

AUUGN

AUUG Inc. Newsletter

Volume 14, Number 2

April 1993

The AUUG Incorporated Newsletter

Volume 14 Number 2

April 1993

CONTENTS

AUUG General Information	3
Editorial	5
AUUG Institutional Members	6
AUUG President's Report	8
Change in AUUG Membership Fees	9
New COSEy alliance formed <i>Adrian Booth</i>	10
AUUG'93 Update	11
Summer Conference Reviews	12
SESSPOOLE Committee AGM Report	13
SESSPOOLE Events	14
WAUG and Perth News	15
WAUG Meeting Review	16
AUUG Canberra Chapter- Upcoming Events	17
UniForum NZ'93	17
Calendar of Events	18
Announcement of SAGE-AU and USENIX/SAGE	19
SAGE-AU Information Sheet	20
SAGE-AU Inaugural Conference and Annual General Meeting	21
Open System Publications	23
ACSnet Survey	24
Book Reviews	27
Books- Special Offers	
Unix Power Tools	30
Unix User's Handbook	34
Wizard's Bookshelf	35
TMX Advertisement	36
!AUUGN - from AUUGN Volume 1, Number 1	
Letter from the promised land <i>John Lions</i>	37
<i>vpcheck</i> A Daemon to Balance Vector and Scalar Usage <i>Frank Crawford</i>	40
Softway Advertisement	45
386BSD: A Look Under The Hood <i>Andrew McRae</i>	46

Load Sharing in a Distributed Environment	<i>Andy Bond</i>	54
From login: - Volume 18, Number 1		
An Update on UNIX-Related Standards Activities		66
Unix Tricks 'n' Traps		86
User Support Mailbox		87
Summary of Management Committee Minutes - 18th February 1993		89
AUUG Membership Categories		92
AUUG Forms		93

Copyright © 1993 AUUG Incorporated. All rights reserved.

AUUGN is the journal of AUUG Incorporated, an organisation with the aim of promoting knowledge and understanding of Open Systems including but not restricted to the UNIX* system, networking, graphics, user interfaces and programming and development environments, and related standards.

Copying without fee is permitted provided that copies are made without modification, and are not made or distributed for commercial advantage. Credit to AUUGN and the author must be given. Abstracting with credit is permitted. No other reproduction is permitted without prior permission of AUUG Incorporated.

* UNIX is a registered trademark of UNIX System Laboratories, Incorporated

AUUG General Information

Memberships and Subscriptions

Membership, Change of Address, and Subscription forms can be found at the end of this issue.

Membership and General Correspondence

All correspondence for the AUUG should be addressed to:-

The AUUG Secretary,
P.O. Box 366,
Kensington, N.S.W. 2033.
AUSTRALIA

Phone: (02) 361 5994
Fax: (02) 332 4066
Email: auug@munniari.oz.au

AUUG Business Manager

Liz Fraumann,
P.O. Box 366,
Kensington, N.S.W. 2033.
AUSTRALIA

Phone: +61 2 953 3542
Fax: +61 2 953 3542
Email: eaf@softway.sw.oz.au

AUUG Executive

President

Phil McCrea
phil@softway.oz.au
Softway Pty. Ltd.
79 Myrtle Street
Chippendale NSW 2008

Vice-President

Glenn Huxtable
glenn@cs.uwa.oz.au
University of Western Australia
Computer Science Department
Nedlands WA 6009

Secretary

Peter Wishart
pjw@lobo.canberra.edu.au
EASAMS Australia
Level 6
60 Marcus Clark St.
Canberra ACT 2600

Treasurer

Frank Crawford
frank@atom.ansto.gov.au
Australian Supercomputing Technology
Private Mail Bag 1
Menai NSW 2234

Committee
Members

Rolf Jester
rolf.jester@sno.mts.dec.com
Digital Equipment Corporation
P O Box 384
Concord West NSW 2138

Chris Maltby
chris@softway.sw.oz.au
Softway Pty. Ltd.
79 Myrtle Street
Chippendale NSW 2008

John O'Brien
john@wsa.oz.au
Whitesmiths Australia P/L
#5 Woods Centre
ANSTO Business & Tech. Park
Lucas Heights NSW 2234

Michael Paddon
mwp@iconix.oz.au
Iconix Pty Ltd
851 Dandenong Rd
East Malvern VIC 3145

Greg Rose
ggr@acci.com.au
ACCI
723 Swanston St
Carlton VIC 3053

AUUG General Information

Next AUUG Meeting

The AUUG'93 Conference and Exhibition will be held from the 27th to 30th September, 1993, at the Sydney Convention and Exhibition Centre, Darling Harbour, Sydney.

Advertising

Advertisements to be included in AUUGN are welcome. They should conform to the standards of other contributions (see page 5). Advertising rates are \$120 for a quarter page, \$180 for half a page, \$300 for the first A4 page, \$250 for a second page, \$500 for the inside cover and \$750 for the back cover. There is a 20% discount for bulk ordering (ie, when you pay for three issues or more in advance). Contact the business manager for details.

Mailing Lists

For the purchase of the AUUGN mailing list, please contact the AUUG secretariat, phone (02) 361 5994, fax (02) 332 4066.

Back Issues

Various back issues of the AUUGN are available. For availability and prices please contact the AUUG secretariat or write to:

AUUG Inc.
Back Issues Department
PO Box 366
Kensington, NSW, 2033
AUSTRALIA

Conference Proceedings

A limited number of the Conference Proceedings for AUUG'92 are still available, at \$50 each. Contact the AUUG secretariat.

Acknowledgement

This Newsletter was produced with the kind assistance of and on equipment provided by the Australian Nuclear Science and Technology Organisation.

Disclaimer

Opinions expressed by authors and reviewers are not necessarily those of AUUG Incorporated, its Newsletter or its editorial committee.

AUUG Newsletter

Editorial

Welcome to AUUGN Volume 14 Number 2. In this issue we have two summaries of AUUG's summer conferences, one from Tasmania and the other from the Northern Territory. Two papers from the Sydney conference have also been included and a paper from UniForum NZ'92. Information on UniForum NZ'93 has been supplied. I decided to print it, although when members receive this, there will only be a short time before the conference.

Other articles include, information on SAGE-AU, local chapters upcoming events and some important information on AUUG membership fees which should be read by all members.

A number of book reviews have been provided including *two* reviews on UNIX Power Tools from O'Reilly and Associates/Bantam. Unfortunately, due to a number of reasons Dave Newton is unable to continue in the role as a book review editor. Frank and I will handle book reviews for the time being.

Finally, thanks to WAUG for the number of articles contributed to this issue of AUUGN. I hope to see more contributions from other local chapters.

Jagoda Crawford

AUUGN Correspondence

All correspondence regarding the AUUGN should be addressed to:-

AUUGN Editor,
P.O. Box 366,
Kensington, N.S.W. 2033.
AUSTRALIA

Phone: +61 2 717 3885
Fax: +61 2 717 9273
Email: auugn@munnari.oz.au

AUUGN Book Reviews

Anyone interested in reviewing books or with book reviews to submit for publishing in AUUGN please contact the AUUGN editor.

Contributions

The Newsletter is published approximately every two months. The deadlines for contributions for the next issues of AUUGN are:

Volume 14 No 3	Friday 28th May
Volume 14 No 4	Friday 30th July
Volume 14 No 5	Friday 24th September
Volume 14 No 6	Friday 26th November

Contributions should be sent to the Editor at the above address.

I prefer documents to be e-mailed to me, and formatted with troff. I can process mm, me, ms and even man macros, and have tbl, eqn, pic and grap preprocessors, but please note on your submission which macros and preprocessors you are using. If you can't use troff, then just plain text or postscript please.

Hardcopy submissions should be on A4 with 30 mm margins, and 30 mm left at the bottom so that the AUUGN footers can be pasted on to the page. Small page numbers printed in the footer area would help.

AUUG Institutional Members as at 02/04/1993

A.N.U.	Customised Software Solutions Centre
AAII	Cyberdyne Systems Corporation Pty Ltd
Adept Software	Cyberscience Corporation Pty Ltd
Alcatel Australia	Data General Australia
Allaw Technologies	Datacraft Technologies
Amdahl Pacific Services	Deakin University
Andersen Consulting	Deakin University
ANI Manufacturing Group	Defence Housing Authority
ANSTO	Defence Service Homes
Anti-Cancer Council of Victoria	Dept of Agricultural & Rural Affairs
ANZ Banking Group/I.T. Development	Dept of Business & Employment
Attorney-General's Dept	Dept of Defence
AUSOM Inc.	Dept of Education, Qld
Ausonics Pty Ltd	Dept of Industrial Relations,
Auspex Systems Australia	Employment, Training & Further Education
Australian Airlines Limited	Dept of Planning & Housing
Australian Archives	Dept of the Premier and Cabinet
Australian Bureau of Agricultural and Resource Economics	Dept. of Conservation & Environment
Australian Bureau of Statistics	Dept. of Defence
Australian Computing & Communications Institute	Dept. of the Premier and Cabinet
Australian Defence Industries Ltd	Dept. of the Treasury
Australian Electoral Commission	Dept. of Transport
Australian Museum	DEVETIR
Australian National Parks & Wildlife Service	Digital Equipment Corp (Australia) Pty Ltd
Australian Software Innovations	Easams (Australia) Ltd
Australian Taxation Office	EDS (Australia) Pty Ltd
Australian Technology Resources (ACT) Pty Ltd	Electronic Financial Services Limited
Australian Wool Corporation	Emulex Australia Pty Ltd
Automold Plastics Pty Ltd	Equinet Pty Ltd
AWA Defence Industries	Ericsson Australia Pty Ltd
B & D Australia	ESRI Australia Pty Ltd
Bain & Company	FGH Decision Support Systems Pty Ltd
BHA Computer Pty Limited	Financial Network Services
BHP CPD Research & Technology Centre	Fire Fighting Enterprises
BHP Information Technology	Flinders University
BHP Minerals	Fremantle Port Authority
BHP Petroleum	Fujitsu Australia Ltd
BHP Research - Melbourne Laboratories	G. James Australia Pty Ltd
BHP Research - Newcastle Laboratories	GCS Pty Ltd
BICC Communications	Geelong and District Water Board
Bond University	Genasys II Pty Ltd
Burdett, Buckeridge & Young Ltd."	General Automation Pty Ltd
Bureau of Meteorology	GeoVision Australia
Bytecraft Pty Ltd	GIO Australia
C.I.S.R.A.	Golden Circle Australia
C.I.S.U.	Great Barrier Reef Marine Park Authority
Cape Grim B.A.P.S	Gribbles Pathology
Capricorn Coal Management Pty Ltd	Gunnedah Abattoir
Chief Secretary's Dept	Haltek Pty Ltd
CITEC	Hamersley Iron
Classified Computers Pty Ltd	Harris & Sutherland Pty Ltd
Co-Cam Computer Group	Hermes Precisa Australia Pty. Ltd.
Codex Software Development Pty. Ltd.	Honeywell Ltd
Cognos Pty Ltd	Honeywell Ltd
Colonial Mutual	Hong Kong Jockey Club Systems (Australia) Pty Ltd
Com Net Solutions	I.B.A.
Com Tech Communications	IBM Australia Ltd
Commercial Dynamics	Iconix Pty Ltd
Communica Software Consultants	Information Technology Consultants
Composite Buyers Ltd	Insession Pty Ltd
Computechnics Pty Ltd	Insurance & Superannuation Commission
Computer Sciences of Australia Pty Ltd	Internode Systems Pty Ltd
Computer Software Packages	Ipec Management Services
Corinthian Engineering Pty Ltd	IPS Radio & Space Services
CSIRO	James Cook University of North Queensland
Curtin University of Technology	JTEC Pty Ltd

