to a local user process, we use a pool of kernel file-server processes that feed from a common request queue. In addition, we use a datagram-based, specialized remote function call mechanism that draws ideas from work by Nelson [3].

Related extant work on distributed file systems for UNIX is extensive and growing; we comment on a few related papers here. Our work is similar to that of Luderer and others [2] at Bell Labs on S-UNIX/F-UNIX and to a design of Plexus Computers described by Groff [1]. However, these papers indicate a different process architecture with communication based on virtual circuits. The LOCUS project of Popek and others [4, 6] at UCLA is more ambitious; we do not consider replicated files nor transactions. The COCANET project of Rowe and Birman [5] at UC Berkeley is also more ambitious than our work; we do not handle remote processes.

1. Groff, J.R., "Modified UNIX System Tames Network Architecture," *Electronics*, pp. 159-163 (September 22, 1983).

2. Luderer, G.W.R., *et al.*, "A Distributed UNIX System based on a Virtual Circuit Switch," *Proc. 8th Symposium on Operating Systems Principles*, Pacific Grove, Calif., pp. 160-168 (December 1981).

3. Nelson, B.J., "Remote Procedure Call," report number CSL-81-9, Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, Calif. (May 1981).

4. Popek, G., *et al.*, *Proc. 8th Symposium on Operating Systems Principles*, Pacific Grove, Calif., pp. 169-177 (December 1981).

5. Rowe, L.A., and Birman, K.P., "A Local Network Based on the UNIX Operating System," *IEEE Trans. on Software Engr.*, Vol. SE-8, No. 2, pp. 137-146 (March 1982).

6. Walker, B., *et al.*, "The LOCUS Distributed Operating System," *Proc. 9th Symposium on Operating Systems Principles*, Bretton Woods, New Hampshire, pp. 49-70 (October 1983).

Multiprocessor Debugging On A Shared Memory System

Chet Britten
Paul Chen

Metheus Corporation
5289 NE Elam Young Parkway
Hillsboro, Oregon 97123
decvax!teklabs!ogcvax!metheus!chet

This paper discusses a simple debugging technique used in developing a shared memory multiprocessor 68000 UNIX operating system. Techniques applicable to multiprocessor debugging in general, and those dependent on specific hardware are presented. Some of the problems and trade offs are also covered.

In developing the operating system for the Metheus Lambda 750 VLSI design workstation, we first developed several tools for debugging the kernel. Most of the ideas offered could be used by any implementation. Hardware design considerations to make debugging easier are also discussed.

The debugger we used allows us to run either of two processors individually or concurrently. Once the task of allowing breakpoints and tracing is implemented, capabilities can easily be added. The host debugger running on the VAX allows us to see symbolic C stack traces on either processor, set multiple breakpoints on either processor, and examine memory either in different radices or as 68000 instructions.

We have a debug monitor running on one of the processors. This processor saves its own registers on a trace or breakpoint trap, then starts executing the monitor code. The only thing the other processor has to do, on a breakpoint or trace trap, is to dump its registers, indicate it has done so, wait to be told to proceed, then restore its registers and resume execution. While the second processor is waiting the debug monitor can examine or change any of the processor registers, set or remove breakpoints, or set or clear the trace bit.

Having a multiprocessor debugger for the kernel has much more than paid for the effort to implement it.

Languages and Compilers

8:30—10:00 Thursday

The Evolution of the C Language and the Portable C Compiler

S. C. Johnson and L. Rosler
AT&T Bell Laboratories
Murray Hill, New Jersey

The C language has continued to evolve since the publication of the Kernighan and Ritchie book in 1978 established a de facto standard. Among the enhancements to date are enumeration data types, a *void* type for functions that return no value, long (more than eight-character) identifiers, and expanded semantics for structures and unions.

Current directions of C evolution include: the declaration of the types of function arguments and the coercion of actual parameters; the ability to specify explicit constants (variables or aggregates whose values cannot change during execution, and which can therefore be compiled into program text space or into read-only memory); and assembly-language windows that allow access to C variables by name. The talk will describe how these developments are taking place at the same time that the language is being standardized by ANSI and IEEE committees.

During the same time period, the compiler technology that developed the Portable C Compiler (which was used to provide more than 30 production compilers on different machines) has evolved into a next generation of compilers called PCC2. These offer an easier porting process and improved maintainability, based on the use of a description of the target machine to produce the code-generation templates, which support a wider variety of machine features.

The talk will touch briefly on other experiments based on C, including the extension of C to support abstract data types ("classes") with operator overloading, and a dialect of C that compiles to silicon integrated circuits.

How to Feel Better about Knowing It is Written in C

Alan R. Feuer
Catalytix Corporation
Cambridge, Massachusetts

C is a controversial programming language praised for its versatility and portability, and criticized for its low-level nature. Even though it ignores many of the adages that good languages should follow, it has become the language of choice for more and more organizations.

Today C is used in environments and for applications far different from those for which it was originally designed. While its spread is heartening to those of us who feel that C is a good tool for producing software, there is one significant aspect of C that we should not ignore: C is a dangerous language.

There are many ways in which the unknowing programmer can fall prey to C's pitfalls: by marching off the end of an array, by referencing through a dangling pointer, by passing the wrong kind of arguments to a function, by forgetting a vital pair of parentheses. In none of these instances is the language much help in finding the error.

But C is not hopelessly perilous. This paper discusses some of what can be done to make C a safer language and introduces an important new tool for developing reliable software in C.

An Implementation of the B Programming Language

Lambert Meertens
Steven Pemberton

Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam

The B programming language has been designed for use in personal computing. B is a fairly conventional language, but for one thing: the design has fully concentrated on ease of learning and use.

B offers all the advantages of strong typing found in languages such as Pascal. Two nice data types of B are the *list* (a bag or collection of items of one type) and the *table* (an array with indexes of any one type and stored values of any one type). B's lists and tables are fully dynamic. The number of types in B is small: 5, and they may easily be combined to implement other types.

Unlike most typed languages, B has no declarations, the types being inferred from context.

Apart from sheer exhaustion of memory, B does not allow limits to be imposed by the implementation. So identifiers may have any length, numbers may have any magnitude, a list may contain any number of items, and so on.

To support top-down programming B has *refinements*, which behave like parameterless light-weight procedures.

Indentation is used to indicate nesting. This obviates **begin** ... **end**. It also prevents confusion due to contradiction between indentation and program structure.

Global variables are permanent in B and values such as tables may be extended at will. Therefore, there is no need for an extra file concept and the bother of special file handling commands.

For the language to be available for the intended group of users it must be implemented on relatively cheap computers. However, B is not suitable for tiny computers like 8K 8 bit micro-computers. This would have been 'designing for the past'. Systems that have the capacity needed for a smooth B implementation are just now entering the top segment of the personal-computer market.

A pilot implementation of B has been in operation at the CWI since May 1981, running on a VAX and a PDP 11/45. As a pilot, this version was not intended for general release. A new implementation has just been completed. The new version is more portable. It is coded, like its predecessor, in C. The new version is also faster: the source text of programs is parsed before execution, and the data types of B are implemented in an efficient manner. Also, a B-dedicated editor is included.

Objective-C
Object-oriented Programming in C language

Brad J. Cox

Productivity Products Int.
37 High Rock Road
Sandy Hook, Connecticut 06482
(203) 426 1875

Objective-C is a C pre-processor that adds Smalltalk-80 classes, objects, messages, encapsulation and inheritance to the C programmer's toolkit. No C language capabilities are eliminated, and none are changed. Objective-C allows the programmer to choose between conventional C language tools when efficiency and portability are paramount, and object-oriented power-tools when encapsulation and inheritancy are needed for reducing code bulk and complexity. Objective-C is an enhancement to C language, not a stripped-down Smalltalk. It is a production tool for professional C programmers who have determined that C language and the programming style that goes with it fits their problem domain.

