AUUGN

Australian Unix User Group Newsletter

Volume 5 Number 5

Registered by Australia Post Publication No. NBG6524

The Australian UNIX* systems User Group Newsletter

Volume 5 Number 5

October 1984

CONTENTS

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

Editorial

Well, finally, the Australian UNIX systems User Group is official. (Hooray) The office-bearers are:

John Lions - President
Greg Rose - Secretary
Chris Maltby - Treasurer
Colin Webb - Returning Officer
John O'Brien - Assistant Returning Officer
James Mann - Auditor

The Management Committee consists of the President, the Secretary, the Treasurer and four General members. The General members elected to the Management Committee were:

Robert Elz, Ken McDonell, Piers Lauder and Tim Roper

I have reproduced the minutes of the meeting and the amended AUUG constitution at the end of this issue, along with the inevitable forms for founding membership, membership and newsletter subscription. I have also included a list of normal and founding members, and a list of subscribers who qualify for founding membership status.

All correspondence should be addressed to

Greg Rose
Honorary Secretary, AUUG
8 Meadow St
Concord NSW 2137
Australia

Message from the President

I would like to thank those members who attended the meeting in Melbourne in August, who accepted the new constitution substantially as presented, and who chose me as AUUG's first president. I shall endeavour to serve you, and, in conjunction with the new management committee, to make the constitution work effectively.

Since the August meeting, much has happened. I attended the European UNIX system User Group's meeting in September in Cambridge, at the invitation of David Tilbrook. This was an excellent conference, very well organised, with well-prepared talks, and a majority of speakers from North America. A.T. & T. was there in strength. The glamour spot was Sam Leffler's description of recent developments at Lucasfilm, finishing up with the showing of a short cartoon of astonishing technical virtuosity. The graphics presentations from C. Huang of UC San Francisco Computer Graphics Laboratory, and from Greg Chesson of Silicon Graphics, were also memorable. The most thought provoking paper was given by Mike O'Dell on UNIX IPC: past, present and future. My own paper, a survey of some recent developments in this country, seemed to be well received. More details of this conference will be published later.

The commercial exploitation of the UNIX system is creating many new activities and opportunities. Early in September I learned that ``Computerworld'' in Australia had decided to hold a UNIX-related exhibition

and conference in Sydney in May, 1985. Since then I, and more recently the whole management committee, have met with Mr Stephen Moore of Computerworld. We have negotiated an agreement that should prove mutually beneficial to both parties. The Computerworld conference will emphasise the commercial application of UNIX, and should complement, not compete with, the AUUG's traditional conferences. The AUUG, through its management committee and other interested members, will participate actively in organising the programme for the May conference, and in particular, the set of tutorial sessions. You will hear more on this in due course. For now, may I ask any AUUG member who would like to particpate in the organisation and/or presentation of particular tutorials at the May conference to contact me as soon as possible.

The organisation of the Februrary AUUG conference in Wollongong is now well in hand - a call for papers appears elsewhere in this issue. I would particularly like to encourage people with working, effective applications of UNIX outside the university environment to come forward and describe what they are doing - there are many people out there who haven't heard it all before, and who will be very interested to learn from your experiences.

Yours sincerely,
John Lions.

Next AUUG Meeting

The next AUUG Meeting will be held in Wollongong in early February 1985. Further information and the call for papers appears on page XX.

New Magazine

A new glossy magazine called "UNIX/WORLD" has hit the streets. Its definitely worth the introductory subscription fee (US$18 plus postage) and for further information you should write to

Cheryl Hogan
Circulation Director
UNIX/WORLD
289 S. San Antonio Rd.
Los Altos CA 94022
U.S.A.

The network mail address of the President/publisher, John M. Knapp, is

decvax!vortex!u-world!johnk:mulga or
vax135!ihnp4!dual!u-world!johnk:mulga

Contributions

As usual, more contributions please!

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

Books

Keep an eye out for the October 1984 Bell System Technical Journal. It is another special edition on the UNIX system and associated things.

This time we have swags more books to add to the list. If you are not keeping your list up to date, don't worry, I will publish the complete list in the next issue.

Books on UNIX

19. The Business Guide To The UNIX System
Jean L. Yates and Sandra L. Emerson
Addison Wesley

20. The UNIX Guide (2nd Edition)
Pacific Micro

21. The UNIX Operating System Book
M.F. Banahan and A. Rutter

22. Operating Systems Pocket Guide: UNIX
Laurie Blackburn and Marcus Taylor
Pitman Publishing

23. UNIX For People
P. Birns, P. P. Brown and John C. Muster
Prentice-Hall (1984)

24. Real World UNIX
Halamka
(due for release late 84)

25. A Business Guide to Xenix
Yates et. al.
(due for release late 84)

26. UNIX on the IBM PC
Twitty
(due for release late 84)

27. Exploring The UNIX Operating System
Stephen G. Kochan
(due for release late 84)

Books on C

11. C Programming Guidelines
Thomas Plum
Plum Hall (1984)

12. A Book On C
Al Kelley and Ira Pohl
Benjamin/Cummings

13. C Programmer's Library
    Jack J. Purdum et. al.
    Que Corporation

14. Understanding C
    Hunter

15. Introduction to C
    Chirlian

16. C Programming Standards And Guidelines
    Thomas Plum

17. The C Programming Reference Manual
    Samuel P. Harbison and Guy L. Steele
    Prentice-Hall (1984)

18. C Programmer's Handbook
    Hogan
    (due for release late 84)

19. Programming In C On The IBM PC
    Pollack
    (due for release late 84)

20. C Language User's Handbook
    Weber Systems
    (due for release late 84)

Related Books

7. The Small C Handbook
   James Hendrix
   Reston Publishing Co, Inc (1984)
   (Australian Distribution through Prentice-Hall)

8. Comparing And Assessing Programming Languages:
   Ada, C and Pascal
   Alan R. Feuer and Narain Gehani
   Prentice-Hall (1984)

9. UNIX Applications Software Directory (2nd edition)
   Edited by Ray A. Jones
   Onager Publishing

10. The UNIX System Encyclopedia
    (This a vendor directory - Ed)
    Yates Ventures

11. Operating System Design:
    The Xinu Approach
    Douglas Comer
    Prentice-Hall (1984)

12. Using The Horizon(TM) Spreadsheet With
    The UNIX Operating System

Don Beil
Prentice-Hall (1984)

13. Word Processing On The UNIX System
Kreiger
(due for release late 84)

Nets

The following sites have notified me of alterations to their site information. Most changes result from the installation of a new PABX at U.N.S.W.

==========

Name: bio23
Address:School of Biological Sciences
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 697 2008
Machine:
PDP 11/23 + FPU., RL02, Tektronix 4662 plotter, UNIX level 7 AUSAM
Contacts:
Karl Redell (karl:bio23)

==========

Name: cadvax
Address:School of Electrical Engineering and
Computer Science
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 697 4040
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: civil
Address:School of Civil Engineering
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 697 5045
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 2 697 2922
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 2 697 2922
Machine:
    DEC PDP 11/40, RK05, Pertec (20 megabytes), UNIX level 6 AUSAM
Contacts:
Vincent Lawrence (vince:comm40)
==========================================================================
Name: csu40
Address:Computing Services Unit
        University of NSW
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 697 2927
Machine:
    PDP-11/40, 2*RK05-J, RK05-F, DJ-11, 124Kw core.
    Unix level 7 Ausam

    Serves as SUN switching node - no user accounts.  (Has 10 nodes!)
Contacts:
S. F. Mok (mok:csu60)
==========================================================================
Name: csu60
Address:Computing Services Unit
        University of NSW
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 697 2927
Machine:
    PDP-11/60, 2*RK05-J, Ampex DM980 with AED 8000 controller,
    TU10, 2*DZ11, LV11, DP11, DR11-B, user control store.
    Unix level 7 Ausam