AUUG Institutional Members as at 02/04/1993

Knowledge Engineering Pty Ltd
KPMG Solutions
Labtam Australia Pty Ltd
Land Information Centre
Land Titles Office
Leeds & Northrup Australia Pty. Limited
Logica Pty Ltd
Logical Solutions
Macquarie University
Mayne Nickless Courier Systems
McDonnell Douglas Information Systems Pty Ltd
Medical Benefits Funds of Australia Ltd.
Mentor Technologies Pty Ltd
Meridian Information Services Pty Ltd
Metal Trades Industry Association
Mincom Pty Ltd
Minenco Pty Ltd
Mitsui Computer Limited
Motorola Computer Systems
Multibase Pty Ltd
National Library of Australia
NCR Australia
NEC Australia Pty Ltd
NSW Agriculture
Office of Fair Trading
Office of the Director of Public Prosecutions
Olivetti Australia Pty Ltd
Open Software Associates Ltd
Oracle Systems Australia Pty Ltd
OSIX Pty Ltd
Ozware Developments Pty Ltd
Pacific Star Communications
Paxus
Philips PTS
Port of Melbourne Authority
Powerhouse Museum
Prentice Hall Australia
Process Software Solutions Pty Ltd
Prospect Electricity
pTizan Computer Services Pty Ltd
Public Works Department
Pulse Club Computers Pty Ltd
Pyramid Technology Corporation Pty Ltd
Qantek
Quality By Design Pty Ltd
Redland Shire Council
Release4
Rinbina Pty Ltd
Royal Melbourne Institute of Technology
SBC Dominguez Barry
Scitec Communication Systems
Sculptor 4GL+SQL
SEQEB Business Systems
SEQEB Control Centre
Shire of Eltham
Siemens Nixdorf Information Systems Pty Ltd
Snowy Mountains Authority
Software Developments
Softway Pty Ltd
Sony Technology Centre of Australia
South Australian Lands Dept
St Vincent's Private Hospital
St. Gregory's Armenian School
Stallion Technologies Pty Ltd
Standards Australia
State Bank of NSW
State Super (SSIMC)
Steedman Science and Engineering
Steelmark Eagle & Globe
Swinburne Institute of Technology
Sydney Electricity
Sydney Ports Authority
System Builder Development Pty Ltd
TAB of Queensland
Tattersall Sweep Consultation
Technical Software Services
Telecom Australia Corporate Customer
Telecom Network Engineering Computer
Support Services
Telecom Payphone Services
The Far North Qld Electricity Board
The Fulcrum Consulting Group
The Opus Group Australia Pty Ltd
The Preston Group
The Roads and Traffic Authority
The Southport School
The University of Western Australia
TNT Australia Information Technology
Toshiba International Corporation Pty Ltd
Tower Software Engineering Pty Ltd
Tower Technology Pty Ltd
Tradelink Plumbing Supplies Centres
Triad Software Pty Ltd
TurboSoft Pty Ltd
TUSC Computer Systems
UCCQ
Unidata Australia
Unisys
Unisys Australia Ltd
UNIVEL
University of Adelaide
University of Melbourne
University of New South Wales
University of South Australia
University of Sydney
University of Tasmania
University of Technology, Sydney
UNIX System Laboratories
Unixpac Pty Ltd
Victoria University of Technology
VME Systems Pty Ltd
Wacher Pty Ltd
Walter & Eliza Hall Institute
Wang Australia Pty. Ltd.
Water Board
Western Mining Corporation
Workstations Plus
Zircon Systems Pty Ltd

AUUG President's Report

Who defines what 'Open' means?

The topic of 'Open Systems' is now on everyone's lips. In much the same vein as cigarettes in the the 70s suddenly became 'kind to your throats'(!) after the cancer scare was first mooted, almost every computer vendor is now touting themselves as being 'Open'. They have to – every strategic plan that is being written at present recommends a move to Open Systems.

But what does 'Open' mean? Well, it all depends who you ask. The companies still fortunate enough to have a proprietary environment to protect, generally refer to connectivity (or interoperability) when they use the term 'Open'. True, interoperability is certainly one of the attributes of an Open System, but only one.

The Government's View

It is interesting to note the view of the Australian Government when it comes to Open Systems. The Information Exchange Steering Committee (IESC) is the Commonwealth Government's central coordinating and advisory body for developing and promoting IT and Telecommunications (IT&T) policies, strategies and standards.

More specifically the IESC's Open Systems Subcommittee (OSSC), which is chaired by the Department of Finance, has responsibility for Open Systems issues. Within the Department of Finance, the Information Technology and Systems Group (IT&SG) produces an informative publication several times a year – the *Open Systems Newsletter* – which provides an insight into the Government's view of Open Systems. They also produce an equally informative newsletter of a more general nature called *IT News*.

It is pleasing to note that the outlook of the OSSC appears to have changed from that of purely being connectivity based, to a view which now covers the other main attributes of Open Systems:

- scalability
- portability
- interoperability
- compatibility

The Government Guide to Open Systems

The OSSC's current view is contained in a report entitled *Government Guide to Open Systems* (GGOS). Its purpose is to raise awareness of Open Systems issues and to assist Government departments and agencies in formulating strategies to move their computing platforms to a more open environment.

A number of "open frameworks" were reviewed by the OSSC to assess their suitability to meet Commonwealth requirements, and the Application Portability Profile of the National Institute of Standards & Technology (NIST) of the US Department of Commerce, was utilised as the basis of the Guide on account of its close alignment with international standards.

The GGOS has been developed in consultation with Government agencies, industry organisations and overseas governments. Whilst GGOS does not specifically endorse UNIX, the UNIX fraternity will be delighted with the contents of this document, as it recommends the following non proprietary standards (amongst other things):

User Interface:	X Windows
Operating System:	Posix.1,2 and 6 compliance
Languages:	ANSI C
Graphics:	GKS
Comms:	OSI
	TCP/IP (interim)

The OSSC, particularly their Department of Finance members Alan Maclean and Ann Whitehead, should be congratulated on producing GGOS. Mind you, they have taken a (very large!) leaf out of NIST's book. But, whatever the background, the GGOS is worth supporting.

P. McCrea

Changes in AUUG Membership Fees

AUUG Inc has maintained the current level of membership fees for the last four years despite increasing costs. While other societies (e.g. the Australian Computer Society) have seen their membership fees increase in the multiple hundreds of dollars, the AUUG has kept its annual fees for ordinary members down to \$78.

This year we have reluctantly been forced to raise the membership fees to ensure AUUG Inc remains financially viable. In recent years AUUGs profit from the winter conferences has been negligible, resulting in a depletion of our reserves. To rectify this AUUG has recently renegotiated our profit sharing arrangements with ACMS for the winter conferences, resulting in a greater percentage to AUUG, but also with an increase in the upfront cost. Also, Over the past few years AUUG has adopted a more professional approach to our activities with the introduction of a paid Secretariat and by employing a business manager. These factors, together with increasing costs, has forced the committee to raise the membership fees for ordinary members (by \$12) and institutional members (by \$25).

The committee has also been concerned about the relatively small number of student members in AUUG. To encourage more student members and recognising the financial constraints on full time students we have dropped the student membership by \$20.

The following new fees will apply from 1st July 1993:

Ordinary Members:	\$90 (an increase of \$12)
Institutional Members:	\$350 (an increase of \$25)
Student Members:	\$25 (a decrease of \$20)
Newsletter Subscription:	\$90 (unchanged)

With the increase in activities at a local level through the strengthening of local chapters and the continuing success of our national events like AUUG93, the committee looks forward to the continued success of AUUG Inc and an even brighter future.

Peter Wishart AUUG Inc. Secretary

The Joy of Being an



Getting out this publication is no picnic.

- If we print jokes, people say we are silly. If we don't, they say we are too serious.
- If we clip things from other sources, we are too lazy to write them ourselves. If we don't, we are too fond of our own stuff.
- If we don't print contributions, we are also too fond of our own stuff.
- If we make a change in the other writing, we are too critical; if we don't, we are asleep.



- Now, as likely as not, someone will say we swiped this from some other publication. We did!

Acknowledgement: we (Jagoda Crawford) swiped this from DECUS news Volume 15 Number 1, who swiped it from University of WA's UNINEWS of 23-Mar-92, who swiped it from INTUITIONS 6-Mar, who swiped it from INCITE 17-Feb, who swiped it from THE CAPE LIBRARIAN 35, 7-Aug-91, who swiped it from US PUBLIC.

New COSEy alliance formed

Has the UNIX marketplace (finally) got its act together?

It may have taken the threat of Windows NT to do it, but the UNIX marketplace finally seems to be getting its act together.