UNIX Directions

10:30–12:00 Thursday

Behind Every Binary License is the UNIX Heritage

B. E. Redman
P. E. Parseghian

Bell Laboratories

Lately there seems to be much pessimism in some quarters about the future of the UNIX system. Many who have watched its development from the earliest days feel that the system grows only more corrupted and is steadily declining into mediocrity. These issues are addressed in this presentation, with particular attention to motivations, costs, and benefits.

UNIX was originally designed by a talented fraternity with a clear and common vision for a better computing environment. Now the design is controlled by powers with very different goals in mind (e.g., commercial). Although this influence is not directed toward preserving the "purity" of UNIX in some higher sense, it does not necessarily follow that the system is thus "perverted." UNIX has evolved from a simple, elegant model into one that is certainly complex and often appears downright haphazard. It no longer constitutes a statement of smallness, but appears to be suffering a period of unbounded growth. It is generally accepted that the original systems provided a rich environment for a community of sophisticated computer users, as was intended. More recently it seems that UNIX is touted as a computing panacea, and the compromises that have increased its palatability (indeed, popularity) have reduced its effectiveness. Perhaps the most frightening development is the threat that the "total system" (including source code) will no longer be distributed, or will be available only at prohibitively high costs. Is it legitimate to call such a system "UNIX"?

The term "UNIX" has come to represent more than an operating system or computing environment; it represents philosophies about computing. Although the costs seem high and the motivations impure, we must recognize that an important benefit has been realized: the UNIX philosophy has been spread

among the computing masses and has influenced the direction of computing. The commercialization of UNIX is responsible for this. Other systems may have been just as revolutionary, but will never have a similar impact because they were kept private.

How did UNIX get here?
A Sketchy History of the Politics of UNIX Development

Andrew Tannenbaum
MASSCOMP

One way in which UNIX differs from other operating systems is the environment in which it was developed. On one side, UNIX was developed by various groups within Bell Laboratories, in a company which was restricted by consent decree to produce telecommunications products. On the other side, UNIX was leaked to universities, where hackers took it upon themselves to hack UNIX up good.

This paper will discuss the trials and tribulations of UNIX's youth so that folks might better understand why UNIX grew up the way it did.

A Syntax Standard for UNIX Systems Commands

Kathleen Hemenway
AT&T Bell Laboratories, Murray Hill, NJ

Helene Armitage
AT&T Bell Laboratories, Piscataway,NJ

The syntax of UNIX System commands has led some to criticize the system's user interface. The syntax has been criticized for being inconsistent -- there are subtle variations among commands. For example, while some commands allow more than one argument to follow a single delimiter, other commands don't: *ls -l -t* may be abbreviated to *ls -lt*, but *lpstat -d -r* may not be abbreviated to *lpstat -dr*. Similarly, the significance of white space between arguments varies among commands: Some commands require a space before arguments to options, whereas others don't allow space, and in others space is optional. *sort -o file* and *sort -ofile* are synonyms, while *cut -c list* and *cut -clist* are not. These variations frustrate users when they are learning new commands, and the variations cause users to depend on the manual even for frequently used commands.

Inconsistencies aside, the syntax itself has been criticized. Specifically, the use of single letter options has been questioned. Some critics assert that using a single letter takes terseness too far. They argue that because a given letter typically stands for many different words across commands (e.g., the option "-c" stands for column, character, copy, etc.), it is difficult to learn and remember options. Also, the use of delimiters for options has been questioned. While a few commands use "+" to delimit options that add features, most commands use "-" to delimit all options. Since the hyphen is frequently conceptualized as a minus sign, this usage often seems incongruent.

This paper reports a project that addressed these issues. The goal of the project was to identify and evaluate shortcomings in the command syntax and to identify a syntax standard. The standard is to be used in new commands and as a basis for retrofitting the syntax of existing commands. The standard is intended to exert a constructive influence on the evolution of the command set by establishing a

template toward which the command set should evolve.

Decision Avoidance; UNIX from DEC, Finally

Armando Stettner
Digital Equipment Corp.
Continental Blvd. 11
Merrimack, NH 03054

Applications

1:30–3:00 Thursday

MLE (Multi-Lingual Editor)

Scott Bradner
Harvard University

A multi-lingual text editor is being developed at the Computer Based Laboratory, Harvard University. It contains non-ASCII characters necessary for a number of European languages: umlauts, grave and acute accents, polish 'L' and 'l' among others. Greek, modern and Classical, is available with full diacriticals. Hebrew, Russian, Coptic and Arabic are being developed. Provisions have been made for languages (such as Arabic) with more cases than just upper and lower.

One key will allow the user to reverse the direction in which the cursor moves, and searches are prompted for, so that a keyboardist can input individual Hebrew words in English text with the cursor moving from left to right, but could also type extended text with cursor moving from right to left. The user will be able to define the all the keys of the keyboard according to taste so that typists accustomed to a specific keyboard will not have to change their habits.

MPS: A UNIX-Based Microcomputer Message Switching System

T. Scott Pyne
Joseph S. D. Yao
Hadron, Inc.
6th Floor
1945 Gallows Road
Vienna, VA 22180
(703) 790-1840

A message switching system which operates under the Xenix variant of the UNIX operating system has been developed by a Hadron team including the authors for use in a law enforcement environment. The system interfaces to Motorola mobile (in-vehicle) digital computer terminals via a radio link and to remote IBM mainframe-class systems via emulation of either 3780 or 3270 protocols. The package

allows personnel in the field to use the mobile terminals to make inquiries (such as wanted person and/or stolen car checks) directly to state- and national-level data bases such as the FBI's National Crime Information Center, and to forward text messages from unit to unit, or between a unit and the station, in situations where interference, ambiguity, or interception are potential problems.

The project is unique in several ways. We believe it to be the first such implementation to be written in a higher-level language, the first under a general-purpose operating system, and the first on an inexpensive microcomputer. Previous packages of this type were written in assembly language, implemented under proprietary real-time operating systems, and hosted on more expensive minicomputer-class machines. This package is of course written, as the proverb goes, "95% in C."

Taming the Beast (An RSX Emulator for UNIX)

Daniel R. Strick

The notion of simulating the program execution environment of foreign operating systems is not new to UNIX. The programs that maintain these exotic environments are usually called "emulators", probably because the term was used with the RSTS operating system to describe run-time systems that supported the execution of programs built to run on the other DEC PDP11 operating systems, RT11 and RSX11. Since UNIX was originally developed for PDP11s and there was a great deal of software written for the DEC operating systems, it was only natural that emulators would be developed to support the execution of this software in the UNIX environment.

Several RT11 emulators were written, but RSTS and RSX were left alone. RSTS was not attractive because most non-BASIC RSTS programs were run in RT11 emulation mode and RT11 was so much easier to emulate. RSX was probably not attractive because it was perceived primarily as a base for real time applications which could not be supported in a UNIX environment, RSX was much more complicated than RT11, the RSX system calls were not completely documented, and the RSX user interface was unusually ugly.

Time has passed since the first emulators were written. RSTS is nearly extinct, RT11 remains basically a single user system, and RSX has emerged as DEC's candidate for a general purpose operating system for the PDP11. The RSX user interface has improved slightly, but the best place to develop RSX software would be on a UNIX system. An RSX emulator for UNIX would make this easier.