    The CSU production machine - takes over from "csu40" except network.
Contacts:
S. F. Mok (mok:csu60)
==========================================================================
Name: csuvx0
Address:Computing Services Unit
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 697 2926
Machine:
    VAX11/780
Contacts:
==========================================================================
```

Name: csuvxl
Address:Computing Services Unit
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 697 2926
Machine:
VAX11/780
Contacts:
==========

Name: csuvx2
Address:Computing Services Unit
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 697 2926
Machine:
VAX11/780
Contacts:
==========

Name: deakcm
Address:Division of Computing and Mathematics
Deakin University,
Waurn Ponds VIC 3217
AUSTRALIA
Phone: +61 52 47 1319
Machine:
PDP 11/60, RK07, RL01, TM11, DZ11.
UNIX level 7 AUSAM
Contacts:
Craig Bishop (craig:deakcm)
==========

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 2 697 4040
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: elec35
Address:School of Electrical Engineering and
Computer Science
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 697 4040
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 2 697 4040
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 2 697 4040
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 2 697 4040
Machine:
    PDP 11/70, RP04, TE16, 2 * qume micro 5, UNIX level 7 AUSAM
Contacts:
Kevin Hill (kev:elec70a)
Peter Ivanov (peteri:elecvax)
================================================================================
Name: elecvax
Address:School of Electrical Engineering and
        Computer Science
        University of New South Wales
        PO Box 1
        Kensington NSW 2033
Phone: +61 2 697 4040
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 2 697 4372
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: mathvax
Address:School of Mathematics
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 697 2974
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 2 697 4153
Machine:
Contacts:
David Herd (davidh:mech)
==========

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:
Colin Murphy (colin:sri)
==========

Name: srl
Address:School of Electrical Engineering and
Computer Science
University of New South Wales
PO Box 1
Kensington NSW 2033
Phone: +61 2 697 4040
Machine:
PDP 11/34A, RL01, UNIX level 7 AUSAM
Contacts:
Peter Ivanov (peteri:elecvax)
Kevin Hill (kev:elec70a)

==========

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 2 697 4070
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: tictoc
Address:TIME. Office Computers (Research)
6th Floor
221 Miller St.
North Sydney
Phone: +61 2 925 0555
Machine:
VAX-11/750, 2 Mb, UNIBUS with TS11/TU80 lookalike (Keystone)
and 2 x VMZ/32N, Emulex SC750 with Fujitsu 2351A Eagle (474Mb)
and CDC RSD (80Mb pack), Dataproducts B600 line printer;
UNIX System V Release 2.0
Contacts:
John Mackin (john:tictoc)
Geoff Cole (geoff:tictoc)
Ray Loyzaga (ray:tictoc)

==========

Name: timeland
Address:TIME. Office Computers (Software)
7th Floor
221 Miller St.
North Sydney
Phone: +61 2 925 0555
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:
Neil Russell (neilr:timeland)

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 2 697 4030
Machine:
PDP 11/40, RK05, RL02, DR11b, AR11, TA11, UNIX level 7 AUSAM
Contacts:
Ted Spooner (teds:unswpower)

C A L L F O R P A P E R S

Australian UNIX* systems User Group
1985 Summer Meeting

The 1985 Summer Meeting of AUUG will be held at the Department of Computing Science of the University of Wollongong, Wollongong, N.S.W. on **Monday, February 11** and **Tuesday, February 12, 1985.**

The meeting will be devoted to topics of interest to the UNIX community. Presentations are invited on all aspects of the UNIX system and its applications. Presentations in the following categories are especially encouraged:

UNIX Systems
Implementation of network or distributed systems; porting of UNIX to new computers; system management and performance; system standards.

UNIX Applications
Graphics; database systems; statistical systems; office systems; business applications, transaction systems.

UNIX Programming Tools
New utilities; programming environments; new programming languages; their implementation.

Selection of papers for presentation will be based on the submission of a synopsis. A synopsis (extended abstract) must contain sufficient detail to enable the program committee to determine the suitability of the submission for presentation. A synopsis should not exceed 1,000 words. Each synopsis should contain the following information: Title; Name of Author; Affiliation of Author; Mailing Address; Phone Number; Network Address (if available).

Abstracts must be submitted by Friday, December 7, 1984 to the addresses below, by either conventional or electronic mail. Authors will be notified concerning acceptance by January 1, 1985.

The program committee also intends to schedule panel discussions, tutorials and review or overview presentations. We are open to suggestions from the UNIX user community as what sessions should be included. Such comments should be submitted as soon as possible.

AUUG Conference
Department of Computing Science
University of Wollongong
P.O. Box 1144
WOLLONGONG N.S.W. 2500 (042) 270 859
Australia.

Network Address for enquiries,
information and submission of
synopses and suggestions: auugm:uowcsa

* UNIX is a trademark of AT&T Bell Laboratories

Winter 1984 AUUG Meeting. Programme

Monday, August 27

10:30: Introductory Session, Chair: Robert Elz

Introduction, Welcome, and General Business

Rob Pike, *AT&T Bell Laboratories*

The Blit: Merging Bitmap Graphics and UNIX

Introduction to the Business Meeting: John Lions

12:15: Lunch

13:45: Business Meeting: John Lions

14:30: Break

14:45: Technical Session (1): Chair: Ken McDonell

Piers Lauder, *Basser Dept of Computer Science, University of Sydney*

Domain Addressing in SUN III

Peter Ivanov, *Dept of Computer Science, University of New South Wales*

A UNIX Network Information Database - or how to waste more time than reading news.

Glenn Trewitt, *Basser Dept of Computer Science, University of Sydney (on leave from Stanford University)*

Internetwork Protocols - or how I spent my summer

16:00: Break

16:25: Technical Session (2): Chair: Piers Lauder

T.R. Cordingley and D.W.E Blatt, *Maths, Stats, and C.S., Univ Newcastle*

A General Purpose Multiprocessor Kernel Written in C for a UNIX Programming Enviroment

A.J. Hurst, *Computer Science, ANU*

JAS - A Modula Based Single User System

Keynote Address: ***Rob Pike,*** *AT&T Bell Laboratories*

cat -v Considered Harmful

Close at approximately 18:00

Tuesday, August 28

09:00: Business Meeting (2): John Lions

09:45: Break

10:00: Technical Session (3): Chair: Tim Long

John O'Brien, *Fawnray Ltd*
The Architecture of the UNIX Software Factory

Dr Y Kuan Oon, *Community Medicine, Monash University*
Computer Records Considered Harmful Under UNIX

11:00: Break

11:30: Technical Session (4): Chair: Peter Ivanov

Ron Baxter, *C.S.I.R.O. Division of Maths & Stats*
A Review of MH – Real Communicators Don't Use mail

Greg Rose, *Tata Elxsi*
Some Concurrency Issues in Multi-processor UNIX

Ken McDonell, *Dept of Computer Science, Monash University*
The Pyramid 90x: An Overview and Assessment

12:45: Lunch

14:30: Technical Session (5): Chair: Ross Green

Des FitzGerald, *Desmond FitzGerald and Associates P/L.*
An Interactive Graphics Mining Ore Reserves System

John O'Brien, *Fawnray Ltd*
Fear and Loathing on the 32016

16:00: Conclusion

NAME

Australian Unix-system Users' Group Winter Meeting, 1984. Talk Abstracts.

SYNOPSIS

cat -v <<'!Page!11!' | more

DESCRIPTION

The following pages contain abstracts for talks scheduled to be given at the winter meeting of the AUUG, 1984.

Some of these were obtained over networks, and thus can reasonably be trusted. Some were obtained on paper, and transcribed. These may contain transcription errors. Others are purely fiction!

CAVEATS

Many of the words in the following abstracts are registered trade marks of various organizations. Prominent amoung those are UNIX (AT&T Bell laboratories), DEC and VAX (Digital Equipment Corporation), XENIX (Microsoft), and OSx (Pyramid Technology Corporation). The "Blit" is commercially known as the Teletype 5620 Dot Mapped Display terminal.

BUGS

The order of the following abstracts is mostly governed by a desire to save printing costs by compressing two abstracts onto one page wherever possible. Thus, for most purposes, the order should be considered random.

DIAGNOSTICS

Quiet warnings when a speakers time is almost up. Loud warnings when his time has expired. Physical violence if he refuses to stop.

SEE ALSO

The actual talks.

Rob Pike

Bell Labs 2C-521
Murray Hill NJ 07974

The Blit: Merging Bitmap Graphics and UNIX

UNIX is a multiprogramming system. A user can run several programs simultaneously, programs that can interact (such as programs in a pipeline) or can be independent (such as parallel compilations). The syntax and semantics of pipelined processes provide a powerful and convenient form of multiprogramming, but Unix's current mechanisms for dealing with parallel independently executing programs are weak: there is no convenient way to control several processes. The "job control" mechanism of the C-shell merely provides a way to describe which of several processes the user wishes to work with now, and does not provide a capability for letting several programs run simultaneously without interfering with, say, each other's terminal i/o.

Bitmap displays, as they have been traditionally used, take a natural first step towards controlling a multiprogramming system. "Window systems" enable a user to store multiple program contexts on the same screen, but they have only (with a couple of exceptions) been static contexts, and are therefore incapable of handling multiprogramming.

The extension of the window system idea to supporting multiprogramming is simple conceptually, but surprisingly difficult to implement. There are interesting problems to solve in the areas of

— inter-process communication

— graphics support

— user interface.

This talk will address the problems and how they have been solved on the Blit terminal, a bitmap display built specifically for improving the user interface to UNIX. It will also discuss some of the extensions of the ideas developed to other areas, such as game playing and text editing.

Rob Pike

Bell Labs 2C-521
Murray Hill NJ 07974

The Blit: Merging Bitmap Graphics and UNIX

Special shorter abstract in Large, Easy-to-Read-Type:

The Blit terminal developed at Bell Labs by Bart Locanthi and Rob Pike aims at the middle ground between time-sharing and personal computing, providing the advantages of high-speed interactive graphics against the familiar backdrop of the UNIX time-sharing system. The terminal has very inexpensive hardware, cheap and simple enough to take home, but designed with software in mind by the people who planned to write the software and use the terminal. The result is a working environment which augments UNIX rather than compete with it. Two general improvements introduced to UNIX by the Blit world are

- interactive graphics for (potentially) any user
- a multiprogramming terminal to assist the multiprogramming operating system.

This talk will focus on issues of software design (particularly the division of labor between hardware and software, terminal and operating system) and user interface, and will be peppered with examples of Blits in action.

John O'Brien

Fawnray Ltd.

The Architecture of the UNIX Software Factory

It is intended that this talk inform members of the UNIX community of the existence and organizational structure of the UNIX factory. Following the press release earlier in the month, explanation of some details which were not disclosed.

Why should Australia establish a UNIX factory?

Numerous arguments justify a specialist UNIX software house. Reasons include, explosive growth in utilization of UNIX, support of local manufactures, portability and standardization, economies of scale, reduced duplication of effort, and utilisation of research undertaken at tertiary institutions. Commercial UNIX environments require software tools intrinsically different to those provided by AT&T.

What are the primary requirments of the UNIX factory?

Specialist talents are required to fulfil special needs. Talents of this type have been cultured unwittingly by Universities. System Progammers and System Supervisors within these institutions usually have extensive UNIX experience. Tapping this expertise requires an organisation that understands the needs of staff, and customers.

When should this corporation be established?

It was essential that the organisation be established rapidly. Manufactures identified the need to have UNIX ported to their processors. The cost for each manufacturer porting their own UNIX is astronomical. Porting UNIX is only a minor difficulty compared with aquiring good application software. Porting UNIX and system software to manufactures hardware is essential. If the talent is not utilised now, it will go offshore. Australia will drive out it's UNIX expertise.

With whom do you start?

Australian software companies tend to be under capitalised. Why Fawnray? Why AIDC?

Piers Dick-Lauder

Basser Department of Computer Science
University of Sydney

Domain addressing in SUN III

Much work has gone into the implementation of domain addressing in SUN III. The introduction of domains has led to significant routing efficiencies, and enables the dynamic routing functions of SUN to be extended to cover a growth in the number of nodes to several thousand.

The presentation will cover the design, routing capabilities, and installation of domains, and, in particular, will discuss the actual domains to be used in ACSnet.

Ken J. McDonell

Department of Computer Science
Monash University

The Pyramid 90x: An Overview and Assessment

The Pyramid 90x is one of an emerging class of machines designed principally to run the UNIX operating system. The Pyramid architecture features a blend of novel RISC-based ideas, bit-slice componentry and third-party input-output subsystems. Together these form a machine whose base hardware is significantly faster than a DEC VAX 11/780. The more unusual aspects of the hardware will be presented, along with some insight into the potential for future speed enhancements.

Pyramid's UNIX port (OSx) is innovative in that it attempts to provide concurrent support for both AT&T's System V and Berkeley's 4.2BSD versions. This support relies upon an extension of the Berkeley symbolic link concept to conditionally map critical parts of the filesystem (e.g. /bin, /usr/bin, /usr/lib) onto different directories depending upon the system (System V or 4.2BSD) that a procss thinks it is running in. The kernel is 4.2BSD based, but all System V facilities have been provided for the kernel interface, the utilities and the libraries.

Like any new machine and/or UNIX port, the Pyramid has had its teething problems. An attempt will be made to outline the current state of the system based upon user experiences, and to indicate likely future developments. The performance of the current system relative to a VAX 11/780 running 4.2BSD will be discussed.

Des FitzGerald

Desmond FitzGerald and Associates P/L

An Interactive Graphics Mining Ore Reserves System

OREX is a computer based ore reserves system for underground metalliferous mines. OREX makes use of an amalgam of software techniques. Software Tools methods are employed, thus the code is written in Ratfor with one extra preprocessor pass added for backwards and forwards user prompting support. A relational data-base that mostly binds at compile time is also employed. Graphics routines were developed to support the display of pointed to data structures (e.g. polygons). The details of the low level graphics library are hidden from the application by a consistent set of macros. Linking to another graphics library does not require a rewrite, but simply the framing of a new set of macro definitions, and a recompilation.

This talk will give an introduction to the OREX system, and explain the motivation for the use of the software tools methedology.

Glenn Trewitt

Basser Dept of Computer Science
University of Sydney
(On leave from Stanford University)

Internetwork Protocols or How I Spent My Summer

This talk will present a comparison of some widely used internetwork protocols, including IP/TCP and XNS. Details of an implementation of the Xerox XNS protocol for System V may be presented.

Rob Pike

Bell Labs 2C-521
Murray Hill NJ 07974

cat -v Considered Harmful

The UNIX system provides a method of programming — the use of tools — that has been much discussed in books and the literature. As the system ages ("matures" is inappropriate) and is applied to new problems, the command set is modified and extended to provide new functionality.

The cat program is one of the simplest UNIX commands, and its various versions illustrate many of the issues that arise when designing programs for the UNIX environment. In particular, the **-v** option (make control characters visible) found in some versions brings out the difficulties that arise when new functionality is desired, such as the question of whether to write a new command or extend an existing one, and how to design the facility to fit best in the established environment.

The problems of dealing with output to 9600 baud asynchronous terminals also point out some of these issues. It is clear that all programs should not be modified to produce their output a screen at a time, but trying to centralize the solution so all programs have equal but transparent access to such a facility poses new problems. The "right" answer is hard to find, but the lessons from existing tools in the UNIX environment help decide what is right.

Greg Rose

Tata Elxsi

Some concurrecy issues in Multiprocessor UNIX

ENIX is a port of UNIX system V, under development in Sydney. This port runs on the ELXSI 6400 Multiprocessor Computer. For leadup, this talk will give an overview of the EMBOS operating system, and how ENIX worms it's way in. Some implementation details pertaining to UNIX on an architecture without a privileged mode will be discussed. The decoupling of the Enix kernel from the user processes it controls, and the asynchronous behaviour resulting therefrom, is mentioned. A general overview of some other interesting features of this port will also be touched.

Peter Ivanov

Department of Computer Science
University of New South Wales

A UNIX Network Information Database or How to waste more time than reading netnews

This talk will run through the history, current status and apparent trends in Australia's connection to, and use of, the world wide UNIX computer network. The history traces how the selfish acts of a few can lead to the profound confusion of many, and how other well meaning and intelligent people can achieve the same result. The implementation, use and maintenance of a UNIX network information database will be explained and an attempt will be made to sort out some of the confusion experienced by novice users in their efforts to contact friends and relations overseas.

A. J. Hurst

Department of Computer Science
Australian National University.

JAS – A Modula Based Single User System

JAS is a UNIX-like single user operating system implemented in Modula. It was designed to provide a suitable experimental environment for the development of software systems on a Burroughs B1700 architecture.

One essential aspect to the design of JAS was that it provide operating system primitives appropriate to the implementation of intermediate languages. Intermediate languages are used on architectures like the B1700 to provide a "soft" high level environment for the implementation of source languages such as Pascal and Modula, and facilities for compiler construction by migrating some complex run time mechanisms from the compiled code to the machine architecture. The generated code of the compiler is thus made simpler, at the expense of maintaining a microcoded architecture for that language via a suite of interpreters.

Operating systems for such environments must acknowledge the existence of such interpreters, and support both their construction and run-time environment through the provision of appropriate operating system primitives.

This paper describes the design of JAS, its implementation in Modula, and some comments (both objective and subjective) on the experience thereby gained.

T.R. Cordingley and D.W.E. Blatt

Department of Mathematics, Statistics and Computer Science
University of Newcastle, N.S.W.

A General Purpose Multiprocessor Kernel Written in C for a UNIX Programming Environment

The project described in this paper aims to build a high level language environment for user programming of a multiprocessor system.

To make this feasible with limited resources, the system is based on "off the shelf" board level microprocessor products and available software.

The kernel is being developed using the C programming language with assembler support for saving and restoring process environments, interrupt interfaces, and low level synchronization constructs. At the higher level UNIX-like interfaces will be maintained to the extent that is possible, so that existing applications in normal high level languages can be ported to the system easily.

The hardware requirement for the system is a multiprocessor system with shared memory for interprocess communication and process management. The i/o devices need no special configurations.

The kernel, unlike that of UNIX, is being implemented as a set of processes that can be linked in from an object library depending on whether they are needed in an application or not. These processes include device drivers, process scheduling and message buffering processes, real time clock handler and UNIX file system drivers. The basic idea is to make an application configure a kernel for itself through linking the appropriate processes at compile time.

Process synchronization and communication constructs are similar to those that are used in ADA. They will be maintained in libraries that are linked to the user application at compile time also, and are being implemented using explicit scheduling and semaphore primitives which are provided at the lowest level in the kernel. Initially program protection is to be maintained only by the mechanisms such as type constraints enforced by the compilers. Any memory management will have to be performed by the application. These are not important factors in this system because it is for a single application environment and normal multi-user problems will not occur.

Development is all being done in a UNIX environment and the system will continue to support UNIX in a single processor mode on a subset of the multiprocessor application hardware. The multiprocessor kernel is then bootable in the same way that UNIX is booted using the stand alone support programs. This maintains a high level of user support for the program development environment.

Y Kuang Oon

Dept. Community Medicine,
Monash University

Computer Records Considered Harmful under UNIX

Much lament has been heard about the lack of record handling under UNIX/C. There are no standard ISAM file facilities or even a file of records declaration as possible in Pascal. Whether it was an intentional design goal or a wondrous accident – this absence of record handling under UNIX/C encourages a new style of computer programming where computer records are considered archaic and harmful.

Much work in this department in the past three years involves the design and implementation of medical records. The mistake that has often been repeated (until PORTA came along) is to confuse a medical record with a computer record. The fixation on computer records is a curse on the computer industry for the simple reason that it locks data onto one computer system and makes portability of data a re-keying exercise. The portability of data is not only of relevance in the medical field – the portability problem comes in again each time you switch hardware of operating system. The handle on the portability problem of computer records is a **FILE OF ASCII CHARACTERS** or in essence, a UNIX regular file to represent the data you would normally hold in a computer record. The following are presented as further arguments for using the concept of data files instead of one file of records.

1) Massive savings in programming effort as files are accessed, deleted via system calls.
2) Logical organization of data files using hierarchical tree directories.
3) Reliability, security and data integrity – as solid as a computer file. We can rely on UNIX file access control for owner, group, and outsiders.
4) Damage control – lose a traditional file, you may have lost all the records.
5) Key to UNIX – data kept as a file has as a handle the file name.
6) File repair and integrity checks are system utilities.
7) Backup simplified by incremental dump.
8) The use of shell programming makes processing of files as convenient as the old computer record system.
9) Organization of data into files makes possible the use of data languages designed along lines or programming languages to lead to universal portability of data. Porta for medicine – Veta for vets? Denta for dentists?
10) Easy networking – file transfer across networks is a more solid proposition than transfer of a computer record.

Ron Baxter

CSIRO, Division of Maths and Stats

Review of MH - real communicators don't use mail.

MH is a message handling system that can be used instead of the more usual 'mail' commands. Some of the features that make it different are:

— instead of being a single command that then offers the user a series of options, MH provides a set of separate UNIX commands for reading, replying forwarding, scanning headings, and so on. This means that the user-interface is the shell so that aliases and shell scripts can be used to modify this interface

— instead of keeping multiple message in a single file, MH keeps each message in a separate UNIX file, This allows easy cross-referencing using links between files.

— messages can be stored in folders (directories) and then grouped into sub-folders (sub-directories).

The MH system seems to have many advantages for users who handle a considerable amount of mail, and where it is important to maintain a well organized record of correspondence.

John O'Brien

Fawnray Ltd.

Fear and Loathing on the 32016

A talk on the trials and tribulations of porting to the National Semiconductor 32016 (16032). If you like mystery and intrigue this talk is for you. Find out the nice features, the bad features, and those features which cause you to wish you had never started the project.

A brief outline of the 32000 chip family architecture, high-lighting the benefits and short comings of the design. A history of the development that was under taken by Fawnray. Why try and port anything besides UNIX itself. What tools are required to undertake this work with a reasonable degree of success.

The port has been completed and is currently on Beta site testing. Method of presentation will be the view held by the frustrated programmer. Frustration caused by trying to make a complex VLSI circuit behave as described in the data sheets. The treatment will be very light hearted, proving constantly that the programmer is always right, the machine is just stupid.

Domain Addressing in SUN III

Bob Kummerfeld

Piers Lauder

Sydney University

ABSTRACT

A form of hierarchical addressing employing the concept of domain has been introduced in SUN III. Domains can be equivalent to geographical addresses, and allow a uniform presentation of addresses that are universally valid.

1. INTRODUCTION

Computer message addresses are changing. As computer networks are interconnected to form a world-wide net, there is a need for computer message addresses that will be as universally valid and as easy to use as postal addresses or telephone numbers.

Until now, those of us using the Sydney Unix[1] Network[2] have been living in a simple community, as though everyone lived in the same street and it was only necessary to know people's names and house numbers to identify them uniquely. Now a city has grown up around us, with links to other cities and countries, and street addresses are no longer unique.

The concept of a domain has been introduced[3] to describe the arrangement of addresses within computer networks. A domain is a grouping of computers to support a common purpose, such as all those computers running a particular type of network software, or all the computers in a university campus, or all the computers in a particular country. Domains enable us to generate a computer network address that can not only be used to deliver a message to a local destination, but will also deliver a message to the same destination from anywhere in the world.

For example, the following table shows a traditional postal address with the components distinguished by separate lines, together with the equivalent components from a network address:

Piers Lauder	piers
Computer Science	cs
Sydney University	su
Australia	oz

1. UNIX is a trademark of AT&T Bell Laboratories.

2. The latest is version III and will be referred to as SUN III.

3. ARPA RFC 882

When presented to the network, the address would appear as

piers.cs.su.oz

(the line separators have been replaced by periods.)

Previously, the network address for this destination would have been

piers:basser

which has only two components and is probably easier to remember, but is only valid in Australia on SUN sites.

The **:basser** is a node identifier-the name of a particular computer-and it is not something everyone should be required to remember.

There are potentially thousands of network nodes in Australia alone, and for this address form to work, there must be some data-base that knows how to deliver a message to every one of them.

By contrast, for the domain addresses to work anywhere in the world, a local data-base need only retain knowledge of the routes to those major domains outside, and to each of the domains contained within, its own domain. An analogy is telephone switching, where each exchange need only remember its own local numbers, and know how to switch calls out to other area codes.

We have sacrificed some ``user friendliness´´ to reduce a computing task, but consider the problem of communicating with other continents. Here is the address that must be used to reach a colleague in California at UC San Diego Computer Science from anywhere in Australia:

decvax!ucbvax!sdcsvax!sdamos!john:mulga

Here is a more arcane example:

decvax!ucbvax!@SU-SCORE.ARPA:john%unc.csnet@csnet-relay:mulga

This routes first via SUN to **mulga**, then to **decvax** and **ucbvax** by UUCP, then by CSNET to **csnet-relay**, then to **unc.csnet** using CSNET, which then sends to SU-SCORE via ARPANET for delivery to user **john**. Caveat sender.

The domain version of this would be

john.SU-SCORE.ARPA.CSNET.UUCP

but this just reproduces the problems of explicit routing, using domains instead of nodes. What would be best of all is the domain version of the postal address:-

john.cs.stanford.usa

Although addresses have become slightly more complex for communicating within Australia, they will become much less complex for the general case. And of course there are abbreviated forms for addresses within a local domain. In the same way that a letter addressed to someone in Sydney from someone in Melbourne need never specify ``Australia´´, a message from Melbourne to Sydney

need not specify ``.oz''. So, for example, here are the addresses needed to reach the author from various places:

Message source	Address
Computer Science, Sydney University	piers
Psychology Department, Sydney University	piers.cs
Computer Science, Melbourne University	piers.cs.su
Bell Laboratories, NJ, USA	piers.cs.su.oz

The domains taking part in this form of addressing are known as hierarchical domains. That is, each domain is fully contained within another, and can be considered as a geographical mapping. At any one level, there is a domain to describe every area in the world. There are also non-hierarchical domains. For instance, all those sites taking part in the Australian Computer Science Network belong to the domain ``acsnet'', and this domain crosses geographical boundaries. One use of non-hierarchical domains is to consider them as special interest groups, and one may, for instance, address messages to everyone within them. Thus the address

netgurus.*.acsnet

will deliver a message to the identity ``netgurus'' at every site belonging to the domain ``acsnet''.

2. DOMAIN ROUTING MECHANISM

The underlying transport mechanism of SUN III knows how to deliver messages to handlers at nodes. Nodes are linked to neighbouring nodes to form a network. Nodes correspond to computers, and handlers embody protocols that can address users.

SUN III implements domains by attaching a list of accessible domains to each node. The route to a domain is therefore the same as the route to the nearest node (in the network routing sense) in that domain. Nodes are the lowest level in the domain hierarchy, and can be considered as domains with one member.

2.1 Primary and Secondary Domains

While a node may belong to many domains, it is considered to have one primary domain which specifies the set of nodes from which this node will be directly addressable. Any other domains are secondary and specify all other domains whose members will be directly addressable from this node.

Some of the domains will be fully contained within others, as a Computer Science Department is contained within a University, which is contained within a country. These domains, one of which must be the primary, are considered to belong to a local hierarchy. All other domains to which the node belongs are considered to be non-hierarchical. The hierarchy nominates which domains are sub-domains, each domain in the hierarchical list being completely contained within its successor.

The hierarchy is used to detect sub-domains in domain lists presented from other nodes, and to reject different domains with the same name. An obvious example of this problem arises where there are two local domains

called ``cs´´, but one is Computer Science at Melbourne University, and the other is Computer Science at Sydney University.

2.2 Source Address Construction

As messages proceed through the network, they may cross domain boundaries. At each boundary, it is necessary to augment the source address so that the message (or a reply) may be correctly returned to its source.

The routing mechanism at each node notices when a new destination does not belong to the same primary domain as the source (or most recent node in the route), and appends an appropriate part of the local domain hierarchy to the source address. Every time a message arrives at a node, the source address is scanned from the right hand end, and each domain to which the local node belongs is removed. In this manner, at any one node in the route, the source address always contains the minimum set of domains to identify the source uniquely.

An example is given by the progress of a message addressed to a node ``snb´´ in the domain ``btl´´ inside ``usa´´ from the node ``psych23´´ inside the domains ``psych´´, ``su´´ and ``oz´´:

Node	Domains		Addresses	
	Primary	Hierarchy	Source	Destination
psych23	psych	psych.su.oz	psych23	snb.btl.usa
psych44	su	psych.su.oz	psych23.psych	snb.btl.usa
basser40	su	cs.su.oz	psych23.psych	snb.btl.usa
basser	oz	cs.su.oz	psych23.psych.su.oz	snb.btl.usa
research	usa	btl.usa	psych23.psych.su.oz	snb
snb	btl	btl.usa	psych23.psych.su.oz	

The source address is built up at ``psych44´´ from where the message crosses out of the domain ``psych´´ into the domain ``su´´, and at ``basser´´ from where the message crosses out of the domains ``su´´ and ``oz´´ into the domains ``usa´´ and ``btl´´. Notice that as each destination domain is reached, it is removed from the destination address.

3. INTER-NETWORK GATEWAYS USING DOMAINS

Most of the problems with current addressing mechanisms for computer messages stem from the interfaces required at gateways between different networks. The SUN III domain mechanism allows gateway handlers to be attached to the name of a domain at a node. Then instead of being delivered locally, a message addressed to such a domain will be passed to its handler. The handler may massage the message and its address to conform to the new network, and pass it on.

Thus the domain ``UUCP´´ is considered to include all the nodes that communicate via the UUCP network. Those nodes with interfaces to the UUCP network from the SUN network will attach a gateway handler to the domain ``UUCP´´, and pass messages from SUN to UUCP via the handler.

From SUN, the gateways to UUCP are identified by their declared membership of the domain ``UUCP'', and messages may be routed to the nearest node supporting the gateway.[4] A simple example of an address that would make use of a domain gateway is

john.decvax.UUCP

Such an address form is an expediency, pending the introduction of geographic domain addressing on the other network.

4. ROUTING EFFICIENCIES

In practice, the introduction of domains has reduced the number of nodes visible at the highest level by a factor of two. With $O(n^2)$ routing algorithms, this is a great saving. Better still, for nodes at lower levels, the amount of routing information has been cut by up to a factor of ten, making network membership a much more affordable option.

There is another benefit in the area of network topology maintenance. A broadcast address is defined as one that delivers its message to every node in the primary domain of the sender (unless overridden by explicitly naming a domain for the broadcast). This mechanism considerably reduces network routing traffic, which uses broadcast addressing to distribute topology changes, as messages that previously were sent to every node may now be confined within the relevant domain.

4. Of course, the gateway handlers must be fairly intelligent about manipulating the UUCP addresses to correspond to the alternate points of entry to an explicitly routed network, but that's another story.

;login:

The USENIX Association Newsletter

Volume 9 Number 3 **July 1984**

CONTENTS

The deadline for submissions for the September issue of *;login:* is September 4

UNIX and Computer Graphics Workshop

December 13—14, 1984

DoubleTree Hotel
Monterey, CA

USENIX is sponsoring a limited-enrollment workshop on current and future developments in interactive computer graphics under UNIX, including:

- Large scale graphics databases
- Real-time implementations
- UNIX as a graphics development environment
- High speed data transfer
- Future developments and directions.

The workshop will be structured to facilitate in-depth discussions of technical issues, and will have presentations in a number of formats, with ample time for questions and responses. There will be a computer graphics film and video presentation on Friday night.

In order to cover the extra expenses entailed in providing high quality visual presentations, the registration fee will be $200, which will include a reception. The hotel rate for this conference is a special $65/night for either single or double occupancy.

For further details and application information, contact the Program Chair:

Reidar J. Bornholdt
Room 7-444
Columbia University
College of Physicians & Surgeons
630 West 168 Street
New York, NY 10032

{harpo,cmcl2}!cucard!reidar or {ucbvax,decvax}!usenix!reidar

Program Committee:

Reidar Bornholdt, Columbia University, Chair
Lou Katz, Metron Computerware
Tom Duff, Bell Laboratories
Peter Langston, Lucasfilm Ltd.

Communications and Networking Workshop

October 11—12, 1984

Golden, CO

USENIX is sponsoring a limited-enrollment workshop on current and future developments in communications and networking aspects of UNIX.

The program chair is:

Doug McCallum, NBI Inc.
nbires!mccallum

Further information on this workshop will be available through the USENIX office and will be posted to net.usenix.

Future Meetings of the USENIX Association

The Winter 1985 Conference of the USENIX Association will be held January 23–25, 1985, at the Fairmont Hotel in Dallas, Texas. This conference is being held in the same city and at the same time as the /usr/group sponsored trade show, UniForum. Arrangements for cross-registration for those wishing to attend both events are being discussed with /usr/group.

The Summer 1985 Conference will be held June 11–14, 1985, in Portland, Oregon.

Future Meetings of Other UNIX Users Groups

Australian UNIX-Systems Users' Group

The 1984 winter meeting of the AUUG will be held in the Department of Computer Science at the University of Melbourne on August 27 and 28, 1984. An exhibition of UNIX-related computer hardware and software will be held in conjunction with the meeting. The conference dinner will be held Monday; reservations for it must be made with advance registration. A limited number of rooms are available in the University Colleges. The meeting cost will be $30 for advance registrants, $50 on site. For more information, contact

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

or

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

European UNIX Users Group

The EUUG and /usr/group are sponsoring a conference and exhibition at St. Catherine's and Pembroke Colleges, Cambridge University in the United Kingdom on September 19–21, 1984. The invited speakers are: John Lions, Steve Bourne, Pier Dick-Lauder, Tom Killian, Mike Karels and Sam Leffler.

The entrance fees are: technical sessions only – UK*L* 90 (students 70); technical and industrial sessions – UK*L* 120 (students 100); plus 15% VAT. USENIX members pay the same rate as EUUG members. Nonmembers fees are UK*L* 40 additional.

Accommodations are available at the College Residents. Registration and requests for accommodations should be received by the EUUG Secretary before September 3. For information contact:

Mrs. Helen Gibbons
Owls Hall
Buntingford, Herts. SG9 9PL United Kingdom
Phone: 0763 73039

System V Performance Enhancements

Ken Goodwin

Senior Systems Programmer
N.J. State Medical Underwriters, Inc.
2 Princess Road
Lawrenceville, NJ 08648

Described below are a series of modifications made to the UNIX System V kernel and several commands. Changes to the kernel have resulted in a 12% decrease in system overhead, while command changes have increased usability and security. All changes were made on a DEC PDP-11/70 minicomputer, but are applicable to any UNIX system. Percentages and times were obtained using the *clock* subroutine under single user conditions.

User Level DMA I/O

A facility that allows user programs to specify Direct Memory Transfers for disk operations involving **regular** files has been implemented. It is enabled through a bit flag in the *open* or *fcntl* system calls. DMA operations may occur for any read or write operation that begins on physical disk block boundaries and whose transfer size is at least the same size as a physical disk block. However, no restriction is placed on the user program to insure that its transfers meet these criteria. As the I/O request is broken down into transfers to specific blocks by *bmap*, a decision is made as to whether a DMA transfer can take place from/to the disk block. If a DMA transfer can not take place, either because the transfer does not start at a zero block offset or the count of bytes remaining to be transferred is less than the physical size of the block, then the transfer is done through the Buffer Cache as before. If a DMA transfer can take place, then a call to the *dma* subroutine is made. The *dma* subroutine checks the request for validity, initiates the I/O operation asynchronously through the *phybio* subroutine, a modified version of the standard *physio* subroutine, and then updates the count, offset, and base fields in the Per Process Data Area (_u). While this transfer takes place, *dma* checks to see if a Read-Ahead transfer can also be done. This is determined by the presence of a read-ahead block number in *dma*'s parameter list and if the updated count field is still greater than or equal to the physical size of a disk block on the filesystem. If read-ahead can be done, another asynchronous I/O operation is queued through the *phybio* subroutine and the count, base, and offset fields are again updated. *dma* then waits for all the I/O requests that it has initiated to complete. On errors, the count, offset, and base fields are backed up to the point they were at before the error occurred and u_error is set. In this way, up to two disk transfers can be completed per invocation of the *dma* subroutine. This has the effect of minimizing the number of *dma* calls performed and halves the number of loops performed within the *readi* and *writei* subroutines. Changes required to implement DMA I/O were minor. They consist of:

1. A modified version of *physio* (*phybio*) which permits asynchronous DMA queueing of I/O requests between the Disk and user I-Space, D-Space, Kernel I-Space, Kernel D-Space, Supervisor I-Space, or Supervisor D-Space.
2. A *phywait* subroutine implements the *iowait* and *error* functions from the *physio* subroutine.
3. A *dma* subroutine which handles the queueing of the requests, updating the relevant fields, and dealing with error conditions.
4. Changes to the *readi* and *writei* subroutines which allow them to determine if DMA I/O is desired and permitted.

5. A single line change to *bmap* to turn off Read-Ahead calculations if DMA I/O has been selected and the next block in the file is the last block and is partially filled.

This facility has been added to the *cp*, *dd*, and *cpio* commands. For any filesystem using 512 or 1024 byte disk blocks, the performance increase for programs using this facility is approximately twenty-five percent. It also has the added advantage that corruption of the buffer cache by these programs no longer occurs and cache hit ratios thereby increase for other programs. This facility is not for use by every program. Candidates for DMA I/O include programs such as *cp* which never reference a disk block more than once. A possible modification to the stdio library to allow the selection of DMA I/O operations is anticipated. Since stdio already does buffering, there is little need in most cases for additional buffering by the kernel. Higher performance increases are possible with larger physical block sizes. A port of the algorithm to a Berkeley 4.2BSD system is planned.

Faster *exec* System Call

The *exec* system call now uses the new DMA I/O facility to directly load programs into their process space. A great deal of memory-to-memory overhead is thereby avoided. The performance increase for *exec* is greater than thirty percent. Since *exec* does not have the system call overhead of the standard *read* system call, it gains a greater performance improvement than a user level DMA operation. No actual timing comparisons were made to calculate the performance increase for *exec*. For user level DMA I/O requests, each doubling of the read count resulted in a one percent performance improvement. This has been extrapolated to yield the thirty percent figure based on the analogy that an *exec* call translates to a single *read* call of process-size bytes. The performance increase may therefore be much greater.

Improved Read-Ahead

A change to the standard method of calculating read-ahead for files has been implemented. On standard UNIX systems, the last logical block referenced in a file is stored in the incore inode structure as i_lastr. Although this is quite adequate for One-Program One-File scenarios, it has the tendency to turn off read-ahead all together when more than one process is referencing the same file. The *bmap* subroutine determines that read-ahead should be initiated if i_lastr+1 is equal to the current block that the process wants (i.e., the file is apparently being read sequentially). If two processes are reading the same file sequentially, but are several blocks apart in the file, then read-ahead is turned off for the process that is further along in the file. This is due to i_lastr being flip-flopped between the last block read by each process, such that i_lastr+1 never equals the block number desired by either process. The trailing process gains only partial read-ahead. The amount of this read-ahead depends on how fast other processes gobble up buffers containing the blocks that this process has yet to reference and how much further along in the file the leading process is. If the leading process is more than NBUF blocks ahead of the trailing process, then read-ahead will be totally turned off for both processes.

This standard algorithm has been modified to allow read-ahead calculations to be computed on a per-process basis. The i_lastr disk address has been replaced by a u_lbr disk address and an array of disk addresses u_lastr[NOFILE] in the per-process data area (_u). The *rdwr* subroutine has been modified to copy the contents of u_lastr[FD] to u_lbr before calling *readi* or *writei* and to copy u_lbr back to u_lastr[FD] when *readi* or *writei* returns. The *dup* and *fcntl* system calls now copy u_lastr[OLDFD] to u_lastr[NEWFD] and *open* sets u_lastr[FD] to zero. The *bmap* subroutine now uses u_lbr instead of i_lastr. This modification has the added advantage of freeing up much needed data space since the Per Process Data Area is not a permanent physical part of kernel data space while i_lastr was. The amount of data space released is NINODE*sizeof(daddr_t). The *exec* system call sets u_lbr to zero. This is done to allow read-ahead to proceed if DMA *exec*'s are not used. There is no performance change for the One-Process One-File scenario, but this change increases the read-ahead performance of the Many-Process One-File scenarios that are an integral part of large scale database systems.

Process ID Map

The standard Process ID (PID) allocation method in UNIX is to compare the desired new process id against the process ids already allocated in the Process Table. At best, this involves doing (v_eproc−&proc[0]) comparisons during the first MAXPID processes. On any system which manages to age past MAXPID processes, the number of comparisons performed greatly increases. This is due to PID collisions that begin to occur which make the code branch out of the process table scanning loop in order to select an alternate PID. This new PID may also collide.

This allocation method has been replaced by a Process ID map strategy. The process table scanning is now reduced to a single linear search to locate an empty slot and to count up the number of active processes associated with the Process group. This eliminates the PID retry code above the process table scanner and an IF−GOTO statement within the process table scanner. This IF−GOTO statement consumes about 2.7 microseconds on a PDP-11/70.

Originally, there were two process ID maps. One was used to hold process IDs available for use and the other was used to hold process IDs that had already been used. Two pointers were used to reference one as a New PID map and the other as an Old PID map. These pointers were initialized during system generation.