Summary of a recent press release:

Six vendors - HP, IBM, SCO, SunSoft (the software subsidiary of Sun Microsystems), Univel (the joint venture between AT&T and Novell) and UNIX Systems Laboratories have announced that they will deliver a common open software environment (COSE) across their UNIX system platforms.

The six vendors have defined a specification for a "common desktop environment" that gives end users a consistent look and feel across all of these platforms. This specification includes a consistent set of APIs for the desktop that will run across all of these systems. The end user will also see a consistent set of desktop productivity tools, including electronic mail and group calendaring.

The vendors have also decided to adopt common networking products; have endorsed specifications, standards and technologies in the areas of graphics, multimedia, and object technology; and have announced a working group in the area of system administration.

OSF will submit the Motif specification to X/Open for inclusion in X/Open's portability guide. The biggest change this will make to the marketplace is that Sun will finally run Motif on its workstations as standard!

The common desktop environment - which incorporates aspects of HP's Visual User Environment (VUE), IBM's Common User Access model and Workplace Shell, OSF's Motif toolkit and Window Manager, SunSoft's OPEN LOOK and DeskSet productivity tools, and USL's SVR4.2 desktop manager components and scalable systems technologies (whew!) - was demonstrated running across five hardware and software platforms.

The six vendors involved will each sell, deliver and support OSF's DCE, SunSoft's ONC+ (NIS/NIS+, NFS, RPC), and Novell/Univel's NetWare UNIX client networking products.

The six companies plan to support a common core set of graphics facilities from the X Consortium - Xlib/X (for basic 2D graphics), Pexlib/PEX (for 2D/3D geometry graphics), and XIElib/XIE (for advanced imaging).

The six vendors will also submit a joint specification for the Interactive Multimedia Association's request for technology, work together to accelerate the development and delivery of object-based technology, and form a working group to facilitate the rationalisation and rapid acceptance of industry specifications in the systems management arena.

Start of personal comment:

These vendors are obviously very worried about Windows NT. Two players in the UNIX marketplace who are not a part of this announcement - Digital and Silicon Graphics - seem to be getting quite cosy with NT. This alliance is likely to polarise the marketplace into those vendors who support UNIX versus those who support NT.

The move is certainly a good one for UNIX - the standardisation that the marketplace has been crying out for. However, with UNIX systems becoming more of a commodity item, it remains to be seen how the individual vendors will maintain their product differentiation (and therefore their profit margins). We may yet see a shakeout in the marketplace.

It is interesting to speculate what may happen to this alliance if it succeeds in crippling NT. Will the vendors retreat back to their own individual camps, or will the marketplace be powerful enough to prevent such a reversal? Paradoxically, strong competition from NT seems to be the best hope for the future of UNIX.

Adrian Booth, Adrian Booth Computing Consultants <abcc@dialix.oz.au>, (09) 354 4936
From WAUG, the WA Chapter of AUUG

Sydney, April 1993 -- AUUG '93, now only five months away, is well under way to being best ever. April 6th was the closing date for paper submissions and the programme committee was well pleased with response. Papers from around the world were sent in, illustrating once again the draw and interest to the conference and exhibition and Australia in general.

Although very early in the process, organisers anticipate well over 500 delegates to the conference with the possibility of reaching 1,000 and well over 5,000 attendees to the exhibition. Agreements between the Australian Computer Society, ACS, and AUUG will allow the members of the ACS to attend AUUG '93 at the AUUG member price, a significant savings. AUUG '93, participants will, again, receive ACS PCP credits.

Tutorials --

Greg Rose, tutorial chair, informed AUUG he has received several new tutorials. "The 27th of September promises to be very educational and exciting," he said. "It appears AUUG '93 will have between 6 and 10 offerings for tutorials with several full day choices."

Conference Programme --

"We've got a really good programme in store for all attendees," said Programme Chair Piers Lauder. The conference will take a slightly different twist this year by offering 9 general sessions. These will consist of the typical keynotes, plenaries, and the new "footnote." "We have worked very diligently to ensure we have both informative and entertaining speakers," said Lauder.

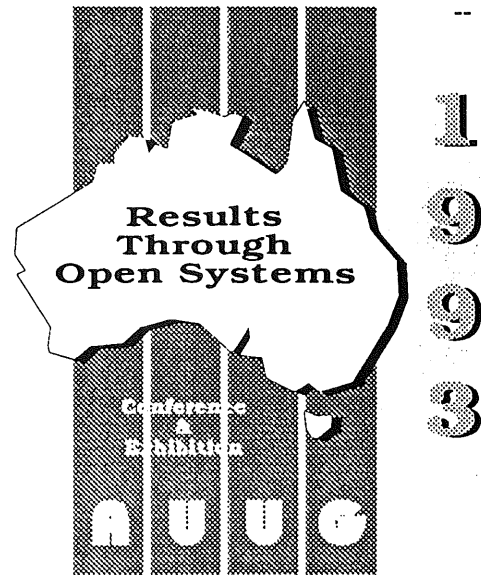
New Stream --

AUUG also informed us that a new stream has been added to the conference. **New Products**, is a stream specifically designed to promote products which are new or less than 6 months old in the Australasian market. Attendees may be required to sign a non-disclosure agreement when attending these sessions. However, we're certain it will be worth the small effort!

BOFs --

Last year in Melbourne, the introduction of Birds -Of -a-Feather (BOFs) proved to be

AUUGN



--
1
9
9
3

27 - 30 September 1993
Sydney Convention & Exhibition Centre
Darling Harbour, Australia

For conference and exhibition details contact: ACMS
+61 2 332-4622 tel. -- +61 2 332-4066 fax
whfoda@acms.auug.oz.au

extremely successful. Organisers request groups, companies, or anyone desiring to hold one of these sessions contact the Liz Fraumann, AUUG Business Manager, soon so schedules can be planned and facility space can be guaranteed.

Exhibition --

"AUUG '93 exhibition will be the largest display of open systems software ever seen in the marketplace," said Wael Foda of ACMS, exhibition organiser. With over 62% of space booked already companies wanting to participate in the exhibition should contact ACMS very soon, before there is no space left.

New this year, an AUUG Village concept will offer an end-user perspective. A replica of a small village with store fronts, businesses, and other community participants the AUUG Village promises to be one of the highlights of the exhibition. Organisers anticipate representatives from the banking, retail, health care, educational, and many other vertical markets to be represented. "This will allow attendees a first hand account from end-users and to hear how open systems have advanced their businesses," stated Foda.

Registration --

Watch your mail! An early bird registration offering will arrive by 5 May!
DON'T MISS THIS OPPORTUNITY TO SAVE!

Summer Conference Reviews

Following are two reviews of AUUG Summer Conferences. Hopefully, we will have reviews of the other summer conferences in the next issue of AUUGN.

Tasmanian Unix Summer Seminar

The fourth annual Unix Summer Seminar was held at the University on 11 February. More than 70 people attended the all-day seminar - a slightly better turnout than last year. The presentations were of high caliber and included 5 speakers from interstate. The Unix Summer Seminar continues to maintain its reputation as "value for money", and is now a well-established item in the annual ACS calendar of events.

One of the highlights of the Seminar was the "Security" afternoon session, featuring Bill Hart, Systems Administrator from CSIRO Marine Labs. Bill described what happened when the Marine Labs had to deal with an "Internet Intruder" (hacker) in December 1992. Bill and co-administrator, Peter Campbell, were able to find out a great deal of information about how the intruder worked. This information was passed on to the police, and was subsequently used to identify the intruder.

Bill's presentation was aptly followed by Roger Fraumann from Unix International speaking about Unix security in more general terms. Roger provided some very interesting statistics about the nature and frequency of computer crime, then went on to cover the major security issues which system administrators and organisations need to address. Roger predicted that within the next few years, pressure from insurance companies would force our organisations to place more emphasis on security -- to preserve the value of our information assets.

Other presentations included:

- AARNet Update, by Ray Jones, Uni.Tasmania
- Sun Solaris 2.0 by Dennis Letizia, Sun
- Multimedia, by Neville Scott, SGI
- RealTime Unix, David Triggs, HP
- Distributed Resource Management, by Darren Rushworth, Oracle
- Internet Information Services and Tools, Steven Bittinger and Steve Andrewartha, Uni.Tasmania

Each Seminar participant received a copy of the newly updated Unix User and Site Directory, a 32-page booklet providing details on more than 230 Unix users at 100 sites around Tasmania. The directory is an excellent resource for helping Unix users to stay in contact with each other, obtain assistance on difficult problems or take advantage of other site's experience with particular hardware or software configurations. Please contact Steven Bittinger to make corrections to the directory, or to obtain additional copies.

Steven.Bittinger@cc.utas.edu.au Phone: +61 02 20 2811
Information Technology Services (Hobart) Fax: +61 02 23 1772
University of Tasmania, Australia A'Link: AUST0221

Northern Territory Unix Summer Seminar

The second AUUG meeting in Darwin (AUUGWet 93) run for 2 days over the 18-19th February. Presentations were given on a wide variety of topics ranging from distributed database, networking, stereo graphic visualisation and geographic information systems. About 80 people attended the conference this year which is a substantial improvement over last year.

Phil Maker

pjm@cs.ntu.edu.au
089-4666666
School of I.T.,
N.T. University,
Darwin, Australia.

SESSPOOLE Committee AGM Report

Clunies Ross Conference Centre

Melbourne VIC, February 26, 1993

The Past...

Since the formation of SESSPOOLE in June 1989, it has existed with no real management structure. Despite the election (if you can call it that) of the current committee (John Carey and Stephen Prince) in August 1991, there has been no clear mandate as to who was the President, the Secretary and Treasurer. For this reason, this report is a joint one the by the current committee.

Since the SESSPOOLE epoch, we have managed to fulfill our initial aims of discussing open systems, and drinking wines, ales, or juices in a social gathering about eight times per year. Although I'm sad to report that the wine sub-committee seems to have fallen by the way-side in recent months.

Despite this informal structure and social gatherings, we are please to report that SESSPOOLE members, *per se*, have successfully organised AUUG summer meetings every year since February 1990, and at a profit I might add.

Out of the currently registered 143 AUUG members in Victoria, I'm sad to report that only about 30 have ever attended the social gatherings with about 12 of these being "die-hards".