This presentation describes some of the problems encountered when developing an RSX emulator for UNIX. These problems are interesting because they arise from significant incompatibilities between RSX and UNIX and therefore reflect basic differences in design philosophies. One way to learn about UNIX is to study what it is not.

UNIX Based Color Graphics Workstation

Rex Mc Dowell

Metheus Corporation
Hillsboro, OR

This paper will explain the color graphics implementation of the Metheus Lambda 750 workstation, which integrates Berkeley 4.1 UNIX with high performance color graphics. With color windows and multiple font sizes and styles, a user can customize his UNIX working environment, as well as supplying a powerful graphics workstation.

The Lambda hardware simplified the implementation. A new graphics driver was added, and only minor modifications of the tty driver was made. The Lambda Workstation was designed to support our VLSI Cad software, which needed fast, interactive graphics. Our UNIX runs on 2 12MHZ 68000's, one actually running UNIX, the other is an IO processor, with its own real time kernel. The graphics engine contains another 12MHZ 68000, acting as a display list manager and transformation processor. This 68000 feeds a 2901 bit slice, which does the actual drawing. The bit slice also does clipping, and can draw into 8 bit planes at 4 million pixels a second. The graphics display is 1k x 1k pixels, of which 1k x 768 is visible. The system can be configured with 8, 16, or 24 bits planes. With 24 bit planes, you can display up to 16 million colors at once.

Each shell's color window, which is just another tty to UNIX, starts out with 2 bit planes. UNIX only knows the number of characters per line and the number of lines per page. The display processor gets an actual clist pointer and uses it displays text on the screen. It alone knows about the current window size and font.

An application program can request any number of bit planes from the pool of 24. This request is made to the Window Manager which runs under the real time kernel. The bit planes assigned can be any anywhere, and the program does not know or care which planes it is writing. The Window Manager also assures that the top window's program has all of its bit planes visible in front all other tty windows. After a user is assigned bit planes from the pool, he may then change the color map of his bit planes, or draw something, without affecting any other tty window. Since each program has its own bit planes and color map, no one ever needs to redraw, when tty windows are popped. The system supports a C Graphics Subroutine Library, which is a Core Graphics subset, or the user may use the graphics primitives directly. Once a user has obtained his bit planes, he may break the entire screen into any number of small rectangler sections called zones, or windows in graphics terminology. Each of these zones has its own writemask, drawing colors, size and transformations stack, all of which are handled by the display processor.

User-Level Window Managers for UNIX

Robert J. K. Jacob

Naval Research Laboratory
Washington, D.C. 20375

Wm manages a collection of windows on a display terminal. Each window has its own shell or other interactive program, running in parallel with those in the other windows. This permits a user to conduct several interactions with the system in parallel, each in its own window. The user can move from one window to another, re-position a window, or create or delete a window at any time without losing his or her place in any of the windows. Windows can overlap or completely obscure one another; obscured windows can be "lifted" up and placed on top of the other windows.

This paper describes how such a window manager for UNIX is implemented entirely as a set of user processes, without modifications to the UNIX kernel. It shows how the simple, but well-chosen facilities provided by the original (Version 6) UNIX kernel are sufficient to support wm. In addition, subsequent versions of wm exploit features of the kernel introduced into newer versions of UNIX to provide faster and more sophisticated window operations, still implemented entirely at the user level.

Implementations I

3:30–5:00 Thursday

Paging in the UNIX System

Keith A. Kelleman

AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974
201-582-3586

Two research derivatives of the UNIX system have supported paging for several years: Reiser 32V, and BSD. Work is under way at AT&T Bell Labs to bring together the features of both of these systems [and others] to form a demand paged kernel for UNIX System V. This talk will discuss three areas of this work: requirements, architecture, and implementation.

The primary requirement of the paging system is that it be upward compatible with its predecessor. This means that old objects must execute unchanged, and that the meaning of system calls should not be changed. For example, fork(2) should be made efficient rather than inventing a new type of call. A second requirement is that the paging system be based on a general machine-independent architecture.

As part of the paging kernel development, a UNIX system memory management architecture is being defined. The architecture is general enough to support both paging and swapping kernels and many different memory management units. The fundamental component of the architecture is the "region." A region is a kernel data structure that represents a potential virtual address space. The basic operations defined for regions are: create/destroy, attach/detach, copy, change size, and load a file. The defined architecture can support potential new UNIX system features such as shared libraries and mapped files.

Given a memory management architecture, many demand paging techniques exist to help implement it efficiently. For example demand-zero and copy-on-write pages allow for efficient implementation of brk(2) and fork(2). Other techniques such as using referenced and modified page bits, demand loading, pre-paging, clustering, region swapping, and "sticky" regions can be used to minimize disk I/O.

Providing a Job Control Facility in UNIX System V

W. P. Weber Jr.
L. S. Weisbrot

AT&T Bell Laboratories
Murray Hill, New Jersey 07974

When users log into UNIX system, they communicate with a shell that reads commands typed at the terminal and arranges for their execution. The shell defines a user interface that is primarily sequential in nature. A user wishing to execute two commands must wait for the first to complete before the second can begin.

Although some parallelism can be achieved via asynchronous command execution (e.g. "&"), the limitation of a single terminal restricts its usefulness. An asynchronously executed command reading from the terminal will compete for input with the shell. Terminal output is multiplexed together with no way for the user to control it. Modifications to the terminal state are seen by both processes.

To overcome these problems, job control provides the ability to divide a terminal into multiple *virtual* terminals. Each of these virtual terminals may be attached to a shell and placed in its own process group; the result is a *layer*. Users create and manage layers. A user can control which layer a command will run in, which layer keyboard input is directed to, and which layer(s) may write to the screen.

This model has been implemented as two cooperating parts: a driver, *sxt*, and a user-level utility, *shl*. *sxt* provides the mapping between multiple virtual terminals and a single real terminal. *shl* provides user control through the creation and manipulation of layers.

The advantages of this model of job control are reflected in its implementation. With the exception of minor changes to the terminal device drivers and the addition of the *sxt* driver itself, no changes are required to the kernel. Also, no changes are required to the shell: the user interface is provided entirely by *shl*, a program which is small and easily modifiable. Thus, those who do not use the facility do not suffer from its inclusion. A final advantage of this model is that no changes are required to existing software to run under the facility. Programs need never know whether they are talking to a real or a virtual terminal.

Loadable Virtual Disk Device Driver and Server

Thomas A. Alborough

Creare R&D Inc.
Box 71
Hanover, NH 03755
(603) 643-3800

A useful feature that can be provided by networking software is an ability for programs on one node to transparently access peripheral devices on a remote node. This can, for example, allow programs on a system with limited disk storage space to access additional space on another system. It can also allow access to devices like magnetic tape that are not available on the local system.

Creare has developed a multi-user, error-free communications link which connects a UNIX system to a remote computer. Use of this link requires a special protocol, so access is not transparent to existing programs. In order to permit transparent access to the remote system we have implemented a UNIX device driver for a virtual block device which presents to user processes all the characteristics of a real disk device. It can be read and written as a raw device or mounted as part of the file system.

OSx: Towards a Single UNIX System for Superminis

Ross Bott

Pyramid Technology Inc.
1295 Charleston Road
P.O. Box 7295
Mountain View, CA 94039

In the high-end minicomputer market, the available UNIX operating systems provide a difficult choice to the current and potential user. On one hand UNIX System V as provided by Western Electric provides a heavily-tested, relatively bug-free operating system with the promise of long-term support and continued software development. In addition, it provides a few features not available in other UNIX systems, and, because of its size, is likely to be the system of choice for many smaller minicomputers and microcomputers. On the other hand, the 4.2 BSD UNIX System released by University of California, Berkeley offers a variety of high-performance features necessary for most supermini applications, e.g., demand-paged virtual memory, a optimized file system, kernel-based networking, etc., plus many user interface features not available in System V. Whichever alternative users select, they must sacrifice performance and/or features.