```
struct map pid1[PMAPSIZ] = {mapdata(PMAPSIZ)};
struct map pid2[PMAPSIZ] = {mapdata(PMAPSIZ)};
struct map *newpid = &pidmap1;
struct map *oldpid = &pidmap2;
```

The newpid map was initialized during system boot by an *mfree*(newpid, MAXPID, 1) call in *main*. Since the size of items allocated and freed in the process ID maps is always of size one and in order to minimize overhead, special versions of the *malloc* and *mfree* subroutines were created. They are called *pid_alloc*(map) and *pid_free*(map, pid). The *exit* system call freed used PID's into the oldpid map when a process died, *pid_free*(oldpid, p_pid). This insured that new process IDs remained unique for as long a time as possible. The *newproc* subroutine used *pid_alloc*(newpid) to allocate a process ID when creating the new process. When the newpid map was exhausted, a switch of the newpid and oldpid pointers was made. This moved the contents of the oldpid map into the newpid map and reinitialized the oldpid map.

```
struct map *swmap;

swmap = newpid;
newpid = oldpid;
oldpid = swmap;
```

Another *pid_alloc*(newpid) was then done and should succeed. A panic situation was generated if this *pid_alloc* failed. However, the only way this could occur is if something overwrote the maps or if the *pid_free* in *exit* could not free a significant number of old PIDs into the oldpid map because PMAPSIZ was too small. PMAPSIZ had to be roughly 30% greater than NPROC to allow for long term fragmentation.

For oldpid and newpid maps which were fragmented and contained around seventy entries, the break-even point for this algorithm was 66 active processes in the process table. If standard *malloc* and *mfree* subroutines are used to manage the PID maps, then the break-even point rose to 71 active processes. This break-even point was only for the first MAXPID processes. It declined once MAXPID was exceeded since collisions would begin to occur at this point under the old algorithm. This algorithm was obviously not for small UNIX systems.

Note: two microseconds were trimmed off the average *malloc* subroutine call by changing the *malloc* code to put the size parameter in a register.

The times below are in microseconds as computed from *clock* subroutine calls. The tests were run single user and are adjusted for extraneous times caused by instructions used to run the timing

tests. "Frag." is short for fragmented and indicates that test was run on a fragmented map of about 70 members. If not specified, then the test was run on an unfragmented map. "Alloc" is for a *malloc* or *pid_alloc* call and "Free" is for *mfree* or *pid_free* calls. "Breakeven" is the number of active processes that must be present for the map strategy to break-even over the old algorithm during a single scan of the process table. Breakeven = Total Frag. / 2.7μs. "Aver." indicates the average overhead of making the subroutine call on an unfragmented map. The column headers are: "Old" – using standard *malloc/mfree* subroutines, "New" – using improved *malloc* subroutine, "Pid" – using *pid_alloc/pid_free*.

Test Times

Test	Old	New	Pid
Aver. Alloc	34.8	32.8	29.8
Aver. Free	47.75	47.75	42.19
Alloc Frag.	35.54	34.43	30.53
Free Frag.	153.84	153.84	146.6
Total Frag.	189.38	188.27	177.13
Breakeven	70.14	69.73	65.6

Note that the overhead of a free operation on a fragmented map is seventy eight percent of the total time in this strategy. This was deemed unacceptable and the problem was subjected to intense analysis. The above PIDMAP algorithm was modified as follows. The *pid_free*(map, pid) was removed from *exit* since this represented seventy eight percent of the overhead. The data structures were changed to:

```
struct map pidmap[MAPSIZ] = {mapdata(PMAPSIZ)};
int usedpids[NPROC+2];
```

PMAPSIZ should now be NPROC+4. The *pidmap* is still initialized in *main* by a *mfree*(pidmap, MAXPID, 1) call. *newproc* calls *pid_alloc* to allocate a process id (p->p_pid = pid_alloc();). *pid_alloc* now does all the dirty work. If the pidmap has not yet been exhausted, then it returns the next available PID. Otherwise, it regenerates the pidmap from the contents of the process table. The process table is scanned and every p_pid that is greater than or equal to STARTPID is placed into a slot in the usedpids array. STARTPID is a define for the first PID which is considered in regenerating the pidmap. It is set to two on my system since PID number 1 is always in use by *init*. It represents the lower bound for the range of recurrently usable PIDs. MAXPID is the upper bound for this range. The usedpids array is then sorted via a shell sort and the sorted contents used to generate the correct entries in the pidmap. Since this operation is only performed once in (MAXPID−NPROC) process generations, great speed was not deemed necessary. Once the pidmap has been regenerated, then *pid_alloc* returns the next available PID as before. This modification resulted in an eighty three percent performance increase over the previous pidmap strategy.

The following test results were obtained by simulating the algorithm under three types of loads. "WC" is a worst case scenario: NRPROC was 400, the number of filled slots in the process table was 350, and ve_proc was set to the address of proc[NPROC] so that every process table slot was examined. "BC" is a best case scenario: NPROC was set to 200, the number of slots filled was 150, and ve_proc was set to the address of proc[150]. "NC" is a normal case scenario which simulates the tests run under the original PIDMAP strategy by forcing the regenerated pidmap to have seventy entries: NPROC was set to 100, number of processes was 70, and ve_proc was set to the address of proc[70]. For all tests, the process table was initialized with non-sequential decreasing value PIDs. This was done so that the maximum size pidmap would result and the sort would be forced to totally invert the contents of the usedpid array. Because of this, the results below should be higher than what would be obtained in actual use. Under live conditions there should be many sequential PIDs in the process table.

"Regen Time" is the time necessary to regenerate the pidmap once it has been exhausted. "Alloc Time" is the average time necessary to fetch a pid from the pidmap while it still has entries in

it. "RT-pid" is the average time to regenerate the pidmap per pid gained (Regen Time / (MAXPID−nproc)). "Breakeven" is the number of process table entries that have to be occupied in order for the algorithm to breakeven with the original linear search. Breakeven = (RT−pid+alloc. time) / 2.7μs.

Test Times

Test	WC	BC	NC
Regen Time	33294.65	16638.65	16640.65
Alloc Time	37.35	27.35	25.35
RT-pid	1.12	.56	.56
Breakeven	14.25	10.34	9.6

Note that the average breakeven point is ten processes in the process table. Small UNIX systems can now take advantage of a PIDMAP algorithm. Also, this new strategy consumes less data space than the original PIDMAP strategy. In case the high Regen times on the pidmap are causing you concern, please note the following. Suppose we assume that we maintain ninety processes in the process table during the first MAXPID processes and the breakeven point is ten. Also, because PIDs have yet to wrap around, we will experience no PID collisions. This means that each *newproc* call will be saving 80 times 2.7 microseconds of CPU time for a total of 216 microseconds (90 processes minus 10 required to breakeven). With MAXPID set to 30000, which is standard for System V, this means we will use up 30,000 PIDs before exhausting the map and forcing a pidmap regeneration. This means that the total CPU time saved before regeneration is 30,000 times 216 microseconds, or 6,480,000 microseconds. After the first regeneration we will be ahead of the original algorithm by over 6,463,359.35 microseconds.

One final note, this is not the optimum PID allocation algorithm. The optimum algorithm could not be implemented on my system as it would require 3,752 bytes of data space. Such data space is hard to come by in a PDP-11/70 System V kernel.

Optimum PID Allocation Strategy

The above algorithm can be vastly improved by replacing the memory map structure (struct map pidmap) with a **bitmap** where each bit represents the current status of a PID. An **off** bit state would indicate that the PID was either in use or had been recently discarded by *exit*. A **on** bit state would indicate that the corresponding PID was available. The bitmap is represented by an array of integers.