As some of you may or may not be aware, AUUG recently published a set of Chapter Rules and Policy. These guidelines did not eventuate without some input from the current committee and members.

The Future...

After the publication of the Chapter Rules and Policy, we (the committee) felt†

† This really had nothing what-so-ever to do with the rumor that the AUUG committee was trying to force monies upon us. :-)

SESSPOOLE should really align itself with these guidelines, hence this the first AGM in SESSPOOLE history.

To meet this objective, we have put together a draft document on the SESSPOOLE Rules and Policy. This document is still open for comment by members or visitors to SESSPOOLE events. The document is really an attempt to formalise the duties of the committee members for the sake of holding future nominations and elections.

After informal discussions with members, another direction we are attempting to take is to change the structure of the meetings. It is felt that members would get more benefit if we alternated formal technical and social events. It is planned to trial this over the next twelve months, and if successfull, expand on it.

Finally, we're also attempting to resolve with the AUUG committee, a number of objections to the published and unpublished chapter rules/policy which we feel could disadvantage SESSPOOLE.

In summary, the future for SESSPOOLE is looking brighter, but we do need a lot more help from volunteers, especially a secretary and program chair.

John Carey, Stephen Prince
SESSPOOLE Management Committee

SESSPOOLE

AUUG Victorian Chapter

SESSPOOLE is the official Victorian chapter of AUUG Inc. It was the first Chapter of the AUUG to be formed, and its members have been involved in the staging of the Victorian AUUG Summer technical meetings every year since 1990. SESSPOOLE currently meets approximately every six weeks to hold alternate social and technical meetings. It is open to all members of AUUG Inc., and visitors who are interested in promoting further knowledge and understanding of UNIX and Open Systems within Victoria.

The purpose of the social meetings is to discuss UNIX and open systems, drinking wines and ales (or fruit juices if alcohol is not their thing), and generally relaxing and socialising over dinner. Whilst the technical meetings provide one or two "stand-up" talks relating to technical or commercial issues, or works in progress of open systems.

The program committee invites interested parties wishing to present their work, to submit informal proposals, ideas, or suggestions on any topics relating to Open Systems. We are interested in talks from both the commercial and research communities.

Social meetings are held in the Bistro of the *Oakleigh Hotel, 1555 Dandenong Road, Oakleigh*, starting at about 6:30pm. Venues for the technical meetings are varied and are announced prior to the event. The dates for the next few meetings are:

Thu, 29 April '93	Technical
Tue, 8 June '93	Social
Wed, 21 July '93	Technical
Thu, 2 September '93	Social
Tue, 12 October '93	Technical
Wed, 24 November '93	Technical
Thu, 16 December '93	Social
Tue, 24 January '94	Social
Wed, 1 March '94	Technical
Thu, 12 April '94	Social

Hope we'll see you there!

To find out more about SESSPOOLE and its activities, contact the committee or look for announcements in the newsgroup aus.auug, or on the mailing list sesspoole@clcs.com.au.

SESSPOOLE Committee			
President:	Stephen Prince Chancery Lane Computer Services Phone: (03) 608 0911 Email: sp@clcs.com.au	Secretary:	Neil Murray Webster Computer Corporation Phone: (03) 764 1100 Email: neil@wcc.oz.au
Treasurer:	John Carey Labtam Australia Phone: (03) 587 1444 Email: john@labtam.oz.au	Programme Chair:	Michael Paddon Iconix Phone: (03) 571 4244 Email: mwp@iconix.oz.au

WAUG and Perth News

Incredibly, some people seem to be confused about the nature of these columns. *Listen very carefully — I will say this only once.* Any opinions I express in my columns in AUUGN are mine, and must not be interpreted as “official” opinions of WAUG or its committee. My columns are not messages from WAUG — they are messages from me. Got that?

Good. Now, what was it I wanted to tell you?

I’m pleased to be able to report that WAUG *is* going to become a formal chapter of AUUG (and may already have done so by the time you read this). The Special General Meeting of WAUG on March 17 passed a motion resolving to do so. The motion was passed 36 to 18.

Adrian Booth was going to have given a short technical talk after the SGM, but the procedure and discussion took longer than expected, and the meeting voted to have the food now and the talk another time. Fortunately, the following week’s meeting (the March Technical Meeting) made up for this with an entertaining and informative talk about Desqview/X by Ian Gold from Quarterdeck. This was well-attended — the previous week’s formalities didn’t seem to have put people off.

At WAUG’s February Technical Meeting Malcolm Halsmith of Telecom gave an interesting talk on ISDN. No-one has yet reviewed this talk, unfortunately, and with everything else that’s been happening, I can’t remember enough details to do it justice. However I do recall that it was definitely a technical talk and not a sales pitch (sales pitches tend to be unwelcome at WAUG meetings). We heard a good deal about how ISDN actually works, how you connect to it, and how it compares with Telecom’s other data services.

To connect to ISDN you need to be within 3.5km of an exchange that is equipped with an ISDN multiplexer. The multiplexer connects to an ISDN exchange. If my memory is correct, there is an ISDN exchange in each capital city. An increasing number of exchanges in cities and large towns have ISDN multiplexers. The multiplexers are quite expensive; Telecom installs one when enough people in an area have asked for ISDN. In the early days of ISDN, the multiplexer software had various problems, which gave ISDN a shaky reputation; we were told these have now been fixed and ISDN is pretty reliable.

The lifeline of Western Australia’s Internauts,† the AARNet link between WA and the national hub, has recently been upgraded to a 2 megabit ISDN Megalink. It previously consisted of two 128 kilobit ISDN Microlinks. The difference is, well, noticeable.

Returning, more or less, to an earlier topic: WAUG becoming a chapter of AUUG means that WAUG members will join AUUG and receive AUUGN. This raises the question of what to do about WAUG’s existing newsletter, YAUN (Yet Another Unix Newsletter). As the editor of YAUN, I would like to see the reviews and articles that currently go in YAUN published in AUUGN instead, for the benefit of all AUUG members.

What’s more, each chapter could have a section in AUUGN, sub-edited by a chapter member who would be in a good position to motivate members of their chapter to contribute. If it wants to, each chapter could also produce its own newsletter containing local information.

Anyway, that’s what I’d like to see happen. Until things are sorted out, I’ll continue to edit YAUN (assuming WAUG don’t stop me) but will also submit to AUUGN’s editor any articles and reviews I think will interest the rest of the country. (I’ll ask for permission from the authors first, of course.)

By the time you read this, Perth’s big AUUG — and now WAUG — event of the year, the Summer Technical Conference, may already have happened. Adrian Booth has organised a programme that should offer something for most members. I don’t want to regurgitate the conference programme here, but almost all of the talks sound interesting to me, including Chris Schoettle on security, Greg Rose on Unix history, Paul Templeman on network backups, and Chris McDonald’s mysterious “How Does My Code Know When it is Running?”. That last title is a question one of Chris’s students asked him, believe it or not.

† Internet users. The word was coined by someone in the Internet Society (ISOC). Messages to the ISOC mailing list often begin with “Dear Internauts”. Dontcha just love it?

During the two days before the conference there will be tutorials, something new to the Perth conferences. Chris Schoettle is giving a full-day tutorial on technical aspects of System V Release 4; this is the one that was given at the 1992 Winter Conference. Greg Rose is giving a half-day tutorial on public domain prototyping tools (Tcl, Tk, and the like). I am giving a half-day one on perl for systems administrators, and I'd like to keep writing this column but I really must prepare some more slides.

FYI: WAUG's postal address is PO Box 877, WEST PERTH WA 6005. Email addresses: waug@uniwa.uwa.edu.au, waug-meetings@uniwa.uwa.edu.au, waug-newsletter@uniwa.uwa.edu.au.

Janet Jackson <janet@cs.uwa.edu.au>
From WAUG, the WA Chapter of AUUG

WAUG Meeting Review

March

DESQview/X

Ian Gold, Quarterdeck Office Systems

The March technical meeting of WAUG was addressed by Mr Ian Gold of Quarterdeck Office Systems on the subject of "DESQview/X", Quarterdeck's implementation of X windows for DOS systems.

Mr Gold commenced his talk by introducing X itself. By a show of hands he established that everyone had heard of X but a much smaller fraction knew what it was and what it would do for them.

He first stressed that X was not:

- X is not married to UNIX (though the first implementation of X was on UNIX systems and a very high proportion of X sites are based on UNIX systems);
- X is not married to TCP/IP (though most X systems do run on TCP/IP);
- X is not a GUI

Which leaves the question of what X is: X is an operating system independent, network protocol independent specification for network hosts (called X servers) which can provide screen, keyboard and mouse services to other network hosts (X clients).

Mr Gold then introduced the Quarterdeck range of products and positioned DESQview/X within that range. Quarterdeck's history has been in the creation of multitasking and memory management products for DOS, DESQview386 is their latest multitasker. DESQview/X is an extension of that product.

Mr Gold demonstrated DESQview/X acting as an X server and, more impressively, as a client. He said that there were three ways in which a DOS machine could act as an X client with DESQview/X:

- A program is included in DESQview which intercepts DOS character I/O calls and allows any text-based DOS program to run in a window on an X server;
- A screen driver for Microsoft Windows is included which enables the entire Microsoft Windows screen (which in turn contains several windows) to become a window on any X server on the network; and
- A developers' kit is available which enables the user to write or port native X applications to DOS.

Mr Gold's talk was very well received: as Glenn said at the close of the meeting "this is the first time I can recall that a speaker has been interrupted halfway through his talk by people wanting to know the price".

Major <major@nsaper.dialix.oz.au>
From WAUG, the WA Chapter of AUUG

Upcoming AUUG Canberra Chapter General Meetings

The dates for the next few meetings are:

Time & Date : 8:30pm Tuesday 11 May
Topic : **Windows NT**
Venue : TBA
Presenter : John Garde (U of C)

Time & Date : 8:30pm Tuesday 15 June
Topic : **Storage Technology at NRIC : a case study**
Venue : TBA
Presenter : Kim Malafant (NRIC)

For more information contact :

John Barlow
Tel : 06 249 2930
e-mail : John.Barlow@arp.anu.edu.au

Mathew Lim
Tel : 06 249 2750
e-mail : M.Lim@anu.edu.au

Greetings AUUG members,

I was delighted to be able to attend AUUG 92, which I thoroughly enjoyed. I would like to take this opportunity to invite you to attend UniForum NZ 93 and to give you a few details about our conference.