This same quandary faces the implementor of a UNIX system on a supermini: The System V and 4.2 BSD system interfaces as well as the underlying code only partially overlap in some aspects and seriously conflict in others. Applications or utilities designed to run in one environment often have subtle incompatibilities when run in the other UNIX system.

This paper describes a project to design and implement a UNIX system which incorporates all of the 4.2 BSD performance features internally while providing a truly dual 4.2/System V interface to utilities and application programs: Neither interface is layered on the other, and users can transparently tailor their environment, both at the command level interface and for software development, to perform either like a fully-compatible 4.2BSD or a full System V interface, or, with some constraints, a hybrid interface. In addition, applications developers can work in a 4.2 environment while testing an application in a separate environment for System V compatibility, or vice versa.

Specific areas of incompatibility will be described in detail, both within the kernel, and at the command and application level. The solutions we used to resolve these conflicts will be laid out, along with some of the alternatives we chose not to take.

The implementation of OSx, the UNIX operating system which was the result of this project, is complete. The effect of the continued evolution of System V (and possibly 4.2) on OSx is considered, and some future directions are proposed.

New 1/2 Inch Tape Options and Trade-Offs for 4BSD UNIX on DEC VAX Processors

Robert J. Kridle

mt Xinu, Inc.
739 Allston Way
Berkeley, CA 94710
(415) 644-0146
ucbvax!mtxinu!kridle

Five tape units representing the spectrum of currently available streaming and start-stop tape equipment were benchmarked under 4.2 BSD UNIX on a DEC VAX 11/750. The performance of each unit was measured in typical UNIX tape applications such as dump/restor and tar archiving as well as under optimal circumstances for maximizing data transfer rates. The start-stop units tested include 45 and 125 ips 1600 bpi drives as well as a new, low cost 125 ips, 6250 bpi system. Two streaming tape units were evaluated. One streaming tape unit tested included a 64 Kbyte cache which allows it to successfully simulate a 125 ips start-stop unit. The other streaming unit, the DEC TU-80, features adaptive mode switching between two streaming speeds and a slow start-stop mode.

The performance of all units is reported and compared. It is shown that the new 4.2BSD "fast file system" makes differences in tape unit capabilities more important in file archiving applications. A suggestion is made for a modification to the 4BSD UNIX tape handler for the TU-80 which could improve its performance significantly.

Implementations II

8:30—10:00 Friday

A Layered Implementation of the UNIX Kernel on the HP9000 Series 500 Computer

Jeff Lindberg

Hewlett-Packard
3404 E Harmony Rd
Fort Collins, CO 80525
hplabs!hpfcla!jbl

An implementation of the UNIX operating system kernel has been layered on top of an existing operating system kernel for the HP9000 Series 500 computer. The mapping of UNIX functional requirements onto the capabilities of the underlying OS are presented in this paper, including the changes and extensions necessary to support UNIX semantic and performance requirements. The paper covers in retrospect the advantages and disadvantages of a layered approach.

Porting Xenix to the Unmapped 8086

John Bruno Hare
Dean Thomas

The Santa Cruz Operation
Santa Cruz, CA 95060

The Santa Cruz Operation is porting Xenix to several unmapped 8086-based computers. We have encountered and overcome several interesting problems in this process.

The 8086 microprocessor is one of the most popular inexpensive CPUs-on-a-chip to emerge in the personal computer market. The 8086, and its essentially equivalent sibling the 8088, combine the versatility of the 16-bit world with the simplicity of less sophisticated processors. Many 16-bit computers employ the 8086/8088, including the IBM PC, the Victor 9000, and Convergent Technologies' Integrated Work Station. These popular microcomputers, however, are all *unmapped* 8086's. This means they do not have a memory management unit (mmu) as most UNIX systems expect.

The Santa Cruz Operation has successfully implemented Xenix on a number of unmapped 8086-based microprocessors. As a result, the UNIX environment has migrated to a very popular low-end processor. This portends an immense widening of the market for Xenix and C based applications.

Writing Device Drivers For Xenix Systems

Jean McNamara
Paresh Vaish
Richard N. Bryant

Intel Corp.
5200 NE. Elam Young Pkwy.
Mail Stop: HF2-1-800
Hillsboro, Oregon 97123
decvax!tektronix!ogcvax!omsvax!rickb

Today most microcomputers and segments of the minicomputer industries are migrating toward non-proprietary operating systems. Commercial customers no longer want to be locked into one company's proprietary software products. Intel's systems group has adopted Xenix as its standard operating system for the commercial marketplace. Xenix is Microsoft Corp. direct port of AT&T UNIX with value added enhancements. UNIX has become the de facto standard for commercial microcomputers. The most important reason for this success is the Xenix/UNIX open system concept. Customers can purchase application packages from a variety of vendors. And, they can migrate to new hardware technologies while protecting their software investment. Peripheral and board manufacturers can add new controllers because Xenix provides a means to easily add new device drivers to the operating system.

The purpose of this paper is to discuss some of the major issues in writing a Xenix device driver. We will define Xenix device drivers then discuss the kernel facilities used by device drivers. Next a typical block device driver is discussed and its structure defined. A similar treatment is provided for a character device driver. A brief discussion of system configuration is given showing how device drivers are installed. Finally some driver programming issues are raised along with an example device driver.

Transparent Implementation of Shared Libraries

Curtis B. Downing

Bunker Ramo Information Systems
35 Nutmeg Drive
Trumbull Industrial Park
Trumbull, Connecticut 06609
203-386-2674
UUCP: [ucbvax!]decvax!bunker!curtis

Frank Farance

Systems Theory Design Corporation
555 Main Street, Suite 705
New York, New York 10044
212-355-4422
UUCP: [ucbvax!]decvax!std!ff

The authors have designed and implemented a shared library utility for software development. The shared library was implemented on a Motorola 68000-based UNIX V7 system. The main feature of this implementation is that the use of shared libraries is transparent to the programmer.

This paper includes a discussion of the following topics:

* The initial motivation for shared libraries.
* The transparency mechanism.
* Compilation and loading of programs using shared libraries.
* Dynamic linking of libraries to each other and to user programs.
* Application tools for using libraries.
* Development tools for building libraries.
* Potential problems and solutions to updating and maintaining shared libraries.
* Hardware and software requirements.
* Kernel enhancements and requirements.
* The potential use of shared libraries for other software systems (e.g. UNIX "libc" library).

UNIX File System Optimization on Microcomputer Systems

David Robboy

Intel Corp.
HF2-1-800
5200 N. E. Elam Young Parkway
Hillsboro, OR 97123
503-681-5490
omsvax!dgr

As microprocessors become more powerful, file systems become more of a bottleneck, particularly in low end microcomputer systems where the cost of disk devices must be minimized. This paper analyzes the sources of file system bottlenecks that arise in one UNIX environment, and presents some strategies for optimization.

We show that system throughput is dominated by the overhead of disk accesses more than by CPU processing or the data transfer rate, and is likely to become more so as processors and controllers improve. Thus reducing the number of disk reads, writes, and seeks is likely to improve system throughput proportionally more than reducing CPU activity or using a faster disk controller. To these ends, it is worthwhile to improve the buffer hit rate and to avoid the fragmentation of files.