```
# define BPINT      16        /* number of bits in an integer */
# define BITON      0177777   /* All bits in an int on (can be -1) */
# define PMAPSIZ    ((MAXPID + (BPINT − 1)) / BPINT)

int       pidmap[PMAPSIZ];
int       *pidoff;
```

pidoff is an integer pointer used to indicate the first non-zero slot in the pidmap array so that a linear search will not be necessary. Initialization and regeneration are now easy. First one turns on all bits in every integer, then one clears the bits that correspond to PIDs already in use (including PID zero), and finally pidoff is set to the first non-zero entry of pidmap. To select a PID, the contents of the word pointed to by pidoff are scanned for an **on** bit. This bit is cleared and the new pid is

((pidoff − pidmap) * BPINT) + (bit offset in *pidoff of cleared bit)

If *pidoff is now zero, then pidoff is incremented to the next non-zero entry. When pidoff exceeds the upper bound of the pidmap array, then the map is regenerated as described above. Most of the above can be implemented as macro calls. Since the entire algorithm can be implemented as in-line code in *newproc* the subroutine overhead is eliminated. Also, no sort is required. This should therefore be the fastest algorithm.

Dialup Line Security

getty

The *getty* command has been modified to allow a parenthesized ENVIRONMENT list to be included as part of it's command line. This was done in order to ease implementation of the dialup security feature in *login*. The TERM environment variable is set for each *getty* line in the *inittab* file for the specific terminal connected to the line. For dialup lines, the TERM variable is set to dialup.