The conference commences with a Cocktail Party on Wednesday night and runs until Saturday midday, when we finish with a closing lunch. We are holding several half day tutorials on the Wednesday prior to the conference.

As well as keynote/plenary sessions we run two streams categorised as Management, Technical and Combined, with a third Pick stream on Friday only. There is a small sponsors exhibition attached to the conference featuring an interoperability display between many of the exhibitors.

The conference is a residential style conference and the registration fee includes all meals except breakfasts. Thursday night we are holding a Mexican Beer Festival (with hats) and Friday night is the Gala Conference Dinner.

AUUG members are entitled to register at the same rate as UniForum NZ members: NZ\$395 up to 16 April, and NZ\$495 after that date. With the exchange rate in your favour this gives you a really good deal.

Solway Park, Masterton, is a resort hotel about 2 hours travel from Wellington. The conference facilities are excellent and there are tennis courts, squash courts, swimming pools, golf course, walking and jogging tracks for the more energetic delegates. A complimentary shuttle bus is available from Wellington airport.

I look forward to seeing you there,

Julie Jones,
UniForum NZ President



UNIFORM NZ '93

NZ UNIX Systems User Group Inc.

10th Annual Conference

19-22 May 1993

Solway Park

Masterton

For registration details phone Julie Jones 64-25-958-245, or Ray Brownrigg 64-4-472-1000, or cut and mail the coupon below to UniForum NZ, P.O. Box 27-149, Mt Roskill, Auckland, New Zealand.

Yes I'd like to know more about UniForum NZ '93

NAME..... Phone.....

COMPANY.....

ADDRESS.....

Calendar of Events

1993

- Feb 22-24 Sun Open Sys. Expo, Chicago, IL
Mar 8 -12 Interop, Washington, D.C.
15-19 UniForum, San Francisco, CA
31-
Apr 4 * Applications Development Symposium - POSTPONED ^
19-21 * Mach III, Santa Fe, NM
19-21 * SANS II - Washington, DC
19-23 IEEE 1003
May 20-22 UniForum NZ, New Zealand
May 25-27 NeXTWORLD, San Francisco, CA
Jun 5-11 DECUS, Atlanta, GA
21-25 * USENIX, Cincinnati, OH
Jul 12-16 IEEE 1003
Aug 1 ACM Siggraph, Anaheim, CA
2 - 3 * Mobile & Location Independent Computing, Cambridge, MA
23-27 Interop, San Francisco, CA
" INET '93, San Francisco, CA
Sept 20-22 *Microkernels II, San Diego, CA
23-24 * SEDMS IV, San Diego, CA
Oct 4-6 * UNIX Security Symposium IV, Santa Clara, CA
18-22 IEEE 1003
Nov 1- 5 * LISA VII, Monterey, CA
Dec 4-10 DECUS, San Francisco, CA

1994

- Jan 17-21 * USENIX, San Francisco, CA
Mar 23-25 UniForum, San Francisco, CA
Spring * C++ Conference
May 7-13 DECUS, New Orleans, LA
Jun 6-10 * USENIX, Boston, MA
Sep 12-16 Interop, San Francisco, CA
Nov 12-18 DECUS, Anaheim, CA

1995

- Jan 16-20 * USENIX, New Orleans, LA
Feb 21-23 UniForum, Dallas, TX
May 13-19 DECUS, New Orleans, LA
Jun 19-22 * USENIX, San Francisco, CA
Nov 2- 8 DECUS, San Francisco, CA

1996

- Jan 22-26 * USENIX, San Diego, CA
Mar 12-14 UniForum, San Francisco, CA
May 18-24 DECUS, Orlando, FL
Nov 16-22 DECUS, Anaheim, CA

This is a combined calendar of planned conferences, symposia, and standards meetings related to the UNIX operating system. If you have a UNIX-related event that you wish to publicize, please contact login@usenix.org. Please provide your information in the same format as above.

* = events sponsored by the USENIX Association.

^ = has been postponed due to insufficient number of submissions received by the program committee.

ACM: Association for Computing Machinery
AUUG: Australian UNIX Users Group
DECUS: Digital Equipment Computer Users Society

EurOpen: European Forum for Open Systems
IEEE: Institute of Electrical and Electronics Engineers
IETF: Internet Engineering Task Force

INET: Internet Society
Interex: Intl Assoc. - Hewlett-Packard Comp. Users
JUS: Japan UNIX Society

LISA: USENIX Systems Administration Conference
SANS: Conf. on Tools & Techniques for System Admin., Networking & Security

SEDMS: Symposium on Experiences with Distributed and Multiprocessor Systems
UKUUG: United Kingdom UNIX Systems Users Group
UniForum: International Association of UNIX and Open Systems Professionals

† This is a re-print from *login*, the USENIX Association Newsletter, Volume 18 Number 1

--

Joint Announcement OF SAGE-AU and USENIX/SAGE, in Australia and USA

SYSTEM ADMINISTRATORS' GUILD OF AUSTRALIA FORMED

Melbourne, Australia

We are pleased to announce the formation of SAGE-AU, the System Administrators' Guild of Australia. SAGE-AU is a professional society for computer system administrators. SAGE-AU will promote interaction between professional systems administrators both in Australia and internationally, improve their levels of knowledge and professional excellence, and advance the status of systems administration as a profession. SAGE-AU is open to any system administrator supporting computer systems or networks.

SAGE-AU is modelled upon USENIX/SAGE. SAGE is an international organisation formed in early 1992 for similar purposes.

SAGE-AU has been formed with an interim management committee, consisting of Hal Miller (Commonwealth Scientific and Industrial Research Organisation, President), Peter Gray (University of Wollongong, Vice President), Frank Crawford (Australian Supercomputing Technology, Secretary), Greg Rose (Australian Computing and Communications Institute, Treasurer), Glenn Huxtable (University of Western Australia), and Keith Haberle (CSIRO). Elections will be held soon after the first SAGE Australia Conference, expected to be held in Melbourne in July.

Preliminary discussions with the USENIX organisation and the Board of Directors of SAGE, undertaken in January, make it possible to simultaneously announce the affiliation of SAGE-AU with SAGE. While details are not yet finalised, it is clear that both organisations expect to profit from the interchange of experience and ideas. Steve Simmons, newly elected President of USENIX/SAGE, said "We are very pleased with the attitude and enthusiasm of SAGE-AU. SAGE was originally formed in the United States but with the goal of representing system administrators internationally. SAGE-AU has, by their swift formation, made that goal much more attainable. Their representation of the Australian system administration community while affiliating with SAGE will help both organisations tremendously. We look forward to a long and fruitful relationship."

"The idea of a systems administrators guild seems to have really hit a worldwide need. We are delighted to be at the forefront of the international organisation process to fill that need, and look forward to having solid professional contact with our peers all over." said Hal Miller, interim President of SAGE-AU. There is already interest in forming SAGE/UK in the United Kingdom.

For further information, write to

Frank Crawford
Australian Supercomputer Technology
Woods Centre
PMB 1
Menai NSW 2234

or send electronic mail to sage-info@mel.dit.csiro.au.

SAGE-AU Information Sheet

The *Systems Administrators Guild of Australia* (SAGE-AU) has been formed with the following aims:

- a. to promote interaction between professional systems administrators within Australia and throughout the world;
- b. to advance the state of knowledge and level of excellence in the profession;
- c. to provide professional education for its members; and
- d. to provide recognition to high achievers amongst its members.

Members are accepted from people who are:

- a. currently employed as a computer systems administrator,
- b. previously employed as a computer systems administrator,
- c. learning to be a computer systems administrator, or
- d. in an otherwise closely related position (*e.g.* manager of such a group, developer of systems administration software, *etc.*).

Non-voting membership categories of *Associate Individual Member* and *Associate Institutional Member* are also available.

Current membership fees, for Foundation Individual Members, are:

Joining Fee	\$10
Annual Fee	\$20

Events in progress and proposed include:

- formation of a SAGE mailing list and/or newsgroup,
- affiliation with USENIX/SAGE,
- an annual LISA style conference, with the first to be held in Melbourne in July of this year,
- an AGM to be held in conjunction with the conference.

It is intended that most activities be conducted by e-mail, although special arrangements will be made for those without such access.

The interim Management Committee is:

President:	Hal Miller
Vice-President:	Peter Gray
Secretary:	Frank Crawford
Treasurer:	Greg Rose
General Committee:	Keith Haberle Glenn Huxtable

The SAGE-AU membership form is available by anonymous FTP from *ftp.mel.dit.csiro.au* in *pub/SAGE-AU/member.ps*. It is in PostScript format. If you are unable to retrieve the file from there, or you require more information, then contact:

Frank Crawford	Phone:	(02) 717 9404
Australian Supercomputing Technology	Fax:	(02) 717 9429
Woods Centre	E-mail:	frank@atom.ansto.gov.au
Private Mailbag 1		
Menai		
NSW 2234		

SAGE-AU Inaugural Conference and Annual General Meeting

Date: To be announced (hopefully July)

Venue: Melbourne (exact venue unknown)

Preliminary Announcement and Call for Papers

The System Administrators Guild of Australia (SAGE-AU) will be hosting a conference in conjunction with its inaugural annual general meeting. The theme of the conference will be

"Administering Networked Computers"

With the coming of age of computer networks, more and more organisations have internal networks of machines sharing information. Administration of these machines involves solving far more complex problems than for standalone computers. System administrators are under increasing pressure to allow more and more interconnection of machines without any loss of reliability or security.

SAGE-AU'93 solicits papers on all aspects of computer administration, particularly on the problems and solutions of administering networked computers.

Conference Details

SAGE-AU'93 will be a 3 day conference. The first day will be dedicated to tutorials on tools and techniques to aid system administration.

The inaugural AGM will be held at the end of the second day.

All other times will be allocated to presentations. A conference dinner will be held on the second night.

The conference will feature a small trade show focusing on system administration tools.