Optimization methods are explored, including more intelligent buffer cacheing, track cacheing, larger block sizes, and periodic reorganization of the file system. The methods used in Berkeley UNIX are examined for their applicability to a microcomputer environment.

Communications

10:30–12:00 Friday

A UNIX to VAX/VMS Communications Link

Clifford N. Cary

Creare R&D Inc.
Box 71
Hanover, NH 03755
(603) 643-3800

Creare has implemented a multi-user, error-free communications link which connects a UNIX system to a VAX/VMS system. The link can be used simultaneously by many users for such purposes as file transfer, interactive terminal sessions, and direct access to remote devices. It uses an asynchronous serial line as the physical connection and is implemented entirely in the form of ordinary user programs. No operating system modifications are required on either system. The DDCMP data link protocol is used to detect and correct transmission errors. The UNIX code is presently implemented on a MASSCOMP MC-500 system.

On each system a pair of background processes run continuously to handle machine-to-machine data transfers and multiplexing and demultiplexing of messages from user processes. Processes which wish to use the link place requests in a queue, which is implemented as a named pipe. Users first issue an Open_channel request to become connected to a partner process on the remote system. The two processes then exchange messages across a private full-duplex channel until the session is terminated by a Close_channel request to the central managing process.

We have so far implemented three kinds of application programs which use the link: file transfer, interactive terminal session (similar to cu), and a virtual disk device driver and server which provides transparent access to the link by all UNIX software.

Loving UUCP, or How I Spent My Summer Vacation

P. Honeyman
D. A. Nowitz

Bell Laboratories
Murray Hill, New Jersey 07974

B. E. Redman

Bell Laboratories
Whippany, New Jersey 07981

In this talk, we describe an experimental version of UUCP, the standard UNIX to UNIX file transfer and remote execution facility.

Hacking UUCP is the rite of passage for most UNIX hackers; over the years, myriad niggling bugs, as well as several serious inherent design flaws and limitations, have been identified in existing versions of UUCP, prompting serious UNIX hackers (and some not so serious ones) to have hack at the source code. This has resulted in a multiplicity of versions, each with its own set of bizarre quirks, flavors, and bugs. (To be fair, though, we'll admit that many of the bugs are shared by the various versions.)

Over a six-week period, we attacked these problems, great and small, in what amounts to a total rewrite of the source code. Among our goals were to make reliable the dialing and login procedures, to ease the task of integrating new types of calling devices, to provide for enhanced protection in file transfer and remote execution (discussed in another talk), to improve the robustness of the spooling schemes, to develop a set of powerful administrative tools, to make the locking mechanism reliable and tolerant, to reduce the CPU overhead in file transfer, to provide a consistent version of UUCP for all combinations of UNIX versions and host processors, to enhance the remote execution capabilities, and, perhaps most important, to reduce the complexity of the algorithms used by UUCP and their implementations.

We accomplished most of our goals, e.g., if the calling equipment is not broken, and the remote is up, and at least one line in L.sys is accurate, then the connection goes through; we understand the vagaries of communicating through switches, such as Micom, Gandalf, Develcon, DATAKIT PS, and 3Com Ethernet, and a number of modems, e.g., VenTel, Vadic, Bell 212/801, and Hayes, as well as being able to connect through a switch to a modem; we reorganized the spool directory to maintain a separate directory for each site, thereby eliminating a particularly vexing instance of fatal positive feedback.

Although this burst of effort has dragged on into a seemingly endless process of searching for further nits to pick, this experimental version of UUCP is now running in a variety of environments, entailing a wide collection of combinations of processors, UNIX versions, and system loads, including several mail and USENET hubs. In this talk, we report on our history, goals, progress, and some amusing anecdotes describing our gaffes.

New Implementation of UUCP

D. A. Nowitz
AT&T Bell Laboratories

P. Honeyman
Princeton University

B. E. Redman
AT&T Bell Laboratories

The UUCP network software has been running for over 5 years. During that time, a number of bugs were uncovered and many enhancements were suggested. Early this year, a group of interested UUCP administrators met to discuss the production of a better version of UUCP. As a result, three people divided up the work and produced a version with the following major enhancements:

The security aspect was reworked to provide more restricted file access, to provide a clearer mechanism for specifying these permissions, and to provide easier initial setup.

A more flexible system for specifying and implementing various connection media is implemented; for example, Develcon and Micom switches, and Ventel dialers.

The spool directory is now implemented as multiple directories; this improves work search time and prevents blocking due to faulty communications to a particular system.

In addition, effort was applied to simplify the code, improve the algorithms, and provide additional administration tools. This talk will focus on the security aspects of the new software.

It is now possible to specify that some remote sites can only send files, they cannot request files; this provides a high degree of security while letting remotes communicate for purposes such as mail. In addition, another option can further restrict communications by not permitting the transfer of queued

local requests during a conversation initiated by a remote site. With this option, the local secure site must call the remote in order to transfer queued files; a masquerader can not call up and receive files destined for someone else.

The directories that can be accessed for reading or writing by a remote machine are specified by READ and WRITE options. Commands that are executed by "uuxqt" can be specified on a machine by machine basis. For machines that have special execution privileges, a VALIDATE option is available to check the login-id against the machine name.

All the options are specified in the PERMISSIONS file. The defaults for the file are for maximum protection; options must be specified to give greater freedom. An attempt was made to make it easier to read and understand this scheme; the entries in the file look like shell or makefile variables, for example LOGNAME=nuucp. In addition, a program is available that reads the PERMISSIONS file and prints an English version of how the UUCP programs will interpret the file.

Mapping the UUCP Network

Rob Kolstad
PARSEC SciCompCorp

Karen Summers-Horton
AT&T Bell Laboratories

The UUCP network encompasses those machine on the UNIX News Network (USENET) and more: approximately 1800 sites as of November 1, 1983. 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 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.

Graphics

1:30—3:00 Friday

UNIX as a Development Base for High Performance Graphics Applications

Thomas Ferrin

Computer Graphics Laboratory
School of Pharmacy
University of California
San Francisco, CA 94143
UUCP address: ucbvax!ucsfcgl!tef

MIDAS (Molecular Interactive Display and Simulation) is a large interactive molecular modeling graphics package developed on UNIX which offers features previously unavailable in macromolecular modeling software. Much of the success obtained with MIDAS can be attributed to the productive environment provided by UNIX. In particular, UNIX has provided an efficient high level language for writing both user programs and a complex device driver, character stream files with arbitrary byte positioning for developing a new data base organization and access primitives, access to system source code for implementing specialized performance enhancements, pipes for logical subprocess division, performance measuring tools such as prof, gprof, vmstat and iostat, program documentation aids, utilities for maintaining revisions to the source code, and, perhaps most importantly, an example of a basic and successful "tool building" philosophy.

This presentation describes the MIDAS molecular modeling system and the impact that the aforementioned UNIX facilities have had on its success. A 16mm color film illustrating some of the unique features available in MIDAS will be shown.

DAPS and GSDL: A Procedural Approach to Graphics

Christos Tountas

Graphic Information System Technology Inc.

Presentation graphic systems generally come in two varieties: subroutine packages and non-procedural end-user systems which are interactive and "user-friendly"

Subroutine packages, used with compiled languages like Fortran and C, are powerful but difficult to use since they require considerable system design and programming effort; a significant portion of application development responsibility is left to the user.