```
/etc/getty ... '(' TERM=vt100 TERMDEV=/dev/tty00 ')' ...
/etc/getty ... '(' TERM=dialup TERMDEV=/dev/tty01 ')' ...
```

For environments with various terminal types, this allows naive users to float from terminal to terminal without worrying about the type of terminal they are logging onto.

login

The *login* command now passes the environment from *getty* down to the shell. It also examines the TERM variable to see if it is a dialup. If TERM is a dialup, then a dialup line security system is invoked. This change replaces an undocumented security feature that was present in the original System V *login*. An */etc/dialups* file which has the same format as the */etc/group* file has been created. It is divided into four colon-separated fields. The first field is the device name of a phone line (*/dev/tty*??). The second field is an encrypted password field, the third field is a privilege level field, and the last field is a comma separated list of login names for users permitted to use this dialup line. If TERM is dialup, then TERMDEV is compared against each line in this file. If no match is found, then no security check is done. Otherwise, the name (*/dev/tty*??) of this line is displayed to help in answering the password or in determining why permission to login was denied. If the password is present, then the user is prompted for the password to this line. If the privilege field is present, then the user's privilege level in the */etc/passwd* file (currently group number) must be less than or equal to the privilege number in the dialups file for this line. If the list of users (field four) exists, then the user's login name must be on that list. These features may be used in any combination. If the user fails any of the security checks, then a permission denied notice is printed and the system hangs up the phone line. This can make it very expensive for someone to try to break into the system. If the user succeeds in passing all the dialin security checks, then *login* proceeds with the user name and password checks. At maximum protection level, a person must know the name and password for an account on the system, must know the password for the dialup line that the person is authorized to use, and must know it's telephone number. Combined with IMM modem security devices, this system provides a more formidable obstacle to intruders than standard UNIX systems. It also allows administrators to assign dialups to particular users or groups to prevent one group from hogging the phone lines.

A modified version of the *passwd* command handles passwords for both the *group* and *dialups* files. Modified versions of the *group* file subroutines provide access to the *dialups* file.

A *dialname* shell command file is called by the shell during the login procedure if TERM is dialup. This shell allows naive users to set their terminal type (TERM) when dialing into the system in a user-friendly manner. It prompts for the *termcap* terminal type and compares it against a known list of terminals attached to the system. If the terminal specified is not on the list, the user is allowed a second chance to change the entry. This second chance is not checked and the entry is assumed to be correct. This allows users using terminals not known to the system to log in properly. If done frequently, then the System Administrator should be requested to add the terminal type to the known terminal list. A RETURN entered to the prompt generates a two column list of known terminals and their *termcap* equivalents.

Trivia Quiz

The following quiz was distributed at the Salt Lake City conference by Rob Pike. Prizes were awarded to the people with the most correct answers. The submission with the most correct answers (60) was from I. P. Stubbies (at team comprising David Tilbrook, Sam Leffler, and presumably others). Since they used a silly name and tried to put one over on the judges, they got a silly prize: a trophy labeled "world's best kibitzer." Jim McKie had the best score for an individual (57) and was awarded an authenticated 1972 DECtape containing UNIX Version 2. Finally, Ron Gomes had 56 correct answers and received an original engraved "Bill Joy" badge, which once belonged to Bill himself, from Sun Microsystems.

The answers to this quiz will appear in the next issue of *;login:*.

1. The source code motel: your source code checks in, but it never checks out. What is it?
2. Who wrote the first UNIX screen editor?
3. Using TSO is like kicking a [what?] down the beach?
4. What is the filename created by the original *dsw*(1)?
5. Which edition of UNIX first had pipes?
6. What is -=O=-?
7. Which Stephen R. Bourne wrote the shell?
8. Adam Buchsbaum's original login was sjb. Who is sjb?
9. What was the original processor in the Teletype DMD-5620?
10. What was the telephone extension of the author of *mpx*(2)?
11. Which machine resulted in the naming of the "NUXI problem"?
12. What customs threat is dangerous only when dropped from an airplane?
13. Who wrote the Bourne shell?
14. What operator in the Mashey shell was replaced by "here documents"?
15. What names appear on the title page of the 3.0 manual?
16. Sort the following into chronological order: a) PWB 1.2, b) V7, c) Whirlwind, e) System V, f) 4.2BSD, g) MERT.
17. The CRAY-2 will be so fast it [what?] in 6 seconds?
18. How many lights are on the front panel of the original 11/70?
19. What does FUBAR mean?
20. What does "joff" stand for?
21. What is "Blit" an acronym of?
22. Who was rabbit!bimmler?
23. Into how many pieces did Ken Thompson's deer disintegrate?
24. What name is most common at USENIX conferences?
25. What is the US patent number for the setuid bit?
26. What is the patent number that appears in UNIX documentation?
27. Who satisfied the patent office of the viability of the setuid bit patent?
28. How many UNIX systems existed when the Second Edition manual was printed?
29. Which Bell Labs location is HL?

30. Who mailed out the Sixth Edition tapes?
31. Which university stole UNIX by phone?
32. Who received the first rubber chicken award?
33. Name a feature of C not in Kernighan and Ritchie.
34. What company did cbosg!ccf work for?
35. What does Bnews do?
36. Who said "Sex, Drugs and UNIX"?
37. What law firm distributed Empire?
38. What computer was requested by Ken Thompson, but refused by management?
39. Who is the most obsessed private pilot in USENIX?
40. What operating system runs on the 3B-20D?
41. Who wrote *find*(1)?
42. In what year did Bell Labs organization charts become proprietary?
43. What is the UNIX epoch in Cleveland?
44. What language preceded C?
45. What language preceded B?
46. What letter is mispunched by *bcd*(6)?
47. What terminal does the Blit emulate?
48. What does "trb" stand for (it's Andy Tannenbaum's login)?
49. allegra!honey is no what?
50. What is the one-line description in vs.c?
51. What is the TU10 tape boot for the PDP-11/70 starting at location 100000 (in octal)?
52. What company owns the trademark on Writer's Workbench[tm] Software?
53. Who designed Belle?
54. Who coined the name "UNIX"?
55. What manual page mentioned Urdu?
56. What politician is mentioned in the UNIX documentation?
57. What program was *compat*(1) written to support?
58. Who is "mctesq"?
59. What was "ubl"?
60. Who bought the first commercial UNIX license?
61. Who bought the first UNIX license?
62. Who signed the Sixth Edition licenses?
63. What color is the front console on the PDP-11/45 (exactly)?
64. How many different meanings does UNIX assign to '.'?
65. Who said, "Smooth rotation butters no parsnips"?
66. What was the original name for *cd*(1)?
67. Which was the first edition of the manual to be typeset?
68. Which was the first edition of UNIX to have standard error/diagnostic output?

69. Who ran the first UNIX Support Group?
70. Whose Ph.D. thesis concerned UNIX paging?
71. Who (other than the obvious) designed the original UNIX file system?
72. Who wrote the PWB shell?
73. Who invented *uucp*?
74. Who thought of PWB?
75. What does *grep* stand for?
76. What hardware device does "dsw" refer to?
77. What was the old name of the "sys" directory?
78. What was the old name of the "dev" directory?
79. Who has written many random number generators, but never one that worked?
80. Where was the first UNIX system outside 127?
81. What was the first UNIX network?
82. What was the original syntax for "ls -l | pr -h"?
83. Why is there a comment in the shell source "/* Must not be a register variable */"?
84. What is it you're not expected to understand?

EUUG
European UNIX† Systems User Group

Newsletter Vol 4 No 1
Spring 1984

† UNIX is a Trademark of Bell Laboratories.

European UNIX System User Group Meeting
Faculty of Science, University of Nijmegen, 16/18 April 1984

Peter Collinson
Secretary

Introduction

This was a very good conference, and if you missed it, then you missed a large number of very good talks, the biggest exhibition at a EUUG conference to date and, of course, some Dutch beer.

This report is being done from my incomprehensible notes and mostly from the abstracts which were submitted before the conference began‡. I must confess to missing some of the sessions; still, that is always going to happen. I also feel that this report is not up to the usual standard because I am doing it too long after the event and this makes it difficult to precis the talks. So, where I am in doubt I have kept quiet. If have got any names wrong, then I am sorry.

However, the intention is to publish a full conference proceedings in the near future. Meanwhile, this can serve as a summary of what happened.

Day 1 - 16th April 1984

Emrys Jones officially opened the conference and chaired the first session.

Item 1: 10.00am **Michael J. Kelly, AT&T/Teletype Corp**

An Intelligent Windowing Graphics Terminal for the UNIX system

Abstract

An important feature of the UNIX System is per-user multiprogramming; that is, each user may control several concurrently executing processes. However, this feature breaks down at the user interface. UNIX systems rely on "dumb" or semi-smart terminals as the primary user interface, and these are not able to maintain several concurrent interfaces. The solution found in BSD, job control, still does not adequately solve the problem of maintaining display contexts for concurrently executing processes.

This talk was about AT&T 5620, the Teletype Corp's version of the Blit terminal. It is a workstation with a high resolution green screen, a mouse and has a reasonably powerful machine to drive it. The machine does not run UNIX because the device is a terminal and not a working CPU in its own right. The processor is a WE 320001 CPU with 64K bytes of ROM, 256K bytes of RAM expandable to 1 Megabyte. The terminal also has an RS232 interface.

The main power of the terminal derives from an interface to UNIX System V, which allows several windows to be placed on the screen. Each window has its own control software running in the terminal and also a user level protocol handler running in the host.

Q. Is the protocol specified and available?

A. Not yet

Q. Are there any cross development tools for the 32000 CPU?

A. Yes.

Q. What is the availability in Europe?

A. Olivetti will distribute it.

Q. Will the software run on any UNIX system other than System V?

---

‡ Thanks to Jaap Akkerhuis for finding them in a machine readable form and sending them to me.

A. No comment.

Item 2: 10.30am **P.Freund, Hewlett Packard**

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

Abstract

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.

This talk discussed HP's approach to the implementation of UNIX System III (with those familiar Berkeley enhancements - i.e. **vi**). The hardware is based on a single chip 32-bit CPU with a stack based architecture. A configuration can consist of several multiple CPU's. HP wanted to support operating systems other than UNIX, and have developed an internal kernel called SUN (no relation to those other folks, folks) onto which UNIX has been layered.

The SUN kernel has memory management, process management, a file system supporting multiple directory formats, device drivers, some I/O primitives, a real-time clock and interprocess message passing. It does not have a human interface because is it designed to support operating systems and not humans. Also, there is no means to load programs.

The talk then described the ins and outs of the implementation which I hope will be reproduced elsewhere. I was left with the feeling that the system worked, but perhaps someone out there in UNIX-land might like to give a slightly less biased report.

☞ Coffee (and a peek at the exhibition) ☜

Session 2 was chaired by Adrian Freed, Ircam, France. I missed the next person's name, sorry.

Item 3: 11.31am **Donald St..?, Amdahl Corp**

Future directions for UNIX at Amdahl

Abstract

This talk will cover future plans for UNIX on Amdahl.

The talk started by summarising the history of UNIX running on Amdahl machines. The system is called UTS and runs in a virtual machine under the VM operating system, the latest release is UTS 2.2 and UTS 2.3 will be released shortly. The current release has 'good V7 compatibility'. The idea of marketing UNIX is to take advantage of an internal product which had been developed for in-house engineering use, to increase their reputation as a software vendor rather than their being known as simply a hardware supplier, and to make inroads into the academic community. Future plans are to: continue leadership in the large system UNIX market and to maintain compatibility with the Bell Labs product at current release levels. I.e. they are working hard on System 5, which is running in-house and at some installations of early customers.

Q. Pricing?

A. An AT&T license is required, plus $1500 per month.

Q. European support?

A. Yes.

Item 4: 11.53am **Larry L. Crume, AT&T Bell Labs**

UNIX* System V Release 2.0 and Future Directions

Abstract

Enhancements to UNIX System V Release 1.0 will be reviewed. These enhancements which are to be released in April 1984 include feature updates and improvements in the following areas: C Compilation System; Job Control/Virtual Terminals; Shell; Commands; Cron Facility; Curses/Terminflo Package; Standard Disk and Tape Names; Accounting Package; Performance; and New Documentation.

Future Directions: Future enhancements to the UNIX System will focus on the user interface. The paging system under development at AT&T Bell Laboratories shows one dimension of the user interface for systems programmers. The architecture must be general enough to support both paging and swapping kernels and many different memory management units.

The work on command syntax and error message handling provides for a consistent user interface and to easily determine and recover for errors.

Work on unbundling and repackaging the UNIX System will provide for a consistent user view of the UNIX System.

To fill that in a little. This year's system from AT&T is called System V, release 2. There are several new applications programs which come with the system and some enhancements to the operating system.

The biggest development objective is to maintain upwards compatibility while improving performance. This does not necessarily mean that the increased speed for increased size trade-offs will be done, but AT&T are looking at some of the UCB enhancements.

Job control has been introduced. This has been done in a totally different way from the UCB implementation because it was not desirable to introduce the necessary new signals. AT&T are committed to support and not extend the current set of system calls. Job control is done by having a number of layers which define virtual terminals. On login, the user talks to a control layer and has the ability to start a number of shells in other layers. Input and output from these virtual terminals can be controlled independently.

The new C compiler will have long variable names. Also, there will be a C cross compiler for the M68000.

Larry then talked about the current ideas of unbundling UNIX. The pieces will consist of a basic set of utilities plus the kernel and yes, your favourite command is bound to be missing. There will then be several add-on pieces, such as the C language tools, the various workbenches, and other utility sets.

On the list of things to be looked at in future releases include: record and file locking, this will initially be the /usr/group standard; bad block handling; and file system integrity, such as protection from unexpected halts, ordered writes to discs, timely flushing of buffers and improved detection of corrupt file systems. A big thing on the list is a paging kernel in order to provide a large address space. Here, the main aim is to not affect users who don't wish to have paging in their machine. To this end, an architecture for memory management has been developed, with the idea that the paging should be easily configurable in or out. This is not on the current release, because it wasn't good enough.

This was an interesting talk, really, packed with a lot of stuff which I feel there is no room to put here. AT&T seem to be very responsible with the developments which they propose, it is perhaps possible that they are trying to generate too many new systems and are not leaving things to settle enough. My other main criticism is that most of the things coming out are fairly mundane and ordinary, the interesting leading edge of technology stuff is staying under wraps until it becomes safe and boring. I suspect that as an academic user, I would like to have Edition 8.

☞ Lunch (and a bottle of beer) ☜

This session was chaired by Bjorn Eriksen.

Item 5: 2.05pm **Bill Murphy, AT&T International**

UNIX Licensing

Abstract

What you can and cannot do for your $43,000, $68,000 etc.

Well, all Bill did was show his face and get off. The idea was that people could tackle him outside the main hall. Which I suppose they did.

Item 6: 2.08pm **Robert Ragan-Kelley, Pyramid Technology Corp.**

OSx: Towards a single UNIX system for super-minis

Abstract

OSx is a dual-port of 4.2BSD and System 5 onto the pyramid 90x computer, a high-end super mini. OSx is designed to be fully compatible with both 4.2 and System 5 in a fashion that neither suffers performance penalities from the coexistence of the other. This paper discusses some of the details of this design, both internal to the kernel and at the user interface level, along with some of the problems we faced in its implementation.

The idea is to implement 'universes', one called **btl** and the other **ucb**. These two names are used as commands to switch between the two universes. It is also possible to have a command line containing commands from one universe in the other, by prepending the alien command by the appropriate universe name.

The implementation started from 4.2BSD and emulates System V. The main reasons for starting from 4.2BSD are the demand paging, the fast file system, flexible file name lengths and the larger block size. Also, the 4.2BSD networking would be difficult to implement on a System V.

The main problems are: the differences in the directory structure, System V FIFO's (named pipes), signal handling, System V IPC and worst, the differences in terminal handling and **ioctl**'s.

Item 7: 2.40pm **Eric Allman, Britton-Lee, Inc.**

The special advantages and difficulties with databases in UNIX (1)

Abstract

Many applications maintain some long term state in the form of a database. In many cases ad hoc algorithms are sufficient (e.g., sequential scans of the password file are adequate on most systems), but often more sophisticated algorithms must be considered (e.g. mailboxes must be locked while mail is being delivered; the dictionary is too large to be practically searched sequentially).

Although ad hoc approaches are acceptable for small applications, larger applications often find it convenient to utilize a full-blown database system. Such systems may include such features as efficient access methods, logical independence from data structure, aggregation, protection, integrity constraints, multi-file capability, concurrency control, crash resilience, audit trails, and transaction control.

The structure of a database system incorporating most of these features is examined. Interfaces, data models, cost/performance tradeoffs, and the special advantages and difficulties UNIX offers to database systems are discussed.

This, the first of two talks, gave a general introduction to database terminology and operations. It seems to me that it would be silly to attempt to summarise his talk here, we'll wait for the paper which will no doubt be considerably more comprehensible than anything I can write.

☞ Coffee ☜

Item 8: 3.42pm **Eric Allman, Britton-Lee, Inc.**

The special advantages and difficulties with databases in UNIX (2)

The second part of the talk was concerned with the facilities which are provided by data base systems and the trade-offs inherent in such systems.

Item 9: 4.01pm **P.B. Pynsent, University of East Anglia**

The Norwich Renal Unit Programme

Abstract

In Europe 120 people per million of the population suffer from chronic renal disease and of those 80% depend on an artificial kidney machine for survival. We have developed a UNIX based computer system which not only provides access to a patient database but also controls kidney machines during the haemodialysis of patients.

The objective of the Norwich Renal Unit project is to improve patient care using computer technology. First we have provided facilities for computer controlled kidney machines to optimise dialysis therapy to the individual patient and secondly we have provided an easy to use patient database to aid the physician in his assessment of patients. The UNIX operating system has proven an ideal environment satisfying both the multitasking and data processing requirements of our project.

I really liked this talk. It is so rare at UNIX gatherings to see people who are doing something real with computers. I think that I mean that the majority of talks at EUUG conferences are really 'Computer Science' and I would like to see more applications oriented presentations.

Item 10: 4.31pm **Andrew Hume, AT&T Bell Labs Computer Research**

Eric: an experimental information manipulation system

Abstract

Eric is a testbed for a model of how the user interacts with a computer system. The major components are the filing system, multi-tasking, the use of forms as the only means of data input and a user interface dependent on a bit mapped graphics display.

The abstract does not do justice to the talk, but my notes are even worse because I spent most of the time concentrating on what was being said.

☞ End of Day 1 (and onto the Hotel Erica for free drinks) ☜

Day 2 - 17th April

Well, I managed to make it out of bed to chair the first session of the day.

Item 11: 9.39am **Kirk McKusick, University of California, Berkeley**

The Dynamic profiling system

Kirk's talk centred on the new profiler **gprof** which comes with 4.2BSD, and described how he had been using it to improve kernel performance. The original UNIX profiler presents a flat profile of program performance with the emphasis of the amount of time spent in a particular routine. **Gprof** does better than this by tracing the path through the code and giving statistics for: how often routines are called; from where; and in turn, which routines were called by the routine under consideration. (Having used it to analyse program performance, it's really good).

Kirk then described a scientific investigation into UNIX kernel performance. He found that **namei**, the main directory search routine in the kernel took one quarter of the system time in 4.2BSD. He described various attempts make this go faster, pointing out that some obvious solutions appeared to improve some aspects of system peformance (i.e. **ls** appeared to be better) while profiling showed that overall system performance had not really improved.

'Make it work and then make it go faster' is an old Bell Labs axiom, perhaps we should add: 'then show it really does go faster under all circumstances'.

Item 12: 10.05am **A. Burns, University of Bradford**

A Comparison of the UNIX and APSE Approaches to Software

Abstract

An important aspect of the Ada project is the attempt to design and implement a standard support environment for the development and maintenance of Ada programs. A number of Ada Programming Support Environment (APSE) projects exist; many are using UNIX as a basis for the work and as the starting point for design. There are however many important differences between the use of software tools under UNIX and that envisaged for an APSE. This paper is concerned with the use of software tools and their interfacing. Comparisons between a UNIX and APSE approach are given.

This was another, much needed, introductory talk.

Item 13: 10.36am **Bill Weir, STC IDEC Ltd**

C Unit Test Harness

Abstract

Module testing is an important and often neglected area of software testing. Traditionally, it has tended to be a largely undocumented operation in which the programmer pokes data interactively at a module until he believes it is working satisfactorily. Studies have shown that tests conducted in this manner are seldom adequate in their coverage of program paths. It is hoped that, by automating much of the process, and by relieving the programmer of the drudgery of creating driver modules, collecting results, etc., more extensive testing at a module level will be encouraged, with consequent reductions in testing and debugging costs at a later stage of the software cycle.

A talk full of interesting ideas. Bill presented a system where the testing of routines can be formalised and automated. I felt that I need something like this, if only to aid regression testing. The main problem was the specification of tests.

Q. Are there any tools to help in building tests?

A. No, not at present.

Q. What is the availability of this?

A. Sorry, don't know.

☞ Coffee ☜

The session chairman was Teus Hagen.

Item 14: 11.26am **M.A. Rathwell, University of Bradford**

Distributed Decision Making under UNIX

Abstract

Distributed Decision Making attempts to meet the need for supporting tasks which involve cooperation and conflict between differentiated organisational units. A DDM system provides a mechanism for linking several decision support systems in an organisation, so enabling groups which are not necessarily linked in a hierarchical manner to cooperate with one another. This is particularly significant in planning operations which require information from different people dispersed throughout an organisation. It can enable independent decision makers to semi-automate their work, explanations for decisions can be made widely available, and the resolution of conflict between nodes with different interests and perspectives can be supported.

Item 15: 11.50am **A.R. Pell, University of Reading**

An Interactive Information Retrieval System for UNIX

Abstract

Many problems exist in office and information systems for which an appropriate solution is an information retrieval system. The aim of the first part of this project has been to build an easy-to-use interactive system. This has involved analysing the concepts and information structures needed, as well as considering the user interface. The emphasis throughout has been to construct the system in such a way that it is easy for a novice user to handle. Building on the established system, the second part of the project will involve experimenting with differing input devices such as touch screens, mouse input, trackballs, voice, etc.

Item 16: 12.15am **Theo de Ridder, IHBO "de Maere**

Automatic Generation of Syntax Directed Screen Editors

Abstract

From a new effective and automatic error-recovery scheme for LALR(1)-parsers a program generator is developed that produces a syntax directed screen editor for any language specification written in LEX and YACC.

☞ Lunch ☜

Session chair was Jim McKie.

Item 17: 2.00pm **J. R. Nicol, University of Lancaster**

CRS - A Powerful Primitive For Resource Sharing in UNIX

Abstract

This abstract focuses on a resource sharing system which we call 'CRS' (Connect Remote Shell). CRS is layered on top of the UNIX operating system and provides a powerful set of network services. The environment in which CRS was developed formerly consisted of a number of PDP-11/44 mini-computers, each running UNIX. A user's computing activities were typically centered on one of these machines. Circumstances changed when we obtained a Cambridge Ring local area network, since we were then presented with the possibility of sharing the available computing resources.

Item 18: 2.37pm **Brian E. Redman, Bell Communications Research**

Honey Danber - The UUCP of the Future

Abstract

In 1978, Mike Lesk was considering a mechanism to aid in the administration of software on the growing number of computers running UNIX at Bell Labs. He envisioned a system that would automatically synchronise several machines with updates from a single source. At the time, no networking software existed upon which to build such a system, so he invented the UNIX to UNIX Copy program (**uucp**) to transfer files among machines. Little did he know that **uucp** would become the foremost file transfer and remote execution facility for untold years to come. That which was created as a temporary measure to get data from one UNIX system to another has endured through time as one of the most beleaguered, yet most critically required UNIX utilities.

Brian's talk centred on a recent re-write of the **uucp** suite of programs which will be available on System 5, release 2. The re-write was undetaken by an ad-hoc committee and the programs are a product of 'software engineering' and not just hacked together. The result is a much more flexible, faster and better controlled **uucp**.

☞ Coffee ☜

Session chair was Keld Simonson.

Item 19: 3.45pm **P Tintel, Bell Telephone Manufacturing Company**

EURONIX: A UNIX based system using European natural languages

Abstract

The UNIX system has gained enormous popularity. It is used in many places for software development, and it is beginning to be used in office environments. However, the average office worker is no computer specialist and he has other demands concerning the system then software developers.

The UNIX user interface as it is today has certain drawbacks for office applications and more specific for office applications in Europe. The manuals are not very readable for someone not familiar with UNIX; all communication with the user is performed in English (or American); and UNIX is not capable of working with the different European characters in a uniform way.

In the Euronix project, focus is on the second and third problems: EURONIX will communicate with the user in the user's natural language and will be able to handle in a uniform way the special European characters.

The implementation is to alter UNIX from working in ASCII to working in the TELETEX character set which can cope with all the European 'funny letters'. In addition, all messages from programs pass through a string data base in the user's natural language.

Item 20: 4.16pm **C. Roberts, Inf. Techn Task Force, EEC**

Standardisation of the national character sets

Abstract

A review of the European Standard policy for national character sets will be given. The way it fits in the current EEC programs will be discussed: the general options of setting standards in character sets, some examples of standards of character sets, as well as the implementation policy (some keyboard experiments).

Item 21: 4.45pm **Emrys Jones, Chairman EUUG**

EUUG Annual General meeting

Abstract

This is the official general meeting to present the new constitution.

This meeting was held as a bootstrap device to get the new constitution off the ground. The preliminary constitution was passed and the new structure formally inaugurated.

☞ And so to Dinner at the Erica ☜

The idea of having a conference dinner was a good one. However, the red wine was awful. There were some after dinner speeches to make it taste better. Theo de Ridder made a speech which I have acquired in machine readable form. I am indebted to Jaap Akkerhuis for twisting his arm, leg or some other piece of anatomy for it.

Item 22: 9.45pm? **Theo de Ridder**

After dinner with Theo

Dear delegates.

In the past scientists used a very sensible language to express and exchange their ideas. The importance of that old dead Latin was that you had to be conscious of its fixed syntax and semantics in order to use it. At this moment I am permitted to make a speech in English, without much knowledge of its syntactic or phonetic structure. I dare to do so because in a more interactive situation no-one ever interrupts me with error messages whatever my mistakes.

Looking around in an UNIX environment the simularity between a programming language and a natural one is remarkable. There is not any formal syntactic or semantic description available and still programs are made and ported all over the world. In case of a programming error the system is kind enough not to complain about its exact type or place, it just gives you a wink that something went wrong. Isn't it a nice paradox that such an honourable human reaction is considered as typically user-unfriendly?

Let me go on with an anthropomorphic view on UNIX. I am aware that most of you do have a rather intimate relation with it. Using the statistical argument that for 90% you are male and will prefer the opposite sex, I conclude UNIX is female! Well gentlemen, what about your behaviour over the last decade? In any case you exploited her. Sometimes you even raped, suppressed or sold her. And some individuals had the courage to publish the insultant proposal to bring her in the public domain! In spite of certain parallel liberation movements in society UNIX was not able to free herself from historical bounds into an independent respectable creature.

There is a more philosophic and fundamental aspect of the human condition than being male or female, and that is being mortal. So, UNIX is not eternal. She must be in one of the binary states, alive or dead, or else she is instable in illness. It is hard to prove where she is. Her growth and continuing changes indicate liveliness. The exponential increase however reveals a disease like cancer. And finally the cult of these meetings, the existence of gurus, and the need for myths synthesize the declaration of a posthumous holiness.

After dinner it is fun to tell each other fairy tales. Maybe you need a tutorial example. I hope it will be simple to apply the following to your own business.

Once upon a time there was a princess called V6. She was considered small and beautiful by her people. Many a handsome academic prince came along to ask for her hand. But her mother, Ma Bell, locked her up to prevent disintegration of the empire. And so she became old and ugly in grim electronic towers. Her only pleasure was looking out of a window into the silicon valley and watching the play of the big cat AT&T with a lot of little licenced mice like ... you.

Day 3 - 18th April

Well, unlike yesterday, prolonged late night discussion kept me in bed to miss the first session. The session chairman was Joachim Wolff.

Item 23: 9.30am **Joe Carfagno, Central Services Organisation**

Using UNIX on a large software project

Abstract

This talk is about the uses of the UNIX operating system in a large software project. Many different implementations and releases of the UNIX system, including the UNIX (1100) system, are used in a variety of ways from software development, testing, project management, site support, and others. This talk will show the versality, flexibility, and portability of the UNIX system.

Item 24: 10.05am **Ludo Vemmekens, Amdahl Nederland B.V.**

Large Systems UNIX: Opportunity for Innovation

Abstract

In bringing UNIX to the large mainframe world, there is a merging of two standards: the UNIX standard of openness, ease of use, ease of development, and the IBM large operating system standard of high reliability, security, performance, and support. The combined standards are a new challenge to the operating system products world.

This paper will discuss in detail the technical issues in making UNIX a viable mainframe operating system. These include reliability, production operations management, communications, memory management, full duplex support, database, security, support, compilers, applications, coexistence with MVS, VM, and UNIX.

☞ Coffee ☜

I got to the hall just in time to see Andrew Hume, session chairman and contestant in the 'hairy knees of the conference competition' make a small opening speech deploring the current fashion of being rude about UCB. More power to his knees.

Item 25: 11.18am **David Tilbrook, Imperial Software Technology**

The 5 Pitfalls of Interactive Graphics

Abstract

The NEWSWHOLE system was created almost 10 years ago. A video tape of it has been widely distributed and used to teach interactive graphics. One way it has been used is to examine the way in which it avoids the 5 pitfalls of interactive graphics: Boredom, Confusion, Discomfort, Frustration and Panic. This presentation will discuss those pitfalls and the design strategy used to avoid them.

The abstract does not point out the thing of which David is most proud. The system worked on a high resolution display and a different cursor shape was used to indicate different operations. The one that springs to mind is the Buddha which meant 'please be patient, I am computing'. This idea is a goody, it's one of those things which when you see you wonder why everyone doesn't use it, but of course, you have to have the idea first.

Item 26: 11.50am **Brian E. Redman, Central Services Organisation**

Behind every Binary License is the UNIX Heritage

Abstract

Lately there seems to be some pessimism about the future of the UNIX system. Many who have watched its development from the earliest days feel that the system appears to grow corrupt and is no longer a model of innovation in operating system design.

Unix was originally designed by a talented fraternity with a clear and common vision for a better environment. Ever since the system has been redesigned by a diversity of people with different goals the tend to be less clear. UNIX has evolved from a simple, elegant model into one that is certainly complex and often seems convoluted. It no longer constitutes a statement of smallness, but appears to be growing unrestricted. ...

This was certainly the funniest talk of the conference, we hope to get a reprint. Contrary to popular opinion, Brian was never in the running for the hairiest knees of the conference award.

☞ Lunch ☜

The afternoon session was chaired by Mike Banahan

Item 27: 2.02pm **Andrew Hume, AT&T Bell Labs Computer Research**

Processes considered as files

Abstract

Tom Killian has implemented images of running processes as full and legitimate elements in the file system. The image for process nnnnn is accessed as '/proc/nnnnn'. Read(2) and write(2) work normally except that some system data is write protected. Ioctl's include stopping and starting a process, masking out signals that cause a stop, and returning a file descriptor for the text file (for the symbol table).

This is a neat idea costing 4K bytes in the kernel.

Item 28: 2.08pm **Daniel Karrenberg, University of Dortmund**

University of Dortmund's Spooling system

The University of Dortmund have several machines running several versions of UNIX but have few printers. They also have no Local Area Network, such as an ethernet. The talk described the spooling system which has been set up to cope with these problems.

Item 29: 2.15pm **Andy Greener, Imperial Software Technology**

EUUG Benchmarks: some results

Abstract