Tutorials

Tutorial sessions will be either half day or full day duration. People wishing to present tutorials should submit an abstract and a preference for a half day or full day slot to the address below. Tutorials should be run in a lecture format.

Papers

Slots are available for 15 minute, 30 minute and 60 minute presentations. 5 minutes should be reserved for questions from the audience.

15 minute slots are less formal and are present to allow people to talk briefly about some topic of interest, administration problem they are having or have solved without having to prepare a formal paper.

People presenting papers in the 30 and 60 minute slots will receive a 50% discount on registration fees.

If you wish to present a paper, send an abstract to the address below. Please indicate whether you wish to use a 15, 30 or 60 minute slot.

Abstracts should be approximately
100 - 200 words in length.

Papers should have a technical orientation and not contain advertising.

Deadlines

No deadlines have so far been established.

Addresses

Send all enquiries to regarding the conference to

Peter Gray Email: pdg@cs.uow.EDU.AU
Professional Officer
Dept of Computer Science
University of Wollongong Phone: +61 42 213770
N.S.W. 2500 Australia Fax: +61 42 213262

Requests for general information about SAGE-AU and membership applications should be addressed to

Frank Crawford
Australian Supercomputer Technology
Woods Centre
PMB 1
Menai NSW 2234

or emailed to sage-info@mel.dit.csiro.au.

Announcing the formation of the NSW chapter of SAGE-AU[†]

An informal meeting will be held on Thursday April 15 in room G92 of the Madsen building Sydney University to discuss formation of a NSW chapter of the Australian System Administrators Guild (SAGE-AU). The official announcement of the formation of SAGE-AU is included after this announcement.

All people interested in joining SAGE-AU or finding out more information are welcome to attend.

For more information please contact:

Peter Gray Internet: pdg@cs.uow.EDU.AU
Professional Officer UUCP: ...!munnar!cs.uow.EDU.AU!pdg
Dept of Computer Science MHSnet: pdg@cs.uow.oz.au
University of Wollongong Phone: +61 42 213770
N.S.W. 2500 Australia Fax : +61 42 213262

[†] Although this will be out of date by the time you receive AUUGN, it is included for information.

Open System Publications

As a service to members, AUUG will source Open System Publications from around the world. This includes various proceeding and other publications from such organisations as

AUUG, UniForum, USENIX, EurOpen, Sinix, *etc.*

For example:

EurOpen Proceedings		USENIX Proceedings	
Dublin	Autumn'83	C++ Conference	Apr'91
Munich	Spring'90	UNIX and Supercomputers Workshop	Sept'88
Trosno	Spring'90	Graphics Workshop IV	Oct'87

AUUG will provide these publications at cost (including freight), but with no handling charge. Delivery times will depend on method of freight which is at the discretion of AUUG and will be based on both freight times and cost.

To take advantage of this offer send, in writing, to the AUUG Secretariat, a list of the publications, making sure that you specify the organisation, an indication of the priority and the delivery address as well as the billing address (if different).

AUUG Inc.
Open System Publication Order
PO Box 366
Kensington, NSW, 2033
AUSTRALIA
Fax: (02) 332 4066

Following is a list of prices† provided by UniForum.

PUBLICATION ORDERS	Price		Postage/Handling		
	Member	Non-Member	Domestic	Canada	Overseas
CommUNIXations back issues*	\$3.95	\$5.00	\$3	\$5	\$5
UniForum Monthly back issues*	3.95	5.00	3	5	5
UniNews Newsletter subscription	30.00	60.00	8	11	30
1992 UniForum Products Directory	45.00	95.00	7	15	55
1992 UniForum Proceedings	20.00	25.00	4	5	11
Your Guide to POSIX	5.00	10.00	3	4	9
POSIX Explored: System Interface	5.00	10.00	3	4	9
Network Substrata	5.00	10.00	2	3	6
Network Applications	5.00	10.00	2	3	6
The UniForum Guide To					
Graphical User Interfaces	4.95	9.95	2	3	6
Electronic Mail De-Mystified	5.00	10.00	3	4	9
The UniForum Guide To					
Distributed Computing(*)	4.95	9.95	2	3	6

† Prices in US dollars
(*) please specify issues

ACSnet Survey

1.1 Introduction

ACSnet is a computer network linking many UNIX hosts in Australia. It provides connections over various media and is linked to AARNet, Internet, USENET, CSnet and many other overseas networks. Until the formation of AARNet it was the only such network available in Australia, and is still the only network of its type available to commercial sites within Australia. The software used for these connections is usually either SUN III or SUN IV (or MHSnet). For the purposes of this survey other software such as UUCP or SLIP is also relevant.

At the AUUG Annual General Meeting held in Melbourne on September 27th, 1990, the members requested that the AUUG Executive investigate ways of making connection to ACSnet easier, especially for sites currently without connections. This survey is aimed at clearly defining what is available and what is needed.

Replies are invited both from sites requiring connections and sites that are willing to accept connections from new sites. Any other site that has relevant information is also welcome to reply (e.g. a site looking at reducing its distance from the backbone).

Please send replies to:

Mail: Attn: Network Survey
AUUG Inc
P.O. Box 366
Kensington N.S.W. 2033

FAX: (02) 332 4066
E-Mail: auug@atom.lhrl.oz

Technical enquiries to:

Michael Paddon (mwp@iconix.oz.au) (03) 571 4244
or
Frank Crawford (frank@atom.lhrl.oz) (02) 717 9404

Thank you

=====

1.2 Contact Details

Name: _____
Address: _____

Phone: _____
Fax: _____
E-Mail: _____

1.3 Site Details

Host Name: _____
Hardware Type: _____
Operating System Version: _____
Location: _____

Existing Sites

If you are willing to accept a new network connection please complete the following section.

Please circle your choice (circle more than one if appropriate).

- B1. Type of software: SUNIII MHSnet UUCP
 TCP/IP SLIP
 Other (Please specify): _____
- B2. Type of connection: Direct Modem/Dialin Modem/Dialout
 X.25/Dialin X.25/Dialout
 Other (Please specify): _____
- B3. If **modem**, connection type: V21 (300 baud) V23 (1200/75) V22 (1200)
 V22bis (2400) V32 (9600) Trailblazer
 Other (Please specify): _____
- B4. Maximum traffic volume (in KB/day): < 1 1-10 10-100
 (not counting netnews) > 100: acceptable volume: _____
- B5. Will you supply a news feed? Yes No
 Limited (Please specify): _____
- B6. Any time restrictions on connection? Please specify: _____
- B7. If the connection requires STD charges (or Yes No
 equivalent) is this acceptable?
- B8. Do you charge for connection? Yes No
 If yes, approximately how much (please
 also specify units, e.g. \$X/MB or flat fee)? _____
- B9. Any other restrictions (e.g. educational
 connections only).?
- B10. Additional Comments:

Book Reviews

Unix User's Handbook^{††}

by Tim Parker
Microtrend

1993, 574 pages, \$US29.95 (Paperback)
ISBN: 0-915391-47-3

Reviewed by
Frank Crawford
Australian Supercomputing Technology
<frank@photon.ansto.gov.au>

This book bills itself as "*The One-Stop Reference for Unix Information*", and attempts to cover the following major versions:

- OSF/1,
- SCO Unix,
- SCO Xenix,
- System V Release 3.2,
- System V Release 4, and
- X/Open

It also relates other major versions back to these, including:

- Berkeley BSD 4.3 and SunOS (related to OSF/1), and
- Solaris (related to System V Release 4).

The book is really a collection of user reference manuals, listing most user commands with a note about which versions support the command. It goes through all the options, an explanation of what the command does, any restriction and an example of its use. Information is also given about differences across Unix versions. Specialist commands, such as those relating to administration or programming, are not included.

The first impression of this book is of the formatting, which I feel is rather unappealing. Once you get past this, the book's contents prove to be rather useful. It gives similarities and differences between systems and makes it obvious that the various Unix versions are more similar than they are different.

As with any such work covering a number of closely related features, there are some errors, for example, there is a claim that `cd` looks for a similar name if the one specified doesn't exist, in

fact, this only occurs for SCO Unix and Xenix. On the other hand, this book has also been useful, introducing commands that I had not previously been aware of.

On the whole, this book would be a useful addition to an experienced Unix user who is required to have knowledge, or to deal, with many different system. It is certainly not the only example of such a book and it isn't an outstanding example of them, but it does the job.

UNIX Power Tools[†]

by Jerry Peek, Tim O'Reilly, and Mike Loukides
O'Reilly and Associates/Bantam
March 1993, 1120 pages, \$US59.95
ISBN: 0-553-35402-7

Reviewed by
Ian Hoyle
BHP Research - Melbourne Laboratories
<ianh@resmel.bhp.com.au>

Well this book is so hot off the presses that the local O'Reilly distributor here in Oz (at least at the time of writing this review) doesn't yet have a price for it, let alone have it listed as being available yet!!

So what exactly is it? Hmmmm, perhaps the description from the book's cover is a good place to start:

"UNIX Power Tools contains literally thousands of tips, scripts, and techniques that make using UNIX easier, more effective, and even more fun. It also provides powerful public domain programs that add even more flexibility to the standard UNIX command set. Every script and program is available on disk, so you can add right to your own set of UNIX utilities"

The authors have collected together from USEnet, from the O'Reilly nutshell series and from various contributors which include such UNIX/USEnet luminaries as Larry Wall, Jonathon Kamens, Gene Spafford and Tom Christiansen, a rich compendium of essential UNIX tidbits in a rather unique format.

One of the first things I noticed is that the book is extensively cross-referenced. A more apt description would be hypertext on paper with

^{††} Please note the discount offered on page 34.

[†] Please note the special overseas offer on page 33.

cross-links going anywhere from other sections in the book to the software to be found on the CDROM (more on that later) that comes with UNIX Power Tools. To aid the reader, the first few pages after the table of contents is devoted to 'how to read this book'. This gave me a chuckle straight away - a picture of a screw is there to warn you if a section may contain material that could 'screw you' and a bomb warns of other possible dilemmas in using the books scripts (referred to as a cross-referenced screw :-)

The authors are at pains to point out that the material is meant to be browsed. There is no set order to be followed (hence the hypertext feel) and in fact the book seems to be more like an almanac or magazine with lots of factual, short articles that can lead you on to new knowledge as you follow your nose using the cross-referencing information.