Non-procedural systems (whether menu- or language-driven) are inherently adequate only for the limited set of operations for which they were designed but become awkward when applied in a wider variety of situations. A first-time graphics user is usually enthusiastic about easy-to-use menu systems but quickly discovers that their friendliness is inversely proportional to their flexibility. Non-procedural systems, especially those that use menus, are not well suited for data-driven, high-volume applications or for unattended operation.

DAPS (Data Analysis and Presentation System) addresses these problems by offering a high-level, interpreted procedural language (which is easier to use that Fortran or C) as well as a library of menu-

and dialog-driven applications which are written in that language. Users who need to develop new applications may do so easily, often just by modifying or extending one of the library execs. Application execs are simple disk files which are dynamically loaded when needed, so large application systems may be easily built and modified with no need for global system re-linking with each modification. In addition, separate hierarchical menu systems may be merged with ease.

GSDL is a recent variant of DAPS suitable for architectural and engineering graphics. It utilizes the same basic procedural, parametrized language but contains a richer variety of graphic object creation and manipulation operations, graphic input, a two-level hierarchical 3D turtle geometry environment, and a variety of projection and hidden line elimination operations.

DAPS and GSDL may be loaded as a single system comprising the capabilities of both. The essential characteristic of both systems is that they integrate procedural and non-procedural, interactive operations in a way that gives the user great power for the development of specialized applications in addition to a growing library of standard ones.

The Design of a Dedicated Graphics Subsystem for a UNIX Machine or How to Make Pictures in Real Time

Jack Burness

MASSCOMP
1 Technology Park
Westford, MA 01886
(617) 692-6200
masscomp!jack

To many people a graphics system appears as a kind of nifty black box which users can instruct to display their data in all kinds of wondrous formats which dazzle the eye. The actual internal operations of the system and the thought processes which led to its design are often invisible to the user. This is the natural result of any complex system. However such knowledge is often interesting and useful, especially as more and more systems become graphically oriented. This presentations describes the thoughts that went into the design and implementation of one graphics system.

Image Processing Work Station

Dale K. Hensley
TRW
O2/1796
1 Space Park
Redondo Beach, CA 90278
(decvax!trw-unix!trwspf!hensley)
(ucbvax!trw-unix!trwspf!hensley)
(trwspf!hensley@lbl-csam)

This talk will discuss the work done at TRW in designing and implementing a portable set of software routines that perform image processing, graphic generation and signal analysis on a MICRO/PDP-11.

The design of the software has evolved over 2 years and can operate on a Comtal 8300, a Comtal Vision 1/20, and a Deanza IP8500. The software runs on a PDP-11/45, MICRO/PDP-11 and a VAX 750. This portion of the talk will focus on design decisions and portability pitfalls.

The workstation implementation of this software was done on the MICRO/PDP-11 for the Space Telescope Science Operations Ground System. This portion of the talk will focus on the current workstation configuration, the problems in moving software from a 32 bit machine (VAX) to a 16 bit machine (MICRO/PDP-11), the performance of the MICRO/PDP-11, and future workstations.

Real-Time UNIX

3:30—5:00 Friday

Life With UNIX in Real-Time

Steven T. Polya
Contel Information Systems

We have been bombarded with information about what can and cannot be done with UNIX in a real-time applications environment. Here we present a case study, one that addresses most of the important topics that arise when one wishes to adapt UNIX to such a real-time application. Specifically, we will address inter-process communication, a commonly addressable data base in RAM, priority scheduling problems, and the use of the UNIX kernel for special non-interruptable operations.

Real-Time Extensions to the UNIX Operating System

Bryon Look
Gary Ho

Data Systems Division
Hewlett Packard
11000 Wolfe Road
Cupertino, CA 95014

This paper discusses real time extensions to the UNIX operating system. The real-time requirements of technical computer systems are first discussed. The deficiencies of the UNIX operating system for real-time applications are then described. Finally, proposals for providing real-time extensions to UNIX are presented.

UNIX Performance Optimization UNIX/68000/SKYFFP

Joseph F. Germann

Sky Computers, Inc.

The combination of the UNIX operating system and the Motorola 68000 microprocessor has become a standard for the high-performance work station type computer system. With all of the attributes that both UNIX and the 68000 boast, one would think that there is little room left for significant performance improvement. The tight integration of the Sky Fast Floating Point (SKYFFP) processor into this computing environment will significantly increase system performance. This holds true not only for tasks that require floating point arithmetic, but also for those functions where time consuming integer arithmetic functions are performed.

Integrating a Peripheral Floating-Point Processor into UNIX

Eryk Vershen

UniSoft Systems
739 Allston Way
Berkeley, CA 94710
ucbvax!unisoft!eryk
(415) 644-1230

The typical methods of supplying floating-point support are software simulation and tightly coupled hardware (e.g. a co-processor). A third technique is to have a peripheral floating-point processor.

This paper descibes how such a processor (the Sky Computers Fast Floating-Point Processor) was integrated into UniSoft's UNIX-derived UniPlus+ kernels. The integration enables user access to the board with no extra per operation overhead. The attendant problems and tradeoffs with respect to efficiency and reliability are also discussed.

Tom Crawley
WAUG Interim Secretary
Psychology Department
University of Western Australia
Nedlands WA 6009

Peter Ivanov
AUUGN Editor
School of EE and CS
University of NSW
Kensington NSW 2033

Dear Peter,

I am writing to inform you of the formation of the Western Australian Unix* Group (WAUG).

WAUG was started by four Unix users, Tom Crawley and Tony Gibbs from Psychology UWA, Glenn Huxtable from Computer Science UWA, and David Ingles from NCR Perth to provide for a mutual exchange of Unix related information and ideas in Western Australia.

Introductory letters were sent to commercial organizations and academic institutions known to be interested in Unix. The group held its first official meeting on November 30 1983 at the NCR offices in Perth, primarily to decide the aims of the group and to get to know each other. Thirteen Unix devotees attended the meeting were it was agreed that the aims of the group were to promote and popularise Unix, to provide advice about Unix to the public, and to offer a forum for discussion about Unix.

The meeting also decided that a sub-committee should look into the drafting of a constitution. An NCR Usergroup's constitutions was supplied as a starting point. However, it was agreed that if the AUUG has a constitution, this could be modified for use by the WAUG, subject of course to the consent of the AUUG. Could you please send me a copy of the AUUG constitution? Also, what methods exist for the affiliation of WAUG to AUUG?

It was decided that the first two meetings would be by 'invitation only' to allow administrative details (ratification of a constitution etc.) to be sorted out. After the Group has settled down, WAUG meetings will be open to anyone interested in Unix. I can contacted at work on (09) 380 3270 or through the SUN network as "tomc:wapsyvax",

Yours sincerely,

Tom Crawley

*Unix is a trade mark of Bell Laboratories.

Department of Computer Science
University of Canterbury Christchurch 1 New Zealand

11 January 1984

Mr K. Hill
School of Electrical Engineering & Computer Science
UNSW
P.O. Box 1
Kensington
N.S.W.
AUSTRALIA 2033

Dear Kevin

I wrote a letter to you on 10 November last enquiring about possible "portable" languages on UNIX that we might bring across to our Data General UNIX. I've not heard from you as yet, but would appreciate any help that you can give.

There's also a further matter I'd appreciate your advice on. We are trying to convert some large pieces of software from RDOS to UNIX,including a full compiler for SIMULA. Under RDOS, these are overlayed programs with user calls to load the appropriate overlay. Is there any comparable facility that users have written for UNIX? On burrowing into "ld" I find that our version (from you) has a "-v" option which is supposed to create some form of overlay, and you can create a file with a Magic Number of 405, but there seems little other support for it. The "link" program for F77 programs has the correct sort of facilities, but it's not at all clear that you can use that for anything other than F77.