The EUUG Bench mark tests have been run on a variety of VAXs running UNIX-5, BSD4.1, BSD4.2. Results are presented and discussed.

The tests were on VAXes and should be shown elsewhere in the newsletter.

Item 30: 2.29pm **Johan P. Moelaert, Twente University of Technology**

A Semaphore Implementation in UNIX

Abstract

For certain purposes the UNIX system lacks strength in its possibilities of interprocess synchronisation. Several processes waiting for one event require an abundancy of process control when implemented with signals, and this is not very reliable. Mutual exclusivity may be implemented using 'open' and - if this fails - 'create', but this is

neither very elegant nor completely foolproof. The pipe mechanism offers a good instrument for synchronisation, but can only be used between processes that have hierarchical relationships. For these reasons we decided to implement Dijkstra's semaphores in the UNIX system.

This was an ingenious solution which added some system calls to do the semaphore user interface. The system uses the in-memory inode table to store state of the semaphore. The talk was the only one to actually put some C up on the screen, and for that reason alone, scored some points.

Item 31: 2.42pm **Neil Mayhew, Bleasdale Computer Systems**

Experiences With Implementing IBM Bisync IN UNIX

Abstract

Bleasdale UNIX micros are used in a variety of situations commercial, scientific and academic, and there is now increasing demand for distributed processing in the form of connecting micros to existing mainframe and database facilities.

Bleasdale's first step towards meeting this demand was to promote file transfer facilities, including RJE (Remote Job Entry), using IBM 3780 Bisync.

Item 32: 2.55pm **Theo de Ridder, IHBO "de Maere**

MABENCH: a portable benchmark machine

Abstract

In the computer science education (HIO) department of our institution we developed a complete synthetic benchmark package BENCH. In BENCH it is possible to specify arbitrary workloads with any number of parallel processes. Scriptfiles can be made by editing or by running application oriented scriptfile generators. In the output all the characteristic performance values (throughput, response time, service time) are given in table format.

The MABENCH system is a small portable machine which can be connected to the system under test via normal terminal connections. The small processor (6809) then sends several scripts to the test machine while performing measurements.

Item 33: 3.10pm **Nick Nei, University of Glasgow**

Measuring the Disk I/O on a VAX

Abstract

This paper describes a project under way at Glasgow University to gather statistics about disk performance on a VAX running Berkeley UNIX 4.1. These results will be used to construct a stochastic model for the behaviour of the disk subsystem. We hope that by modifying the parameters on the model and studying the results we can discover new ways of improving the disk and file system performance.

It seems that the measurements show no really discernable statistical model which is applicable to disc performance.

☞ Coffee ☜

Item 34: 3.30pm **Panel Session chaired by David Tilbrook, IST**

Is there a future for UNIX?

During this session, I had to go to catch a plane. Richard Hellier from the University of Kent kindly agreed to take notes and produce a report. So here it is:

The panel was:

L.L.Crume, AT&T
R.Raglen-Kelly, Pyramid
K.McKusick, UCB
H.J.Thomassen, U of Nijmegen CS Dept.
M.Banahan, The Instruction Set Ltd.
A.Freed, IRCAM.

Each of the panel members made a short statement on the subject:

Does UNIX have a future, and, if so, which way is it going?

Larry Crume expects further developments in networking tools and support and much greater use of windowing software at every level; Many novice interfaces would be required in future. He also mentioned the unbundling of UNIX and the simplification of the licencing process.

Roger Raglen-Kelly was concerned about people hacking UNIX for the sake of UNIX itself. He thought that UNIX was already too big and that something should be done to arrest the growth. What he wanted to see: was better internal support for multiprocessing and true networking; functional and object-oriented programming languages & shells.

Kirk McKusick reminded everyone that the days of formal releases of code from Berkeley were over and that they were returning to being a research institute.

Adrian Freed stressed the importance of separating UNIX from UNIX-based applications; With the preponderance of commercial and industrial users of UNIX, he felt that only a minority of UNIX users cared what the underlying system was. All they are interested in is their Spreadsheet, Data-Base or whatever.

H.J.Thomassen followed up this line, later amplified by Teus Hagen, and pointed out that the Academic Community was moving in a different direction from the business community.

Questions and comment were taken from the audience.

Emrys Jones introduced a point of information, namely that Computer Magazines & Journals, as a group, were now outselling "girlie" magazines. He asked,

Did the panel feel that the "hobbyist" section of the user community would ever have much impact on the UNIX community as a whole?

Due to the fragmentation of the enthusiast sector, no-one anticipated any significant developments from this area.

Teus Hagen pointed out that although business users may be moving one way, any individual user will still be free to follow his own path.

Another speaker emphasised that UNIX should be viewed as a way of doing things rather than as a static entity.

Following a prediction of researchers moving away from UNIX towards, say, expert and knowledge based systems, Eric Allman noted that all this new code would have to be developed somewhere, and probably on UNIX systems.

Kirk McKusick spoke of possible further developments of the UNIX kernel to support concurrent running for tightly-coupled multiprocessor systems.

Replying to a question on the conformity of AT&T products to /usr/group standards, Larry Crume said that AT&T were part of /usr/group and so their code would be 100% compatible with those recommendations.

Several speakers expressed concern that the UNIX tradition of distributing source code, even of the kernel, would not be upheld by the commercial users and software-houses. Larry Crume affirmed that AT&T would continue to supply source code with System V and its successors. Many attendees wanted some form of pressurisation on business users to distribute sources of their products. Before taking up the legal aspects of this topic, Mike Banahan observed that few business users care about source code; All they want is a tool for a job, he felt.

The next question was:

How can software developers protect their investment unless they restrict source?

Eric Allman, speaking for Britton-Lee, pointed out that small firms simply could not afford even one lawsuit to test a copyright case. He quoted 'one week to liquidation' if his firm ever became involved in such an action. Only the giants, like AT&T, could afford such a case. Britton-Lee will sell the source, but at a high price.

There was then a brief discussion of the impact of APSE technology on the UNIX community. Several speakers feel that even if the project fails to achieve its goals, like Multics, it will contribute much to the industry.

In response to the alleged unchecked growth of the UNIX kernel, Kirk McKusick observed that size must be traded off against functionality, i.e. that a kernel with a given set of services must be of at least a certain size.

The final topic was the future of Inter-Process Communication, instigated by Dave Tilbrook. Larry Crume felt that, in the current absence of any formalism for describing IPC we still don't know which way to go. None of the panel members would be drawn on this one.

Eric Allman observed that IPC enables software developers to write code that runs in user space and then move it into the kernel if memory allows.

There was a general view that there must be a logical separation between particular applications and the IPC mechanisms they employ. IPC enables us to *communicate* rather than *convert* software. That is, when some new service becomes available we converse with that rather than its predecessor, and to keep the applications small.

This spawned a nostalgic side-discussion about the good-old-days of Version 6; Eric Allman observed that much of the "elegance" of that system arose from its implementation on a particular machine architecture, i.e. PDP 11, and that many of the techniques employed were simply the only ways to proceed on such a machine.

As a postscript, there were a few questions about the purpose, or lack of it, of having future EUUG meetings.

Tutorial sessions

A number of tutorial sessions were run is parallel to the main meeting. A summary of each session is given here. The summary is derived from the meeting programme as I cannot be in two places at once (sometimes, I can't even be in one place at once).

Day 1 - April 16th

Item 1: 2.00pm **Mike Banahan, The Instruction Set**

Advanced editing: ed, ex, vi, etc.

The common aspects of line-oriented, screen-oriented and more advanced editors will be given. How to write your own editing macros, how to handle key stroke definitions and how do you program in your favourite editor's very own command language.

Item 2: 3.45pm **Bill Murphy, AT&T International**

Licensing, workbenches and UNIX System V 2.0

An overview of all the UNIX packages from AT&T Int. are given. Prices, availability and future developments to be expected. Most of the topics about licensing will be addressed.

Day 2 - April 17th

Item 3: 9.30am **Mike Banahan, The Instruction Set**

Advanced Shell programming

Just using one command with some arguments is easy. How you can make use of all the features of the UNIX command interpreter is some more difficult. It is addressed how you can make your command life easy - no flags, just combinations of existing commands. Yes, you can program in Shell.

Item 4: 11.15am **Jim McKie, Centrum voor Wiskunde en Informatica**

UNIX 4.2 BSD

What 4.2BSD gives you, doesn't give you, gives you too little of, and, of course, what it gives you too much of.

Day 3 - April 18th

Item 5: 9.30am **Nigel Martin, University College London**

The UNIX market

An overview of the market situation concerning machines with UNIX will be given. How do they differ and what kind of choice should be taken? The direction of this market. Slides of the rivals will be shown.

Item 6: 11.15am **Teus Hagen, Centrum voor Wiskunde en Informatica**

UNIX networking with UUCP

How do you connect your machine to the outside world. Maintenance, structure, software and connection costs for an UUCP link. Statistics, software layers and tools are also addressed.

Item 7: 2.00pm **Jaap Akkerhuis, Centrum voor Wiskunde en Informatica**

Advanced Text processing

Text processing is not like word processing. To layout your text nicely, you should know more about the possibilities of n/troff, eqn, tbl.

Endpiece

Well, that's it for another EUUG conference. The papers which constute the proceedings will be published separately.

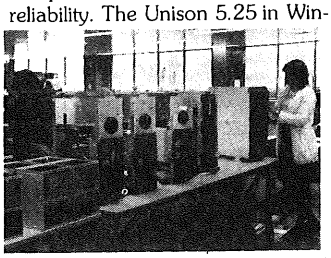
Thanks are due many people who worked very hard to make the event a success. Hendrik-Jan Thomassen and George Rolf headed up a team of tireless workers. David Tilbrook did the programme organisation aided and abetted by Teus Hagen. The EUUG secretariat, Helen and Debbie, did their usual good job.

And the winner of the 'hairy knees of the conference' award? Well, it was a close run thing. But since there were only two competitors, and both were Australian, the winner must be Richard Grevis.

Clippings this issue are from "What's New In Computing", July and August 1984.

ENHANCED SUPERMICROS

Email Computer Systems has announced a range of enhancements to its range of Unison supermicro computers to increase speed and reliability. The Unison 5.25 in Win-

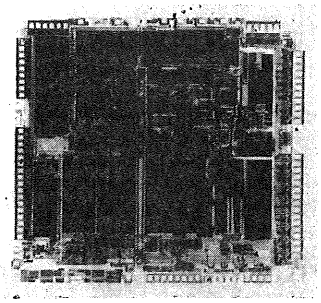


chester disc system with the ST506 controller can control up to two drives ranging in capacity from 10 to 160 Mbytes. The controller, which includes the Motorola 68450 DMA controller and Western Digital's 1010 VLSI device, enables disc access at up to five times the speed of the IMI Winchester drives currently used in the Unison range. The SMD-IMI translator board provides current Unison users with an upgrade path to obtain higher performance from an IMI drive and retain their current Winchester discs. New magnetic tape controller boards have a resident M68000 processor to buffer the I/O logic and will form the basis of the PICOBUS interface to drive 9-track streaming tape units. Another product soon to be released is a distributed processor approach to an 8-channel VDU controller which has its own M68000 processor on board and will speed up I/O transactions. Email has two new fast memory boards; a 512 Kbit unmapped, zero-wait state system memory board designed to run the UNIX kernel; and a 128 Kbit fast managed RAM board developed to eliminate all memory wait states. Email has also been developing and benchmarking 10 and 12 MHz versions of the CPU board for the Unison product range (the Unison currently uses a 8 MHz CPU). These enchancements will enable the Unison range to support up to 16 users on the UNIX operating system. Unisam, a powerful file handler, supplements UNIX by providing a multi-user record management system that can be interfaced to a variety of languages and has a hardware-independent VDU I/O system. Email is also developing an interface which will enable Unison computers to communicate via Ethernet, providing a complete office automation environment.

Email Computer Systems.
Cnr Canterbury & Liverpool Roads,
Kilsyth 3137

FULL 32-BIT MICROPROCESSOR

The Motorola MC68020 microprocessor is presented as the first true 32-bit design in that it includes non-multiplexed 32-bit internal/external data and address paths — all 32-bit registers, all 32-bit arithmetic and logic units, all 32-bit programme counters and all 32-bit stack pointers. The MC68020 is manufactured using a 2-micron HCMOS process that integrates some 200,000 individual transistors on a 375 x 350-mil die which is packaged in a space-efficient, 114-lead, pin grid array package. The MC68020 operates at a 16.67 MHz clock frequency (60 ns clock



period), dissipates less than 1.5 W, and is completely upward user-object code compatible with earlier members of the M68000 family which has always offered a 32-bit architecture and now provides the full power of 32 bits with the MC68020. In addition, the device includes many new features which fully exploit the complete 32-bit design. The full 32-bit design structure of the MC68020 provides for direct linear access to 4 gigabytes of logical memory and eliminates instruction timing differences for byte, word and long word operations. The device processes instructions at a sustained rate of 2-3 million instructions per second (MIPS) and at burst rates exceeding 8 MIPS. The MC68020 fully supports virtual memory and virtual machine concepts. Virtual memory gives each programme access to the 4-gigabyte logical address space while allowing the system to contain significantly less physical memory. The MC68020 provides a means to extend its architecture to off-chip devices through its co-processor interface. This interface allows devices such as the MC68881 floating point co-processor to execute instructions as a part of the main instruction stream. The 256-byte instruction cache allows simultaneous data and instruction accesses and execution as well as overall improvement of the performance of the system. The MC68020 contains a bus interface that provides, on a cycle-by-cycle basis, the ability to adjust its data bus width to accommodate 8-or 16-or 32-bit devices. The programmer is no longer required to develop programmes that are data bus dependent, as the MC68020 will dynamically make the necessary adjustments. Additionally, existing 8 and 16-bit peripheral subsystems can be used with the 32-bit MC68020 by taking advantage of this feature. The MC68020 also offers extra addressing modes and instructions to further support high level language development. In addition, 32-bit extensions to existing instructions are provided. The additional addressing modes include full 32-bit displacements, true memory indirection and scaled indexing. The new instructions include an entire family of bit-field operators, double-ended bounds checking, BCD data compression and expansion, module support and enhanced system calling functions. To efficiently handle task switching, as well as to separate task-related exceptions from system-related exceptions, two system stack pointers are available to the supervisor programmes. The master stack-pointer is associated with the user task so that all task-related exceptions are carried within the user's process control block. When a non-task related exception occurs, the interrupt stack is used for the remainder of system level processing until a new user task is initiated. Motorola offers cross support under its Unix operating system-derived system V/68 operating system and its real time Versados operating system. These operating systems are available on either the Exormacs multi-user development system or the VME/10 microcomputer system. A high-performance operating environment is offered in the form of the Benchmark 20 system, which provides the capability of evaluating the MC68020. This system contains a high-performance CPU board as well as one megabyte of dynamic RAM

Motorola Semiconductor Products.
250 Pacific Highway,
North Sydney 2060

UNIX-BASED SOFTWARE FAMILY

The Ventures Division of Cincom has released the CX/Xtend family of software, designed specifically for UNIX-based operating environments. The fully integrated family of products includes both application development aids and end-user productivity tools. CS/DBX is a resource efficient data base system that is fully compatible with Cincom's Total data base management system, accommodates network and hierarchical data structures, and, through a sequential view processor, supports complete inversion capabilities. Comprehensive logging, recovery, and restart features are built-in. Any number of users may access the data base concurrently. A powerful application development aid, CS/TMX offers high level facilities for generating, coding, and maintaining CRT screen displays independently of user-written programmes. CS/TMX also includes a comprehensive security system that can be used to restrict a user's access to only authorised programmes, at authorised terminals. Facilities are also provided to monitor a control network activity. Applications developed for use with CS/TMX are completely independent of terminal devices. CS/RMX provides relational access to the data base for inquiries and report writing. Also implemented on mainframes and minicomputers, the system allows end-users to quickly retrieve selected information by using English-like commands. Retrieved data can be displayed on a terminal, directed to a printed report, or both. Users can specify multiple selection criteria, perform arithmetic operations on the data as it is retrieved, and control the sequence of the output. Output formats to either the screen or printed reports may be generated automatically, or designed by the user. CS/Xpress is an interactive, interpretive system designed to make it easier to create complete application systems. A user can interactively design a complete screen format and describe each field's data edit characteristics. The user can enter and edit data and store it in the data base. Data fields that will be used as keys to the records on file can be specified. CS/Xpress files are created dynamically, together with a directory describing the characteristics of each new file. Users can display records and/or update or delete records that are already on the data base. All operations may be performed directly without any programming. CX/Xport provides control software to pass information from one computer to another. A programme running on one computer (which could be a PC, Unix-based system, or another mini or mainframe) can directly access data maintained in a CS/DBX or Total data base on another computer. Users can also pass data from machine to machine, a message at a time. CS/Xport can also be used to move text files from one computer to another, maintaining the files in the host machine's format. This product makes it easier to implement a network of similar or differing machines that share access to a distributed data base.

Cincom Systems of Australia Pty Ltd.
220 Pacific Highway,
Crows Nest 2065

UNIX SUPERMICRO

The Morrow Tricept Supermicro supports four to eight users running the UNIX System V operating system on the 16/32-bit MC68000 microprocessor, and has optional slave processor boards based on the 80188 CPU and running the MS-DOS operating system. Hardware includes an MC68000 CPU running at up to 10 MHz, with an on-board MC68451 memory-management unit; 512 Kbytes of main memory (expandable to 2 Mbytes); an I/O controller with four RS232C serial ports (expandable to eight ports); a Centronics compatible parallel printer port; hard and floppy disc DMA controllers; and 80188-based slave processors with 128 or 512 Kbytes of on-board dual-port RAM. All boards plug into the 14-slot IEEE-696 (S-100) bus backplane. Mass storage includes one to four 16 or 34 Mbyte 5¼ in Winchester disc drives (up to 136 Mbytes storage); an optional 5 Mbyte removable hard-disc cartridge drive; one to four 400 Kbyte 5¼ in floppy-disc drives; and one to four optional 1.3 Mbyte 8 in floppies. Unisoft has added such enhancements as record-locking and IEEE floating-point capability. An optimising C compiler comes with the system; other available languages include BASIC, COBOL, FORTRAN 77, Ada and Pascal. Tricept's serial I/O controller communicates with terminals and printers at rates of up to 19.2 Kbaud. The hard disc controller features a seek time of 85 ms with the standard 16 Mbyte drive, or 35 - 45 ms with the optional 34 Mbyte drive; data transfer rates are up to 625 KB/s. Planned enhancements include boards providing Ethernet support and PC graphics capability.

Automation Statham Pty Ltd.
47 Birch Street,
Bankstown 2200

DESKTOP ENGINEERING WORKSTATIONS

The HP 9000 Series 200 line of Motorola chip-based computers has been expanded with the model 217 and the model 237. BASIC and Pascal operating systems, as well as the HP-UX operating system (derived from UNIX), provide a growth path across the entire line. The model 217 engineering workstation is a modular computer using the MC 68010 processor with memory management and 8 MHz clock. It has a 14 in green-phosphor monitor with 512 x 390 resolution, and 512 Kbytes of RAM, expandable to 4 Mbytes using a new 1 Mbyte RAM board. A mouse is available as an option. The model 237 offers a 17 in bit-mapped display and shares the same operating system, peripherals, interfaces and accessories, as other HP Series 200 models. The model 237 graphics workstation uses the MC 68000 processor at 12.5 MHz with memory management and cache memory. It has a 17 in, no-flicker monitor with a 1024 x 768 bit-mapped display and 512 Kbytes of RAM, expandable to 7 Mbytes using the 1 Mbyte RAM board. The mouse is standard on this workstation. Other additional options include: a floating-point maths processor that provides up to three-fold improvement in performance, which is especially useful when using the seven-channel, 55,000-samples/s A/D card in data-acquisition applications; and Rev. 3.0 BASIC and Pascal operating environments which offer up to 50% performance improvement over previous versions, and also support the 150 cps ThinkJet printer and the HP 9122D double-sided 3½ in dual microfloppy disc drive. Extra HP software for the Series 200 line includes two new terminal emulators for the DEC VT100 and the TEK 4010, and HP TechWriter, which permits merging of graphics and text in the same document.

Hewlett Packard Australia Ltd.
31-41 Joseph Street,
Blackburn 3130

C COMPILER

C86 is a complete C compiler for the IBM PC and machines running CP/M-86 or MS-DOS which offers: the production of tight code; identifier names up to 31 characters long for better readability and maintainability; a library source with routines for Unix I/O, redirection, 8087 support, sorting, math and trigonometry; optional unsigned char and unsigned long data types for compatibility with popular 8088 C compilers; programme overlay support, allowing large programmes to run on small machines; the support of a nested comment option; command line name definitions for use of the conditional compilation features; and optional production of a listing of the source programme after macro substitution to simplify the debugging of macros and conditional compilation commands.

Software Source Pty Ltd.
344-348 Oxford Street,
Bondi Junction 2022

16-BIT MICROSYSTEM

The Dual 83/80 multi-user, multi-tasking system provides support for four users (expandable to 16), an MC68000 based central processor with memory management (10 MHz clock rate), 512 Kbytes of dynamic RAM (expandable up to 6.25 Mbytes), a four port serial serial I/O board with DMA, a 32/64 Kbyte EPROM board containing system boot, a 20-slot IEEE 696/ S-100 backplane, a nonvolatile clock/calendar, an SMD disc controller, an 80 Mbyte 8 in Winchester drive with 20 - 25 ms average access time, a floppy disc controller, a 1.25 Mbyte floppy disc drive, a multi-user UNIX operating system licence, a UNIX (System V) operating system, a C language/ compiler, and rack mountable cabinets. Options include additional disc storage (to 160 Mbytes with Dual drive), additional serial I/O ports, D/A and A/D converter boards, a nine track tape drive, a nine track interface, a floating point processor, a CAT 1600 Graphics board, a variety of operating software/languages (including RTK, UNIX System III, Forth, Fortram-77, Pascal, RM/COBOL, BASIC and Lisp), and applications software (LEX, MBSI, Viewcomp, MicroINGRES, Unify and Precision Visuals). System 83/80 operates at temperatures up to 60°C and comes with a 12 months warranty.

Dual Systems Australia
55 Phillip Street,
Parramatta 2150

UNIX BUSINESS APPLICATIONS GENERATOR

Today, claimed to be the first business applications generator for UNIX, was developed on a Wicat 200, by a team from bbj Computer Services and Wicat. Today interfaces to the UNIX file system utility C-ISAM and is comprised of two parts; the system administrator module (for defining terminals to the system, user passwords etc.) and the developer module, which creates the application. There will be a dual approach, firstly to the user who wants to utilise a broad range of commercially accepted applications developed using Today and running under a tun-time only version, and secondly to users developing and running their own applications using a full development version. Users will receive comprehensive support from Wicat, including training, assistance to new users and problem solving.

Wicat Computer of Australia Pty Ltd.
88 Christie Street,
St Leonards 2065

DATA PROCESSING PRODUCTS

The AT&T range of data processing products includes a personal computer, an interface between personal computers and the 3B line, and a local area and campus network. Also available to end-users of its 3B2 and 3B5 computers, are 3B network and protocol converters to allow asynchronous terminals to communicate with synchronous systems. The AT&T Personal Computer runs the same MS-DOS operating system and business software as other leading personal computers and accepts the same plug-in accessory circuit boards, however, it is said to have greater speed, more features, and a higher level of standard equipment than its leading competition. The personal computer can operate in an integrated computing environment with AT&T UNIX — based 3B computers through the AT&T PC Interface, which allows the computer to act as one of up to 16 workstations in a network with the more powerful 32-bit 3B super minicomputers. The Information Systems Network (ISN) is a local area and campus network that combines fibre optics and existing, standard copper wire to link workstations, terminals, PCs and minicomputers and communications processors into one system. ISN also complements AT&T communications systems such as System 75 and 85 and the Dimension PBX family. AT&T also have a comprehensive series of software products for both the 3B and Personal Computer product lines, including word processing and spreadsheet programmes, and industry-specific software such as the AT&T Gift Registry for retailers.

AT & T International (Aust) Ltd.
60 Margaret Street,
Sydney 2000

MULTI-USER SUPERMICRO

The ICL CLAN supermicro computer system supports both the PICK and UNIX operating systems, is based on the Motorola 68000 microprocessor and will initially support up to 16 users concurrently. A basic ICL CLAN system consists of a 40 Mbyte Winchester disc, 0.5 Mbytes of memory, a 20 Mbyte cartridge tape and a basic I/O board which can support up to six users via four asynchronous and two synchronous RS232 ports. All facilities are contained in a single compact free standing cabinet. Current enhancements include the addition of 0.5 Mbyte store modules up to a maximum of 2.0 Mbytes for PICK machines and 3.0 Mbytes for UNIX systems. A further three Winchester discs may also be added. providing a total capacity of 160 Mbytes while a second I/O board provides ten additional asynchronous RS232 ports for local use. A wide variety of applications software is readily available under both PICK and UNIX.

ICL Australia Pty Ltd.
14 Rodborough Road,
Frenchs Forest 2086

UNIX TRAINING SERIES

Three series of training tapes introducing the UNIX system, and examining its operation and applications, have been produced by Telemedia. UNIX Overview is a six part introductory course which is suitable for use either on a self-learning basis or in small groups and takes 6 - 9 h to complete. UNIX Fundamentals consists of 15 separate courses, includes hands-on experience and covers such areas as path names, directory commands, file access permissions, file name generation and the 30 most commonly-used UNIX commands. The series takes up to 30 h to complete and is designed to be used on a self-instructional format or in small groups. The UNIX Fundamentals series provides the basic skills needed to proceed with further UNIX training, such as C language programming. The courses provide a complete introduction to all aspects of the language and are suitable for both UNIX and non-UNIX C applications. Among the 16 course headings are: introduction to C language; creating a C programme; control statements; conversions; pointers and addresses; functions; storage classes; and structures and unions. The series takes a minimum of 24 h to complete. All three series can be purchased, rented as part of a long term contract or rented on a short term, one-off basis.

Deltak Pty Ltd.
53 Walker Street,
North Sydney 2060

Netnews

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

From jr@forosl.UUCP Fri Sep 28 13:55:10 1984
Date: Fri, 28-Sep-84 13:55:10 AEST
Newsgroups: net.lang.c,net.unix-wizards
Subject: Master listing of predefined CPP symbols

Hi. As promised (a *long* time ago), here's my updated list of predefined CPP symbols, which are generally used to allow machine and/or operating system specific code to be #ifdef'ed. Very few of these are actually documented anywhere; the purpose of this list is to try to take some of the "folklore" and make the information available to everyone. The data here is collected from a number of sources; please see the "acknowledgements" list at the end of this article. I don't have any data on anything earlier than V7.

THE LIST:

name	description	availability
__DATE__	date of compilation	Decus C
__FILE__	current source file name	most CPP's
__LINE__	line number within current source file	most CPP's
__PAGE__	page number within source	Data General C's
AOSVS	Data General AOS/VS operating system	Data General C's
aosvs	Data General AOS/VS operating system	Data General C's
cpm	CP/M operating system	"p" in Dr. Dobbs Jul-84
DATAGENERAL	Data General hardware	Data General C's
datageneral	Data General hardware	Data General C's
decus	Decus C (PDP-11 - RSX & RT-11)	Decus C
DGUX	Data General UNIX (DG/UX)	DG/UX
dgux	Data General UNIX (DG/UX)	DG/UX
gcos	Honeywell 6000 GCOS op.sys.	AT&T C compilers?
ibm	IBM (and Amdahl?) mainframes	AT&T C compilers?
interdata	Interdata 8/32	AT&T C compilers?
hp9000s200	Hewlitt Packard 9000	HP-UX
hp9000s500	Hewlitt Packard 9000	HP-UX
kl10	DEC-20 KL10 processor	U.Utah PCC?
lint	UNIX's lint	most UNIXes
m68000	Motorola M68000 family	CCI's 68000
m68k	Motorola M68000 family	Motorola SysV port
mbb ????	BBN C70	BBN UNIX?
mc68000	Motorola MC68000	Sun Microsystems, Fortune, other ports
mert	Multi-Executive Real Time??	AT&T C compilers?
nomacarg	CPP doesn't support macro args	Decus C
ON_SEL	Gould Concept 32 (obsolete OS?)	?
orion	ORION supermicro (4.1c BSD)	ORION UNIX

os	IBM's OS/360 and /370	AT&T C compilers?
PDP11	DEC's PDP-11 minicomputers	UNIX V7
pdp11	DEC's PDP-11 minicomputers	BTL C/UNIX, Decus C for RSX and RT-11, early Plexus ports?
PWB	Programmer's Workbench	PWB/UNIX
RES	Bell Labs' Comp. Sci. Research Group?	AT&T UNIX
rsx	DEC's RSX operating system on PDP-11	Decus C for RSX
RT	UNIX/RT (real time)	UNIX/RT
sel	Gould Concept 32	?
selport	Gould Concept 32	?
sun	SUN Microsystems	?
TM_DPS6	target machine is Honeywell DPS 6	Waterloo compilers
TM_L66	target machine is Honeywell Level 66	Waterloo compilers
tops20	TOPS-20 operating system for DEC-20	U.Utah PCC?
TS	UNIX/TS (timesharing system)	AT&T
TS_GCOS	target system is Honeywell GCOS 8	Waterloo compilers
TS_MOD400	target system is Honeywell GCOS 6 mod 400	Waterloo compilers
tss	IBM's Time Sharing System (for 360/370)	AT&T?
u370	UNIX on IBM/370?	AT&T
u3b	UNIX on AT&T 3B-20	AT&T
u3b2	UNIX on AT&T 3B-2	AT&T
u3b5	UNIX on AT&T 3B-5	AT&T
univac	Univac/1100 UNIX	AT&T?
unix	The you-know-what operating system	most UNIX's
vax	DEC VAX-11 minicomputers (UNIX or VMS)	UNIX, VAX-11 C, some early Unisoft 68k's
vaxllc	DEC's VAX-11 C compiler	VAX-11 C
vms	DEC's VMS operating system for VAXen	VAX-11 C
z8000	Zilog Z8000	Zilog C compiler?

MY LIST OF SUGGESTED ADDITIONS:

name	description
ccpm86	Concurrent CP/M-86
cpm68k	CP/M-68K
cpm80	CP/M-80
cpm86	CP/M-86
gnu	Stallman's (sp?) GNU (GNU's Not Unix)
i8080	Intel 8080-compatible: 8080, 8085, Z80
i8086	Intel 8086 and 8088 processors
i80186	Intel 80186 and friends
i80286	Intel 80286 and friends
mc68008	Motorola MC68008 processor
mc68010	Motorola MC68010 processor
mc68020	Motorola MC68020 processor
mpm	MP/M
msdos	Microsoft's MS-DOS operating system
pcdos	IBM and/or Microsoft's PC-DOS operating system
xinu	Purdue's XINU (Xinu Is Not Unix)
z80	Zilog Z80
z800	Zilog Z800, if they ever ship any

ACKNOWLEDGEMENTS: The data in the list is from the one in Steve Bourne's

"The UNIX System", and from Usenet articles and/or mail messages from the people listed below. If you mailed me something and don't see your name listed, then the mail got lost (so please resend it).

DBrown@HI-MULTICS.ARPA, Mike Brzustowicz <mab@aids-unix.ARPA>, John Gilmore <gnu@sun.UUCP>, Bob Gray <bob@hwcs.UUCP>, Tony Hansen <hansen@pegasus.UUCP>, Guy Harris <guy@rlgvax.UUCP>, Bob Larson <BLARSON@USC-ECLB.ARPA>, J Lepreau <j@utah-cs.UUCP>, Michael Meissner <mrm@datagen.UUCP>, Martin Minow <minow@decvax.UUCP>, Craig Partridge <craig@loki.UUCP>, John Rogers <jr@forosl.UUCP>, George Rosenberg <george@idis.UUCP>, Donn Terry <donn@hp-dcd.UUCP>, Tom Truscott <trt@rti-sel.UUCP>, Mike Zaleski <mzal@pegasus.UUCP>

I plan to maintain this list, so if anyone has any additions or corrections, please send them to me at:

UUCP: {ihnp4,cbosgd,harpo,dual,amd}!fortune!forosl!jr
ARPANET: hpda!fortune!forosl!jr@Berkeley (untested)
ENET: RHEA::DECWRL::"amd!fortune!forosl!jr" (untested)

Happy hacking!
JR (John Rogers)

From ag5@pucc-i Sat Sep 29 04:25:22 1984
Newsgroups: net.mail
Subject: UUCP ==> BITNET gateways found!!

Since I posted my request for UUCP/BITNET gateways some time ago, I have accumulated some information, which is listed below:

Apparently, two gateway sites exist for this purpose. The UUCP site at Penn State University (psuvax1) and another site (wjhl2) operate gateways. The impression I get is that one can mail thru psuvax1 in the following manner:

. . . !psuvax1!bituser%bitsite.BITNET

The wjhl2 site is just in the process of installing their BITNET mailer. They have two possible formats for mailing; you may want to try first

.. . !wjhl2!BITSITE!bituser

Please notice that the BITNET sitename is REQUIRED to be in upper case.

In a few weeks, you should be able to mail like the following:

. . . !wjhl2!bituser@BITSITE.BITNET

Since the "@" often causes trouble with mailers which your message may have to pass through, you can simply mail a letter to the letter.

Until their mailer is functioning entirely (when you can use the second and third formats), there will not be a return path generated in the message sent to the bitnet user (i.e., there will be no "Return-Path: <blah blah blah>" line.

Henry C. Mensch | Purdue University Computing Center
{decvax|ucbvax|sequent|icalqa|inuxc|uiucdcs|ihnp4}!pur-ee!pucc-i!ag5

From ag5@pucc-i Sat Sep 29 04:35:32 1984
Newsgroups: net.mail
Subject: BITNET <==> ARPA/CSNET gateway info

In my inquiries about gateways to/from UUCP, ARPA/CSNET, and BITNET I have come across a variety of helpful information. That which is below describes wiscvm.ARPA, which acts as a gateway from/to ARPA/CSNET and BITNET. This seems to work from UUCP sites thru the ARPA gateway at Berkeley, also.

BITNET - CSNET ELECTRONIC MAIL RELAY

The University of Wisconsin - Madison will provide electronic mail relay service between BITNET and CSNET beginning September 10, 1984. The gateway will reside on the WISCVM system (an IBM 4341 running VM/SP rel. 3) which is a node on both CSNET (Arpanet componenet) and BITNET.

All CSNET sites including those on Arpanet, Phonenet and Telenet will be serviced by the relay. CSNET sites need not modify their current mail systems. Mail originating on BITNET must be preceded by a Batch SMTP header and must conform to RFC 733 or RFC 822.

Communication between BITNET hosts and the gateway, WISCVM, is achieved via the IBM RSCS Networking Program. Communication between CSNET sites and WISCVM is accomplished via the SMTP/TCP/IP protocols (in the case of CSNET Phonenet, mail will also pass through the CSNET Phonenet relay, CSNET-RELAY).

Procedure for sending mail from CSNET hosts to BITNET hosts:

Recipients at BITNET hosts should be addressed as:

user_id%bitnet_host.bitnet@wiscvm.arpa

Including the domains ("bitnet" and "arpa") is preferred but not necessary.

Example: Mail to be sent to Jones at CUNYVM should be addressed as:

jones%cunyvm.bitnet@wiscvm.arpa

Mail text lines greater than 80 characters will be folded before forwarding to BITNET hosts.

Procedure for sending mail from BITNET hosts to CSNET hosts:

The mail relay runs as a disconnected virtual machine, SMTPUSER, on WISCVM. Mail to be relayed to CSNET and ARPANET hosts should be PUNCHed, NoHeader, Class M, to SMTPUSER at WISCVM.

Mail must be in Batch SMTP format (see example below). The actual SMTP mail header must be a legal RFC 822 or RFC 733 header.

The mail relay will strip off the Batch SMTP commands and re-write any RFC 733 headers to be in compliance with RFC 822.

All BITNET addresses will be re-written so that they can be replied to from CSNET hosts. Example:

Jones@CUNYVM will be re-written by the relay as:
jones%cunyvm.bitnet@wiscvm.arpa

Undeliverable mail will be returned to the address specified in the Batch SMTP "MAIL FROM" command.

Batch SMTP Example:

```
HELO CUNYVM.BITNET                          -\
TICK 0001                                     !
VERB ON                                       !-> Batch SMTP Commands
MAIL FROM:<JONES@CUNYVM.BITNET>               !
RCPT TO:<ward@uwvax.arpa>                     !
DATA                                        -/
Date: Wed, 25 Jul 84 17:02:23 cdt           -\
To: ward@uwvax                                !->  SMTP Mail Header
From: jones@cunyvm.bitnet                     !
Subject: bsmtp example                      -/
```

Mail text (Note: a blank line between mail header and text is required. A single character line containing only a period must follow the mail text)

I hope this proves helpful to someone out there in net-land!.

Henry C. Mensch | Purdue University Computing Center
{decvax|ucbvax|sequent|icalqa|inuxc|uiucdcs|ihnp4}!pur-ee!pucc-i!ag5

From john:tictoc Thu Sep 6 17:33:51 1984
To: auugn:elecvax
Subject: contribution

Here is a copy of some mail I sent to Robert Elz which might be of interest to people at sites with SC750 controllers and/or Fujitsu Eagles.

> From john Sat Sep 1 15:54:15 1984
> To: kre:munnari
> Subject: SC750 and Eagle glitches
>
> Robert:
>
> Here is the information on the problem we had with our SC750 as I mentioned to you at the AUUG meeting. The problem first manifested itself in this way: we would get "Device error on Eagle drive 0 slice x" messages, with erl = 4 (Register Modification Refused). These would tend to happen especially at times when disk activity suddenly increased (e.g., the one user on the system finished an edit and did a make). We eventually tracked it down to be due to the fact that, after a soft ECC error, sometimes the massbus byte count register in the controller contains 0xFE000000 (which seems very odd on the face of it: the drive still has 512 bytes to go but the transfer into memory is complete).
>
> Under these conditions the driver was restarting the I/O. This meant that you would get a subsequent "Invalid Map" error, since the transfer was 64K long and it never sets up any map registers except the first couple. That in turn was resulting in the "Register Modification Refused", since it thinks it must have Drive Ready after the error but in that case it doesn't, so the Drive Clear command is ignored and causes the RMR bit to be set, which is picked up later on.
>
> We applied the following change to io/gdhp.c:

```
225c225
<     if (mp->mbabcr) {
---
>     if (mp->mbabcr & 0xFFFF) {
```

> which has worked very well.
>
> We also had a fair amount of trouble booting the Eagle at first. This was due to the fact that, as soon as the boot issues a read command, it calls a routine called "rdy" which reads RMCS1 and loops until DRY is set. The problem is, with the SC750 and Eagle (it doesn't happen with our CDC 9710 RSD), sometimes the bit has not gone to zero by the first time it is read, so the routine returns immediately! We fixed this (and added an error check) in the following manner (recalling that space in the boot is short):
>
> Change to /usr/src/stand/vax/boots/gdboot2b.s:

```
40d36
<     .set    pERR,14
200,202d195
```

```
<    movl    $63,r12          # tictoc - insert some delay
< rdy0:      sobgeq  r12,rdy0
<    bbs     $pERR,r0,rperr  # tictoc
```

I hope all this is of some interest to you. If anyone is having problems with Eagles on SC750s, feel free to put them in touch with us: ours is working well now, and if we had known what we now know when we first got the machine we would have had a lot less headaches!

Regards,

John Mackin.

From stephenf:elec70b Fri Sep 7 12:07:19 1984
To: auugn
Subject: Note for AUUGN
cc: judy:basser

You might like to put a note in AUUGN about the following book (Geoff Whale has a copy and you might like to check it out). The book is "Starting with UNIX" by P.J. Brown, published by Addison Wesley.

There is a section in Chapter 2 titled "Finding other people's passwords" which suggests writing a dummy login program to catch your friends, or better yet, root, because then you can change anyone's files.

In another section, it suggests creating files called "*" in the directory of your friends, so that when they go to remove the file, they will remove all their files.

Sydney Uni uses this book as a text. Many of the students who have tried this in the last week or so (and have been brought before the powers that be for doing it) have gotten the idea from this book.

- Steve

From decvax!idis!cmucspt!bww@cmu-cs-g.arpa Fri Sep 14 20:17:26 1984
Date: 13 Sep 1984 16:55:59-EDT
From: decvax!idis!Bradley.White@CMU-CS-G.ARPA
To: idis!decvax!mulga!auugn.elecvax@cmu-cs-pt.arpa
Subject: UNIX is a trademark

From the July 1984 issue of "$ echo", published by the Software Sales and Marketing organization of the Computer Systems division of AT&T Technologies, Inc.

- -

Use of the Trademark UNIX

UNIX is an unregistered trademark of AT&T Bell Laboratories used to identify its particular brand of software. The trademark is used in conjunction with several time-sharing operating systems developed at Bell Laboratories and licensed by AT&T Technologies, and might be used in the future on other kinds of software and products.

A trademark identifies the source of a product. Some trademark owners license their trademarks for use by others. A product marked with such a trademark might come from either the trademark owner or from one of its licensees. However, currently it is AT&T's policy not to license parties outside the company to use the trademark UNIX to identify their products. There are specific provisions in our software agreements for UNIX operating systems on this point.

Notwithstanding this policy, anyone may use the trademark UNIX to refer to the UNIX operating systems developed at AT&T Bell Laboratories. However, to protect Bell Laboratories' interest in the trademark, we must ask that others use the trademark correctly. Following are several comments on correct and incorrect use of the trademark. The comments are organized in outline form for convenient reference.

- The trademark UNIX must always appear in a form that is typographically distinct.

- The trademark UNIX must be clearly and legibly identified as a trademark of AT&T Bell Laboratories at least once in any article, advertisement, or document in which the trademark appears, preferably the first time such trademark is used.

- The trademark UNIX is an unregistered trademark of AT&T Bell Laboratories. It is incorrect to use the symbol (R) in connection with the trademark UNIX or to state that UNIX is a registered trademark or service mark.

- Parties outside AT&T may not state or imply that they furnish UNIX operating systems to others and may not use the trademark UNIX in the name of software that they furnish to others. Even if such parties are licensed by AT&T to use UNIX operating systems or to furnish object code derived from such operating systems to others, they are not licensed to use the trademark to identify their product.

- The trademark UNIX may not be used in the name of a publication, business, or other organization (such as a user group).

- The trademark UNIX may not be used as a noun, but must always be used as an adjective modifying a common noun as in "UNIX operating system."

- The trademark UNIX must always be used to modify a common name for something that is a product with which the trademark is used. For example, it is incorrect to refer to "a UNIX user," "UNIX terminals," or "UNIX support." Correct usage is "a user of UNIX operating system." "terminal on a computer running a UNIX operating system," or "support for UNIX operating system."

- A way to check whether a use of the trademark is correct is to mentally insert the word "Brand" between the trademark and the common name. "UNIX Brand operating system" sounds reasonable but "UNIX Brand user" does not.

- The trademark UNIX may not be used in a hyphenated expression such as "UNIX-based" or UNIX-like."

- The trademark UNIX may not be combined with the trademark of another party unless the independence of the trademark is clear.

- Reference to "the UNIX operating system" is inappropriate. There are several UNIX operating systems. For a collective term, use "UNIX operating system," if that is what is meant.

- It is inappropriate to use the trademark UNIX in any label (such as file name, subroutine call or the like) in any software.

Minutes of the AUUG General Meeting

These are the minutes of the first General Meeting of the Australian UNIX systems Users Group, held at the University of Melbourne on August 27, 1984.

Minutes of the Inaugural meeting.

Meeting opened 13:45 on the 27th August 1984.

Convener John Lions took the chair.

Amendments to the draft constitution as published in AUUGN:
(13) replace "A general meeting of the AUUG may"
by "An ordinary general meeting of the AUUG shall".
Moved Ken McDonell, Seconded Robert Elz.
Carried.

(15) replace "six" by "twelve".
Moved Colin Webb, Seconded John O'Brien.
Carried.

(14) replace "three calendar months" by "two calendar months".
Moved John O'Brien, seconded Chris Maltby.
Carried.

Motion that "the constitution as amended be adopted."
Moved Juris Reinfelds, Seconded Robert Elz.
Carried.

Motion that "the Management Committee seek legal opinion on the constitution and report back to the next meeting of the AUUG."
Moved Juris Reinfelds, Seconded Greg Rose.
Carried.

Motion for a "vote of thanks to John Lions and others involved in in drafting the constitution."
Moved Colin Webb, Seconded by general acclaim.

Election of officers and committee of management.

No records kept of nominators, seconders, acceptances or the order of nominations.

As there were two positions for returning officers, and two nominations, the returning officers assumed their positions for the counting of subsequent written ballots.

President:	John Lions (elected unopposed)
Secretary:	John Macken
	Greg Rose (elected)
Treasurer:	Chris Maltby (elected)
	Geoff Cole
Management Committee:	(four members to be elected)
	Peter Ivanov (declined)
	Ken McDonell (elected)
	Ross Nealon

	David Horsfall
	Robert Elz (elected)
	Ron Baxter
	Doug Richardson
	Piers Dick-Lauder (elected)
	Tim Roper (elected)
	Rod Curtin
	Phil Chadwick
Returning Officers:	(two to be elected)
	John O'Brien (elected unopposed)
	Colin Webb (elected unopposed)
Auditor:	James Mann (elected unopposed)

Meeting adjourned 14:55.

Meeting re-opened 09:00 on the 28th of August 1984.

Motion that "current subscribers to AUUGN may become financial members to January 1, and that the cost for Ordinary members be $45, $30 for Student members.
Moved by Peter Ivanov, seconded John Lions from chair.

Amendment that "$45" become "$50".
Moved Robert Elz, Seconded Richard Hibbard.
Amendment carried.

Amended motion carried.

Suggestions for subsequent meetings:

Wollongong	Feb 85.
Queensland	Aug 85
Perth	Feb 86

Motion that "The first Annual General Meeting be held on or about the 29th August 1985 in Brisbane."
Moved Richard Hibbard, Seconded Robert Elz.
Carried.

Motion that "the next meeting be held in Wollongong in February 1985."
Moved John Lions from chair, Seconded Greg Rose.
Carried.

The elected officers were announced.

Meeting closed 9:30.

Adopted 27/8/84
Rules and By-Laws for
the Australian Unix systems User Group

I. RULES

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

(2) Office of the Association.

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

(3) Definitions.

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

(4) Aims.

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

For the furtherance of these aims and to achieve its purposes, the AUUG may carry out any or all of the following activities: conduct technical meetings, conferences, discussion groups, panels, lectures and other types of meeting; prepare and distribute a newsletter and other publications; collect software and distribute said software to its members for their use; verify licenses of members for the purposes of administering the services of the AUUG; subscribe to or cooperate with or affiliate with or amalgamate with other associations formed elsewhere with similar aims; accumulate assets; and establish and promote other activities not included in the above list consistent with its aims for the benefit of its members.

(5) Eligibility for Membership.

Any individual or organisation who subscribes to the aims of the association, and who agrees to be bound by its rules and regulations and who has not been previously expelled from the association shall be eligible to join the AUUG.

(6) Application for Membership.

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

(7) Commencement of Membership.

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

(8) Renewal of Membership.

Upon completion of the initial membership period and any subsequent periods, membership may be renewed for a further period of twelve months by payment of an additional annual membership subscription.

(9) Termination of Membership.

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

(10) Amount of Guarantee.

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

(11) **Expulsion of Members.**
Upon receipt of a petition so requesting from twenty or more members, or half the membership, whichever is less, the Management Committee shall call upon any member to explain any alleged misconduct, and the Management Committee shall have power to suspend or expel any member who in its opinion has either been guilty of misconduct or has acted prejudicially to the interests of the AUUG or who has wilfully infringed any of the Rules or By-Laws of the AUUG.

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

(13) **Ordinary General Meetings.**
A ordinary general meeting of the AUUG shall be called by the Management Committee in conjunction with any technical meeting or conference or other function where attendance by a quarter or more of the voting members is expected by the Management Committee.
The business that may be conducted at such a meeting shall be as prescribed in the By-Laws. Notice of such meetings together with the agenda shall be sent to all voting members by mail at least four weeks before the date set for the meeting.

(14) **Extraordinary General Meetings.**
Upon receipt of a petition so requesting from twenty or more members, or half the membership, whichever is less, the Secretary shall call an Extraordinary General meeting of the AUUG for a date no later than two calendar months after receipt of the petition, and the business of the meeting shall be confined to matters described in the petition and to other matters specifically provided for in these rules and recorded in the written agenda sent to all members by mail at least four weeks before the date set for the meeting.

(15) **Quorum for a General Meeting.**
For each general meeting, the quorum shall be twelve members personally present and entitled to vote.

(16) **Officers.**
The Officers of the AUUG shall be: the **President**; the **Secretary**; the **Treasurer**; the **Returning Officer**; the **Assistant Returning Officer**; and the **Auditor.**

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

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

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

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

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

(21) **Management Committee Meetings.**
The Management Committee shall meet formally at least twice per year. Notification of time, place and agenda for each meeting shall be made in writing to each member of the Committee by the Secretary at least four weeks in advance. All members of the AUUG are entitled to be present at such meetings, and may speak when invited by the Chairman, but only members of the Management Committee may vote. The quorum for such meeting shall be four.

Resolutions of the committee shall require a simple majority of the members present and voting. The chairman shall have a casting vote in the event of a tie.

(22) **Distribution of Income.**
The income and property of the AUUG however derived shall be applied solely towards the aims and purposes of the AUUG as set out in these Rules, and no portion thereof shall be paid or transferred directly or indirectly by way of dividend to any member of the AUUG at any time.
The AUUG shall not appoint a person who is a member of the Management Committee to any office in the gift of the association to the holder of which there is payable any remuneration by way of salary, fees or allowances.
Notwithstanding the above the AUUG may compensate the reasonable expenses actually incurred by any member in the conduct of the business of the AUUG under the direction of the Management Committee.

(23) **Chapters.**
Ten or more members of the AUUG may petition the Management Committee to form a **chapter** of the AUUG.
General rules for the organisation, operation, obligations and privileges of chapters shall be as resolved by the Management Committee or the membership as a whole from time to time.
Each chapter shall appoint a chapter committee consisting of at least a Chapter Chairman and a Secretary/Treasurer.
The chapter committee may convene meetings consistent with the aims of the AUUG, but may not enter into any financial commitments on behalf of or in the name of the AUUG except with the written approval of the Management Committee.

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

(25) **Dissolution of the AUUG.**
Upon receipt of a petition requesting the dissolution of the AUUG from twenty or more members, or half the membership, whichever is less, the Secretary shall arrange for the question to be put to the membership by ballot no later than one month after the date that he receives the petition.
If two-thirds of the members voting agree, the AUUG shall be dissolved. If upon the dissolution of the AUUG there remains after satisfaction of all its debts and liabilities any property whatsoever, the same shall not be paid to or distributed among the members or Chapters if any, but shall be given or transferred to some public educational institution, or other institution to be determined at or before the time of dissolution by resolution of the membership.

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

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

Adopted 27/8/84
Rules and By-Laws for
the Australian Unix systems User Group

II. BY-LAWS.

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

(29) **Ordinary Members.**
Any person who is eligible to be a member may become an Ordinary Member.

(30) **Institutional Members.**
Any person or organisation who is eligible to be a member may become an Institutional Member.

(31) **Student Members.**
Any full-time student who is eligible to be a member may become a Student Member.

(32) **Honorary Life Members.**
Any person who is an Ordinary Member of at least five years standing and who has rendered special services to the AUUG may be elected via a ballot of the members as an Honorary Life member.

(33) Rights of Members.

Each member shall be entitled to attend all meetings of the AUUG, including meetings of the Management Committee, provided any prescribed attendance fee is paid.
Each member shall be sent a copy of the association's newsletter.
Each member entitled to vote in a ballot shall be sent notice in writing of all ballots and copies in writing of the annual reports of the Secretary, Treasurer and Auditor.

(34) Obligations of Members.

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

(35) Voting Rights.

All Ordinary, Institutional and Honorary Life Members whose membership is current shall be entitled to cast one vote. Any voting member may award his proxy to another voting member for the period of a single General meeting providing he so notifies the Secretary in writing at least 24 hours before the appointed time of commencement of the meeting.

(36) Membership Subscriptions and Fees.

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

(37) Chairman of Meetings.

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

(38) Duties of the Secretary.

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

(39) Duties of the Treasurer.

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

(40) Execution of Contracts.

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

(41) Disbursements.

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

(42) Conduct of General Meetings.

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

(43) **Voting.**

All voting by the members with respect to the election of Officers and General Committee Members, with respect to the election of Honorary Life Members, with respect to changes to these Rules and By-Laws, and all other substantive matters shall be conducted by postal ballot.

Every voting member of record as of the date of entry of a ballot into the mails shall be entitled to vote in the ballot.

On all questions to be put to a ballot, the Secretary shall designate a date for the ballot to be placed in the mails, and the due date shall be four weeks after that date. The Returning Officer shall nominate the address to which voters shall return completed ballot papers by mail. A ballot will not be counted if it is received after the due date or if the ballot paper does not comply with the instructions printed on it.

The ballots will be received by the Returning Officer, and counted by him and the Assistant Returning Officer. The Returning Officer shall report the result of the ballot in writing to the Secretary no later than two weeks after the due date.

(44) **Conduct of Elections.**

Elections shall be held annually for all positions of Officer and General Committee Member.

Nominations for each position shall be received by the Secretary up until the first day of May each year. Each nomination must be in writing, must name the position or positions sought, must be signed by at least three voting members, and must be countersigned by the nominated member who must be a financial voting member of the AUUG.

The Management Committee shall ensure that at least one valid nomination is obtained for each position such that if no further nominations are received all offices and positions may be filled. Where only one valid nomination is received for a particular position by the close of nominations, the nominee shall be declared elected forthwith, and no ballot for that position shall be held.

Within first seven days of May, the Secretary shall advise the Returning Officer of all valid nominations received, and if a ballot is required shall advise him of a date no later than the fifteenth day of May for the ballot for all contested positions, and shall provide him with a list of voting members.

While any Ordinary Member may be nominated to more than one office or position, no person shall be elected to more than one position. Ballots shall be determined in the following order: for President, for Secretary, for Treasurer, for General Committee Member, for Returning Officers, and for Auditor.

(45) **Election of Honorary Life Members.**

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

Who's Who

The following subscribers, to Volume 5 of AUUGN, are eligible to become founding members of the AUUG. I have NOT included institutions in this list. Apply to the AUUG Secretary for consideration.

Ros Anderson
Trevor Barton
Peter J. Billam
Colin Boswell
Stephen M. Brady
Keith Brister
Chris Campbell
Dr. Eddy K. S. Chan
Dr. Chris Coles
R E M Cooper
Kevin Dawson
Wayne Edwards
M.C. Er
Desmond Fitzgerald
Dr. Ivan Fris
Mr J. Di Giacomo
Ian Grigg
Robert Hegedus
Bill Hibbert
S.K. Hockley
David Horsfall
Peter Ivanov
Ian Johnstone
Martin Kenny
Carl D. Kneipp
Steven Landers
Prof. John Lions
Peter Mason
David McSweeney
David Millsom
Dr. R B Newell
Mr. Paul Obrien
Peter Pamment
Mr. Bill Petheram
Michael J. F. Poulsen
Roy Rankin
Ted Rigby
Ken Robinson
Tim Roper
Jim Rutherford
David Sanchez
Warren Simon
Graham Smith
Armando P. Stettner
Phil Sutherland
Carole Sweaney
Mr. Milton F. Thrasher
Colin Webb
Dr. Bradley W. White
T. Willoughby

R. Balsdon
David Bassell
Edd Birch
Brian Boutel
Daniel Braniss
Perry Brown
Malcolm Cardis
Roger A. Clarke
David Colhoun
Philip County
Mr John DeAno
H. D. Ellis
John Field
Stephen Frede
Leslie B. Frohoff
Paul Gillis
Lindsay Harris
David Herd
Roger G. Hicks
D P Hodgson
Steven Hudson
Dennis Jarvis
Steve Jordan
Harry Khoury
Bob Kummerfeld
Mr Yong Hiong Leong
Dr. R.J. Lobb
Craig McGregor
Felicia Meagher
Lyn Moon
Kazuhiko Nishioka
Alan Owen
Ian Paton
Michael A. Podhorodecki
Chris Price
Hamish Reid
Ian Roberts
Michael A. Robinson
Michael Rourke
Colin Ruthven
Gershon Shamay
Lionel Singer
Ian Smith
Rick Stevenson
Peter Swain
Prof. G. Tate
K C Toh
Peter Webb
Oki Widjaja
Norman Wilson

Christopher Barter
Michael Belfer
D. Blackman
Don Bowen
Mike Brennan
Henry W. and Edith Burk
David Carrington
Jeffrey Cohen
Peter Collinson
Dr. Cameron Davidson
Leeanne Diggelmann
M. J. Ellis
Leon Fittinghoff
Adrian Freed
Ross Gayler
Richard Grevis
C.G. Hartmann
Prof. J. B. Hext
Kevin Hill
John Holden
David Hunt
Steve Jenkin
Shirley Keeting
John Knaggs
Robin Lamacraft
M J Liebelt
Tim Long
Jim McKie
G Michalk
Graham Neale
Albert Nymeyer
W. David Owen
Veronica Paul
Zdravko Podolski
Greg Quinn
Kevin Reville
David Robinson
John Rogers
Chris Rowles
Claude Sammut
Michael Sidhom
Anne Smith
Malcolm Smith
Tom Strong
Geoff Swan
Derek Thomas
Bob Trewin
Rob Webb
Nigel Williams
Clive Winkler

Richard Wolff John Wulff

The following subscribers have been granted "founding membership" of the AUUG.

Derek Austin	Peter D. Barnes	Rod Bilson	B.C.P. Borun
Bruce Butterfield	Phil Chadwick	Geoff Cole	Tom Crawley
Robert Elz	Darryl Godfrey	Clary Harridge	Glenn Huxtable
Eoin Hyden	Greg Kable	Michael Kearney	Chris Maltby
Ken McDonell	Willma Nelowkin	Dr. Y. Kuang Oon	Rob Pike
Robert Posener	Greg Rose	Munro Saunders	Tim Segall
David Stirling	Ian F. Turner	Steffan P. Weiss	

The following people have been accepted as "normal members" of the AUUG.

K. W. Anderson	Ron Baxter	Robert Douglas	Gordon Helliwell,
Richard C. Hibbard	James D. Mann	Graham Menhennitt	Stephen Prince
Gregory Sharp	Mr. Robert L. Smith	Peter Wishart	

Finally we have two subscribers to the newsletter, Rick Southern and T. A. Nemeth.

Australian UNIX* systems User Group
(AUUG)

Current Subscriber Membership Application

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

Signed _____ Date _____

==========

Name _____

Mailing address for AUUG information _____

_____

_____

UNIX Network address _____

	YES	NO
I agree to my name and address being made available to software/wardware vendors	☐	☐

==========

Office use only 10/84

* UNIX is a trademark of AT&T Bell Laboratories

Australian UNIX* systems User Group
(AUUG)

Membership Application

I, _____ do hereby apply for ordinary($50)/student(30)** membership of the Australian UNIX systems User Group and do agree to abide by the rules of the association especially with respect to non-disclosure of confidential and restricted licensed information. I understand that the membership fee entitles me to receive the Australian UNIX systems User Group Newsletter and I enclose payment of $_____ herewith.

Signed _____ Date _____

==========

Name _____

Mailing address for AUUG information _____

_____

_____

Telephone number (including area code) _____

UNIX Network address _____

	YES	NO
I agree to my name and address being made available to software/wardware vendors	☐	☐

==========

Student Member Certification

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

Expected date of graduation _____

Faculty signature _____ Date _____

==========

Office use only 10/84

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

Australian UNIX* systems User Group Newsletter
(AUUGN)

Subscription Application

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

Signed _____ Date _____

==========

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

Name _____

Mailing address _____

_____

_____

Telephone number (including area code) _____

UNIX Network address _____

	YES	NO
I agree to my name and address being made available to software/wardware vendors	☐	☐

10/84

* UNIX is a trademark of AT&T Bell Laboratories

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

+61 2 697 4040
(and eventually +61 2 697 4042)