My first impressions would include:

- big ... weighs in at ~1100 pages
- punchy, informative style.
- very thorough. It contains most of the UNIX tips & advice that I've heard of or read on USEnet over the years condensed into book form. Most of the sections in the 55(!) chapters are very short *i.e.* less than a page.

There are nine parts:

- Making yourself at home.
How to set up the shell(s), prompt, terminal, password etc.
- Let the Computer do the Dirty Work
Using the command line, aliases, history, job control & I/O
- Working with the Filesystem
Moving around, using find, file management
- Looking Inside Files
Regular expressions, comparing files, working with text.
- Text Editing
Using vi, emacs, sed, miscellaneous stuff
- Managing Processes
Starting, stopping, killing processes, timing, delayed execution
- Terminals and Printers
Setting serial lines, terminals, printing,
- Shell Programming
Programming for the uninitiated and initiated :-) script debugging, and "C Shell Programming ... NOT"

(that gave me a laugh :-)

- Miscellaneous
various stuff including what's on the CD & how to access/install it

— the CDROM has tons of stuff on it. eg PD utilities such as perl, gnu emacs, pbmplus, psutils, sc, ispell, sc etc etc (most of them being very up to date) as well as all of the scripts in the book.

ALSO, since the CD is in ISO 9660 format it can be used on lots of machines. There are precompiled binaries for SunOS 4.1.1, Ultrix 4.1, AIX 3.2, HP-UX 8.07, Xenix 2.3.2 & SCO Unix 3.2.x.

All up I'd call this book one of the `_must_haves_` for those of you that really want to stretch their UNIX knowledge that little bit further. With that said you'd all better start saving towards getting it

I luv it already

UNIX Power Tools

by Jerry Peek, Tim O'Reilly, and Mike Loukides
O'Reilly and Associates/Bantam Books
March 1993, 1120 pages, + CD-ROM
\$US59.95 (Paperback)
ISBN: 0-553-35402-7

Reviewed by
Frank Crawford
Australian Supercomputing Technology
<frank@photon.ansto.gov.au>

This has to be the most exciting book released for the UNIX market since the original Nutshell series. It is a book aimed at the advanced UNIX user, giving tips, techniques and explanations which make use of the many facilities available on UNIX systems.

Like the Nutshell books before it, this book makes use of the widespread knowledge available on Usenet. Although Jerry Peek, Tim O'Reilly and Mike Loukides are the listed authors, they in fact acknowledge 35 people by name and many other unknown contributors. Those named include Larry Wall, Randal Schwartz, Chris Torek, Jonathan Karmens and Tom Christiansen. Other articles are taken from the various Nutshell books.

Not only are the sources exceptional, the format is unusual, it is a *hypertext-on-paper* which is designed at making the information more accessible. The whole book is a collection of articles, all cross-referenced, where appropriate, and sometimes on overlapping topics, which gives useful tips. This is where the CD-ROM

also comes in, it supplies the source (and binaries for some common systems) to implement the suggestions given in the book. For example there are numerous *alias*'s given, and on the CD-ROM there is a file *csd_init* which includes them all. For more complicated examples, e.g. the terminal type identification program *qterm*, an explanation of the functionality is given and both the source and binaries are supplied.

Although this is a book for *power* UNIX users, there is plenty of information for novices, however, the best approach would be for a local *guru* to supply them with copies of relevant sections rather than being left to cope with it themselves. Even more importantly, the local *guru* will find this book to be invaluable. No matter what they know, there is always new information to be found, or even just to be refreshed, explanations about why it works, or hints about things they have overlooked.

The approach is like Usenet in a book, it provides answers to questions, supplies software to solve problems and explains how UNIX can be made to work for you. You are expected to have a reasonable knowledge of UNIX, but the book can then be used to increase your knowledge dramatically. Certainly this book is not without faults, but these are minor, and result from the book being a static entity. Most of the code seems to be a release or so old, some of the information may date, but in general, this won't greatly affect their usefulness.

UNIX Power Tools is essential for anyone using UNIX heavily or having to supply advice to others using UNIX systems. The information is extensive and appropriate to all levels, but more so for the advanced or *power* user. The CD-ROM is an added benefit, for those on the Internet it provides pointers to useful utilities, while for those without such access, it gives them easy access to source code they may not normally obtain.

Managing NFS and NIS

by Hal Stern,
O'Reilly and Associates, Inc.

Greg Rose
Australian Computing and Communications Institute
<ggr@koonda.acci.com.au>

I have only very recently become interested in NIS, and I already knew enough about NFS to survive, so I've really never read any more about either than was contained in a few *man* pages.

Since my experience with the Nutshell series has always been good, I had no hesitation in choosing Stern's *Managing NFS and NIS* to be my introduction to the big picture.

I was not disappointed. The book very concisely addressed more questions than I knew how to ask when I started. The writing style was clear and relatively concise, if a little bit dry. There are numerous examples, usually interrelated enough that the reader can maintain context between them. Unfortunately this makes some of the standalone examples a bit cryptic, but not distressingly so.

My major gripes with the book were only two:

- some of the command examples simply assumed that the reader was quite familiar with, and used exclusively, the C shell (except when displaying entries from */etc/rc* or the like, where the issue was ignored). I find this a particularly inappropriate assumption for the non-Sun market.
- there were sections which seemed wildly inappropriate to the subject at hand, such as debugging hardware problems in ethernet cabling and routers, and a whole appendix on transmission line theory.

The book first treats NIS, the Network Information Service. I think this is an advantage, since a lot of people might stop reading after NFS if it was treated first. I recently came to the stunning realisation that NIS actually solves a very real class of problems, albeit in a typically cryptic UNIX manner. This book helped me to discover just how well the problems are addressed.

The section on NFS was also very clear and easy to understand, and extended my own knowledge.

Additionally, there were chapters on Diskless Clients, which appeared mostly to address Sun workstations, Network Security, Centralising Mail Services, Diagnostic and Administrative Tools, Debugging Network Problems, Performance Analysis and Tuning, the Automounter, and PC/NFS.

When all of these are taken into account, Stern's book is an extremely useful addition to any network administrator's bookshelf. The title of the book is a bit of a misnomer; it really should be something like "Managing a Network Which Just Happens to Include NIS and NFS", but I can see why they didn't use that. Either way, I highly recommend it.



O'REILLY & ASSOCIATES, INC.
FOR IMMEDIATE RELEASE
February 26, 1993

Contact: Brian Erwin
O'Reilly & Associates
707/829-0515

Nancy Kaplan
Bantam Electronic Publishing
212/492-9545

**NEW BOOK WITH "SHRINK-WRAPPED" SOFTWARE
LETS YOU BECOME A UNIX POWER USER**
An O'Reilly & Associates, Inc./Bantam Book

"For UNIX, the biggest difference between a power user and a duffer is that a power user knows what he's doing and why he's doing it. Our goal is to help you become 'creative' about UNIX: to get you to the point where you can analyze your own problems and come up with your own solutions for them."

Mike Loukides, from **UNIX Power Tools**

SEBASTOPOL, CA — In the tradition of the classic bestseller, **DOS Power Tools**, **UNIX Power Tools** (March 1993/1,119 pages/one CD-ROM disk/\$59.95/ISBN 0-553-35402-7) is the definitive resource on the UNIX operating system for intermediate-to-advanced users. Complete with a CD-ROM disk containing the best public domain software available for UNIX, **UNIX Power Tools** offers a wealth of solutions-based tips, tricks, and concepts that will help users harness the power of the UNIX environment.

Written by Jerry Peek, Tim O'Reilly, and Mike Loukides, the authors and editors of the best-selling **Nutshell Handbooks on UNIX**, **UNIX Power Tools** is co-published by Bantam Computer Books and O'Reilly & Associates, the leading publisher of books covering UNIX.

"I think we have successfully combined our talents to produce a work that is unlike any other computer book that I've ever seen," said Ron Petruska, Bantam's editorial director and editor of **UNIX Power Tools**." This book will definitely be an instant classic of UNIX publishing."

Unlike most UNIX books, which are intended to be read cover-to-cover, **UNIX**
(MORE)

103 MORRIS STREET, SUITE A • SEBASTOPOL CA 95472 • (800)338-6887 • (707)829-0515 • FAX(707)829-0104



Power Tools is a browser's book, designed in an easy-to-follow two-color format that invites cross-referencing and helps users find the specific tools they need to become more efficient.

The book consists of 55 topically organized chapters, each of which contains short (less than two pages) articles culled from the experiences of some of the best-known authorities in the UNIX community. Every major aspect of the UNIX operating system is covered, including basic design, file management, managing devices, communications and networking, editing text, writing scripts, handling processes, and customizing the UNIX environment.

UNIX POWER TOOLS CD-ROM

The included CD-ROM contains not just the source code but pre-compiled binaries for all the best free software UNIX power users need to maximize productivity. Binaries are provided for Sun 3, Sun 4, SCO UNIX, SCO Xenix, IBM RS/6000, and DECstation platforms. The disk is in ISO-9660 format with Rock Ridge extensions, so it's mountable as a UNIX filesystem. Programs can be run right from the disk or installed on the hard disk. The programs are available on alternate media for an extra cost.

In choosing the software for this collection, the authors have stayed away from programs that are useful only to systems administrators or programmers, or programs that require a workstation window system to run. Included are:

- * **Perl**, an interpreted language that provides a superset of sed, awk, and shell programming, plus many unique features. Perl has become the preferred tool for many UNIX system administrators, but it's also wonderful for users.
- * **Gnu Emacs**, the most powerful text editor available for UNIX.
- * **pnmplus**, a collection of utilities for manipulating bitmap, color and grayscale images, and converting between image formats.
- * **sc**, a powerful spreadsheet program that runs on a ASCII terminal.
- * **ispell**, an interactive spelling checker that will make you wonder how you ever put up with the UNIX spell program.
- * **screen**, a utility that allows you to "detach" a login session so you can resume it from another terminal.
- * Every **shell, sed, and awk** script that's described in the book.
- * Many other useful programs, including a version of grep that finds matches that are only "approximately right," alternate shells like bash and tesh, enhanced GNU versions of programs like awk, tar, and find, as well as versions of many useful utilities that are found on some but not all UNIX systems, including compress, patch, and RCS (Revision Control System).