Can you give me any pointers, such as people who may have tackled this problem? If not, possibly you could pass this on to Peter Ivanov for possible inclusion in an "AUUGN".

Thanks for all your help.

Yours sincerely

Dick Cooper

R.E.M. Cooper

Senior Lecturer
Head of Department

University of Waikato
PRIVATE BAG : HAMILTON : NEW ZEALAND : TELEPHONE 62-889

DEPARTMENT OF COMPUTER SCIENCE

1 December, 1983

P. Ivanov,
Editor, AUUGN,
School of Electrical Engineering &
Computer Science,
University of New South Wales,
P.O. Box 1,
Kensington 2033,
AUSTRALIA.

Dear Peter,

/usr/group/NZ

Thank you very much for your informative and helpful letter of 24 November sending me, among other goodies, a list of subscribers to your Newsletter in this country. Many of the names I already had on our database (with revised addresses in some cases!) For your interest I include a printout of our current member/contact database.

???	represent currently unknown phone numbers or extensions.
+	is against an AT&T contact name.
~	is against another user group overseas contact
*	is against a Steering Committee member's name.

As you will see from the enclosed first interim newsletter we are just in the stages of forming, hoping to "launch" properly in May next year. Any other international contacts you can let me have would be most welcome. With AT&T efforts for the South/most-of the Pacific being conducted from Sydney I suspect it might be a "good thing" if we considered getting together as a sort of PUUG - a bit like EUUG?

May I, however, pick your brains directly on a few points which would be useful for our next committee meeting in about 12 days time? What kinds of member (+ A$ sub) does AUUG have? We just have tentative thoughts at the moment. Can non-members purchase AUUGN? If so what subscription rate? We are interested in the Network News from Edinburgh but, at the moment, NZPO does not provide dial-up services - so we have to wait a few months (at least). How does it work from your end? Who's your technical expert on it (for when we can get in on things)? You'll probably know that manufacturer support for UNIX user groups is strong in UK/Europe/USA - is it in Australia? Have you had a "database" machine donated? How did you persuade them? (if you did?)

I could think of dozens more detailed questions if I sat here long enough, but those are the sorts of things which are concerning us at the moment in our attempt to really get started with a swing as a financially sound, active, helpful group!

Yours,

(K. Hopper) Secretary Steering Committee

/usr/group/NZ

INTERIM NEWSLETTER No. 1 November, 1983

Welcome to the happy band of UNIX users! This interim newsletter is the first of two or three which will probably appear before our formal inaugural general meeting in May next year (DON'T FORGET - put the dates in your diary NOW -- 16 - 19 May at Waikato University).

Since we aren't yet formally incorporated and registered as a charity we can't yet ask for a subscription (fear not - we'll get round to it!) - so these newsletter must be single sheet duplicated stuff. We'll just have to try and ensure that they are interesting.

> (Thinks! Did you know that over 15 makes of microcomputer - and at least twice as many models - are available in New Zealand running a version of UNIX (or a "look-alike")? No prizes for naming them all!)

User Group Objectives

At the last Steering Committee meeting we spent quite a long time discussing the objectives we variously had in mind for forming the group - what we were prepared to put into it (apart from money) - and what we hoped to get out! Like any other good/useful organisation much of our success will come from what we put in.

As a starting point in our discussions we took an interesting article by Teus Hagen in the European Unix User Group News (EUUGN) Vol. 3 No. 2 entitled "Cookbook for setting up a National UNIX systems Users Group". While predicting a European environment, section 19 stated that the Goals should probably be:

- exchange of information about UNIX.
- distribution of software/documentation, as far as licensing allows.
- advice on hardware and software for UNIX.
- preparation of the UNIX market.

These ideas seemed pretty good to us although we believe that we ought not to exclude "look-alike" systems since most of these are products produced principally by microcomputer manufacturers based upon their own UNIX commercial licence. The "New Zealand scene" is full of micros offering this facility and these users need just as much support as those who have direct licences from AT&T.

Also in Teus Hagen's article is the suggestion that the means for achieving these objectives should include

- annual (closed) meeting for members only.
- annual open conference for members, vendors and others interested in UNIX.
- UNIX networking.
- membership of any international UNIX group.
- communications arrangements with vendors & members.

Again these seemed pretty good to us - except that there is, at the moment no appropriate international group - we could, however, join EUUG and possibly also /usr/group in the USA to keep up with their activities. UNIX networking and general communications arrangements also seem pretty much dependent on finance and the generosity (hopefully!) of vendors (many of you will, for example, have read of DECs support of UNIX Network News in Europe - they provided a VAX free!). A newsletter (better than this one, of course) is an

obvious service, but the use of a database maintained by the group and accessible to members on a 'dial-in' basis is the ideal to aim for. We leave you to work out the sums.

> (IBM-watchers! Even the giant has weaknesses - it won't "get into bed with" AT&T (another giant) because, perhaps, of anti-trust problems in USA. Rumour hath it, however, that "a well-known operating system will have another look-alike (no relation of course) on a shortly-to-be-announced-new-micro-computer from a world-famous manufacturer".)

User Group Membership

Our preliminary Steering Committee discussions seem to indicate that there will be three classes of membership, all of whom must prove that they have legitimate use of machines with a UNIX licence (direct from AT&T or otherwise):

- Individual member
- Institutional member - a non-profit organisation site.
- Corporate member - a commercial organisation site.

Discussions on annual membership fees and any difference in privileges are still going on and can only really be settled when the services wanted and likely number of members is known more accurately!

> (How many of you have read the August issue of BYTE magazine? It is a special about C. Most of the articles are quite good, discussing as well as C some UNIX features.)

Workshop/Exhibition

Just a few short notes for this newsletter to whet your appetites and make sure you ask the boss to send you along.

The workshop looks like having a general section of interest to all and two concurrent sections - tutorial and 'techincal'. The general section will include keynote speakers from overseas (we hope) and general discussion papers on UNIX history, philosophy, licencing and likely future developments. The tutorial section will introduce those who are interested to some of the more elementary features of UNIX both for the user and 'system administrator' - talking about setting up and tuning a system for an organisation and even its individual users. The technical section will include papers on porting they system to a new machine, networking, inter-system communication, etc. It promises to be quite exciting.

The technical exhibition which goes with the workshop hopes to have lots of representative hardware and software and hopes to have most of them "talking to each other" - and to equipment elsewhere!

> (Individual delegate fee will probably be $100 - $150 to cover invited speaker expenses. Full board at the university comes to $32 per night!! - say $250 + travel approx!)

Notes on Correspondence

Thanks for all your letters of interest. What we want now are ideas about what we should - or should not - be considering for our formal articles of association - objectives, means, costs etc. A brief note or letter now could save a lot of argument and discussion at our first general meeting.

> (DIARY ENTRY! 1400 18 May 1984 General Meeting)

kh

Welcome to the big year - yes, big because it is only a few short months to the First General Meeting of our group. By the way, in arranging the detailed workshop timetable (of which more later) we have moved the meeting to the Thursday - 17 May!

Most of you will be either about to go off or have just returned from your summer holiday as you read this. Sun and sand and sea sound ideal in summer weather, don't they? Those of you about to depart on holiday may like to spend a little while when fishing or boating or just sunning to think just how we can help each other in this group - then drop me a line when you get back to work. The same need to think about this applies to us all of course and for those who have returned to bulging in-trays the time to do it is NOW - before getting immersed in the everyday grind again.

> (Don't sit counting shells on the sea-shore like Sister Suzie - but count UNIX ones instead! Apart from csh, rsh and sh which are the "originals" which came from Bell and UCB, there must now be as many "business" shells as there are UNIX look-alike systems. For those with insomnia shell-counting is recommended instead of counting sheep!)

Helping Each Other

Despite the early stage in our organisation and, at the moment, our relatively small numbers (we have broken the fifty barrier - and rising), a number of you have already rung up for help and been passed on to someone else or occasionally been offered assistance directly. This is very encouraging indeed and means that we are beginning to do what we are forming the group for. When we get ourselves formally organised we can find out more about each other and have a better idea of where our individual strengths lie. We can produce lists of members and, depending on our budget, offer various other helping hands. In the meanwhile, however, thanks for helping and keep up the good work.

> (Did you know that at least three versions of Cobol run under UNIX on various machines?)

Correspondence

We've had several very interesting letters in the last few weeks which are worth giving wider publicity. There are a few of you who pay your A$24 for AUUGN - the newsletter of our Australian colleagues. They are interested in our group formation and have wished us well - although I suspect this may be with tongue in cheek since they might lose a few subscriptions.

The subject of a group logo has been discussed in correspondence - and by the Steering Committee too! With some reservations "usr/group/NZ" seems to be acceptable as a short title. Logos of the form like:

```
                U
usr/group/NZ        {U IX in blue?}
                I
                X
```

are being bandied around and we've even had the suggestion

```
N                N U Z
UNIX     or        N
Z                  I
                   X100
```

for the newsletter heading. No prizes offered, but any suggestions which will look attractive, be in keeping with the nature of UNIX and our group will certainly be considered. Any artists among you? Oh! One small reservation - we'd like to be able to reproduce the final result on a 'dot' type printer/plotter from troff source.

The third most significant area of correspondence related to the workshop and our earlier budgetary cost suggestions in the last newsletter. See the workshop section of this newsletter for results!

> (Of the six or seven relational-style database systems now offered under UNIX, how many were designed and built on the UNIX building-block philosophy? How many were just "ported" to run under UNIX? How many were designed in the traditional way - but to run under UNIX? No answers given - but ask yourself why the questions were asked and what would be the significance of the answers for their use!!)

Workshop

We've now got a workshop time-table of two and a half days with the exhibition extending up to the Saturday morning when exhibitors will probably be taking things away!! Wednesday afternoon (16 May) will be a separate (and separately charged - about $30) 'Introduction to UNIX' session for those of you who feel the need for a mini-course - although it will be fairly high pressure (allowing you to think about it and ask questions of people over the next couple of days!).

Registration will start at about 1700 on the Wednesday with Exhibitors Presentations scheduled for a couple of hours in the evening (and Thursday evening for other exhibitors). For those who didn't register on Wednesday the first keynote speaker on Thursday morning won't start too early to let you register first thing. Thursday finishes with the General Meeting after afternoon tea - the intervening time being taken up with parallel tutorial and technical sessions. After a keynote speaker to start Friday (probably on "The Newcastle Connection") four presentations on international UNIX activites, the future of UNIX, commercial users and UNIX, and, possibly, data communications and UNIX will be followed by a controversial Panel Discussion (very tentatively entitled "will UNIX become another COBOL?").

Although we haven't yet been able to work out a fully detailed budget for the workshop, it now appears that we should be able to keep our costs down enough to make $100 for the two days a likely maximum. With two days accommodation maximum needed (@ $32 each day (including meals)) then you can work out your own budgets based on $100 + the optional introductory session at $30. I think you will agree this is a little better than originally suggested.

By the way, since we haven't at the moment any formal membership this will be the one and only workshop open to non-members! In that sense it will be unique. Come along yourself, bring a friend (husband, wife, colleague) and make it unique in other ways too!

> (A clue - read Software - Practice & Experience,
> Vol. 12, pp. 1147-1162 (1982)
>
> An invitation - read next month's Interface - more articles
> on UNIX!)

Details of Australian UNIX User Group Meeting
February 20 and 21, 1984.

The Meeting will be held on February 20 and 21, 1984 on the campus of Sydney University. Registration for the meeting will be between 8.30 and 9.15 on Monday morning (20.2.84). This will occur in the foyer of the "Footbrdige Theatre" located at position C14 on a University of Sydney grid map. The entrance to the theatre faces Parramatta Road.

The first day will be devoted to general interest non-technical papers on Unix, with the second day being of a more technical nature on applications and system development. As there is a strong interest in tutorials for people with little or no Unix experience, these will be held starting at 9am on Tuesday 21st February. Lectures and practical sessions will run until 5.30pm. those with little or no Unix experience. The fee is $100 per person ($60 for the morning alone) Places are limited so contact Piers Lauder on (02)449-0220 as soon as possible.

As the Early Bird registration is now closed, further registration fees will only be accepted at the meeting. Registration fees are $30 and $20 for students. If you plan to attend the meeting, please contact Margaret on (02)449-0220 so that seating can be guaranteed.

The manufacturers/suppliers attending the meeting and showing equipment include:

Manufacturer	Equipment
Email	Unison
NCR	NCR tower
Wicat	Wicat
Hewlett-Packard	HP9000, H9836
Perkin-Elmer	PE
Horizon	Horizon
Sigma Data	Sigma Convergent
TCG	Plexus, Sun
Digital Electronics	Unity
Computer Enterprises	
Tektronics	
I.C.L.	

If accommodation is required it is available at:

Women's College	Contact Mr. Len Cupitt (02)51-3761 $28 - full board single/day $45 - full board twin/day
City Gardens Motel	(02)690-9100 Studio apartments $56 plus $6 per extra person. 4 people per apartment 1 bedroom apartments $62 plus $6 per extra person. 5 people per apartment.
University Motor Inn	(02)660-5777 1 or 2 people sharing a room $52.50 3 people sharing a room $58.00.

Australian UNIX User Group Meeting
Provisional Program
Monday, February 20th, 1984

Judy Kay
Lecturer, Sydney University
A Review of books on UNIX

John Lions
Professor of Computer Science, University of New South Wales
The Future of Unix

Chris Rowles
Siromath Pty.Ltd.
Installing and Managing a UNIX System

Boris Schlensky
Managing Director, Digital Electronics Pty.Ltd.
Setting up an Australian UNIX Software Factory

Richard Tweedie
Managing director, Siromath Pty.Ltd.
UNIX and Siromath

Presentations by Various Manufacturers
UNIX Hardware

Australian UNIX User Group Meeting
Provisional Program
Tuesday, February 21st. 1984

Australian UNIX Users Group Business
Adoption of Group Constitution and other matters

Ron Baxter
CSIRO
A Review of Statistical Software

Jason Catlett
Sydney University
Expert Systems on UNIX

Bruce Ellis
Sydney University
Bedbug; an intimate debugger

Robert Elz
Melbourne University
Searching Paths, Quickly

Robert Elz and Piers Lauder
Melbourne University and Sydney University
State of the Australian UNIX Network

John Gibbons
CSIRO
Porting UNIX to the "micronode"

Ross Green
Benchmarks I have run

Richard Grevis
University of New South Wales
A Graphics Editor called TESS

Tim Long
TATA-ELXSI
Porting UNIX to a Fast Multi-processor

Glynn Peady
Australian Atomic Energy Commission
Interfacing the Quadritek 1600 Photo-typesetter to "troff"

Roy Rankin
Sydney University
Implementing "f77" on a PDP-11/34

Peter Ivanov
AUUGN Editor
School of EE and CS
University of New South Wales
PO Box 1
Kensington NSW 2033
AUSTRALIA

+61 2 662 3781