(MORE)

ABOUT THE AUTHORS

Jerry Peek is a user consultant and writer for O'Reilly & Associates. He has been a programmer, UNIX user consultant, course developer and trainer. His most recent book was *MH & xmh: E-Mail For Users and Programmers* (O'Reilly, 1992).

Tim O'Reilly is founder and president of O'Reilly & Associates, publisher of the X Window System Series and the popular *Nutshell Handbooks for UNIX and the Internet*. He has written or edited many technical books.

Mike Loukides is an editor with O'Reilly & Associates. He previously worked at Multiflow Computer, where he created all of Multiflow's documentation on programming languages.

O'REILLY & ASSOCIATES

O'Reilly & Associates is the leading publisher of books on "open systems," recognized worldwide for its definitive books on UNIX, the X Window System, and the Internet. Its editors are "computer people" who use the software they write about. The company's planning and review cycles link together authors, computer vendors, and technical experts throughout the industry, in a creative collaboration that mirrors the strengths of the "open systems" philosophy. Individuals who wish to purchase copies of the book directly, and corporations wishing to buy the book in bulk either for internal use or to redistribute it to their customers, may do so by calling 1-800-998-9938 or (707) 829-0515.

BANTAM ELECTRONIC PUBLISHING

Bantam Computer Books are published by Bantam Electronic Publishing, a division of Bantam Doubleday Dell Publishing Group, Inc. Founded in November 1983, Bantam Electronic Publishing is one of the fastest growing computer book publishers to emerge in recent years. Bantam Electronic Publishing is committed to the introduction of timely, authoritative computer books, written by industry experts in cooperation with leading hardware and software companies. When appropriate, the books include software packages related to the content of the book. Bantam Computer Books are available through major book and software chains, independent book stores, and software distributors. Books may also be ordered by calling 1-800-223-6834, Extension 9479 or (212) 492-9479 (in New York State).

#

* Products and names mentioned in this document are trademarks of their respective companies.

O'REILLY & ASSOCIATES, INC.

103 MORRIS STREET, SUITE A • SEBASTOPOL CA 95472 • (800)338-6887 • (707)829-0515 • FAX (707)829-0104



O'REILLY & ASSOCIATES, INC.

UNIX POWER TOOLS AVAILABLE DIRECT TO OVERSEAS CUSTOMERS

UNIX Power Tools

by Jerry Peek, Tim O'Reilly, and Mike Loukides

1,168 pages, ISBN 0-553-35402-7, \$59.95

(Special overseas price \$85.00 includes air courier delivery
to most countries for one copy of the book)

Includes CD-ROM

(Available exclusively to individual customers
direct from O'Reilly & Associates. Bookstore and distributor
accounts must contact Bantam books.)

Individuals and corporations, both domestically and internationally, can
order this book directly from O'Reilly & Associates, 7 a.m. to 5 p.m. Pacific Time. Phone
(707/829-0515 or 800/998-9938 in the U.S. or Canada), FAX (707/829-0104), e-mail
(order@ora.com) or write O'Reilly & Associates, 103 Morris St., Sebastopol, CA.,
95472, USA.

Please note that **UNIX Power Tools** is an exception to our normal domestic and inter-
national distribution agreements:

- * O'Reilly can accept international orders from individuals for
UNIX Power Tools ONLY and not for any other O'Reilly book.
- * Overseas orders for O'Reilly books other than **UNIX Power Tools**
will be returned unfilled.
- * All international orders must be prepaid by credit card, or with
a cheque in U.S. dollars on a U.S. bank and will be shipped by
air courier only.
- * O'Reilly cannot accept bookstore or book distributor orders for this
books; please contact Bantam Books. To find bookstores where **UNIX
Power Tools** is available, contact Bantam Books at 666 Fifth Avenue,
New York, NY 10103; or phone 212/493-9666.

#

103 MORRIS STREET, SUITE A • SEBASTOPOL CA 95472 • (800)998-9938 • (707)829-0515 • FAX (707)829-0104



Please accept with our compliments:

Unix User's Handbook

by Tim Parker

**Slawson Communications is pleased to announce
a special discount program for
Unix User Groups.**

All members are entitled to order single copies of *Unix User's Handbook* at a 20% discount off the \$29.95 retail price. That makes the single copy price only \$23.96.

Members need simply call our toll free order line:

1-800-752-9766

and reference our Unix Users Group Special. Orders may be charged to Visa or MasterCard. Shipping is via Air Printed Matter and charges vary by country.

For additional savings, we offer a special 40% discount when you purchase 5 or more copies to a single address. We only request prepayment, including freight (credit cards welcome). Your special group price is just \$17.97 per copy, a \$12 discount. A great way to give your members an extra discount or save your user group treasury.

Slawson Communications, Inc., 2075 Corte del Nogal, Suite E, Carlsbad, California 92009-1414
Telephone: 619/929-1979 Telefax: 619/929-1008

Wizard's Bookshelf

The whole world has enthusiastically embraced System V Release 4 - just ask any UNIX International vendor representative.

Well, as Mark Twain said, when you are in the majority it is time to reform. I'd like to see a new, modern implementation of UNIX, based on microkernel technology, and offering more functionality and a richer programming environment.

Some things I'd really like to see from a system administration perspective are:

- Virtual (logical) disks, so I can dynamically change partition sizes;
- ACLs, so I can fine-tune access to data;
- A "least privilege" security system, where I can allow a person or program the ability to, say, control printer queues, but no other administrative privileges.

I estimate that the provision of these three features would reduce my workload by 30% - 70%, depending upon the site. Just think how much more time I could spend on long-term planning and tuning, or reading network news!

I've read a little about the way SVR4 provides some of this functionality (or will Real Soon Now). Like most of System V, they appear to have taken several elegant, simple concepts and uglified (a.k.a. *productionised*) them into an incoherent unity†.

This, and an interest in Mach, got me interested in OSF/1 (which is Mach-based). Instead of reading the contentless OSF glossies, I bought a copy of *Guide to OSF/1*‡.

The book is based on an OSF technical overview, which means it essentially consists of a series of slides followed by a brief commentary. This made the presentation flow quickly, although in a somewhat jerky manner at times.

Anyone interested in microkernels and their use in building a UNIX-like system from one would find the first several chapters interesting, with topics such as threads, tasks, processes, messages, ports, and virtual memory.

Then come several chapters of interest to administrators - dynamic device configuration, filesystems, the logical volume manager, and security. (OSF/1 provides ACLs and "least privilege" at the C2 level, features which are not required by the "Orange Book" until the B3 and B2 levels, respectively.)

Programming under OSF/1 is then covered, including areas such as the OSF/1 programming environment, the run-time loader, programming with threads, and internationalisation.

The book concludes with a chapter on DCE, followed by several appendices containing various OSF white papers which cover remote procedure calls, directory services, security services, and filesystems in distributed computing environments, plus their (somewhat dated) comparison of OSF/1 and SVR4.

This book's technical content is quite high. It would serve as a general introduction to modern operating system concepts as they relate to UNIX, as well as an informative overview of OSF/1. While I never expect to administer many OSF/1-based systems, at least I can sigh and dream of what UNIX could have been.

Adrian Booth, Adrian Booth Computing Consultants <abcc@dialix.oz.au>, (09) 354 4936
From WAUG, the WA Chapter of AUUG

† An example is the STREAMS interface. Dennis Ritchie, who invented the original elegant concept, publicly refers to the productionised version as "sewers".

‡ *Guide to OSF/1: A Technical Synopsis*
O'Reilly & Associates, Inc.
First Edition, June 1991
ISBN: 0-937175-78-1

TMX



TMX puts the world in the palm of your hand.

The **M**essage **eX**change is a new service which offers real connectivity to news and services all around the world. Offered on Unix, PC and Mac platforms, TMX can connect you to hundreds of different sources including some of the most interesting and esoteric forums you will ever come across. All you need to start with is your PC and a modem.

Access to TMX

To connect to TMX, a variety of software is available to allow your PC, Unix based system, or Macintosh to access the network.

Clarinet

Connection to TMX also automatically gives you access to Clarinet, an electronic publishing network service that (for a small fee) provides professional news and information plus international computer industry news, technology-related wire stories and other major interest groups.

Forums

TMX gives you access to an incredible variety of special interest groups engaged in a wide selection of activities from company mergers and technology announcements, to using baroque instruments. You can participate, or simply listen in and observe in your chosen topic. The Message eXchange makes it simple and quick.

Electronic Mail

With TMX, You can send electronic mail to users of most electronic mail systems, on many different computer

systems such as Unix, MS-DOS, VMS or Macintosh. Through gateway services you can access other networks around the world including the Internet, UUNET, MCI Mail, ATT Mail, SprintMail, BITNET and many more.

File Transfer and Public Domain Access

A further service available via TMX is remote file transfer - that is the ability to retrieve programs or information from archive services. Huge numbers of public domain programs for Unix, PC, Macintosh and other systems are available from archives.

Special Interest Groups

Through TMX you can subscribe to special interest groups - thus enabling you to exchange news articles on various topics. These could range from cold fusion technology to the best restaurant in town.

Charges

Connections to TMX are based on an initial set up cost, plus ongoing connection fees. Depending on your requirements, various packages are available. A special discount is available to educational institutions and AUUG members.

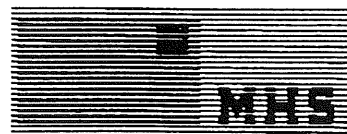
**For more information, please contact
Message Handling Systems Pty Ltd**

ACN 003 606 899

1st Floor, 2 King Street, Newtown NSW 2042 AUSTRALIA
Telephone: (02) 550 4448 (Sydney) (008) 806 962 (Interstate)

Facsimile: (02) 519 2551

E-mail address: enquiry@mhs.oz.au



MESSAGE HANDLING SYSTEMS PTY LIMITED

© Message Handling Systems, 1993. All rights reserved. All trademarks are acknowledged.

!AUUGN

26 Morris Place
Morris NJ 07940
18 August 1978

The following letter has been re-printed from AUUGN Volume 1 Number 1. We apologise for the production quality, but, there should be no problem understanding it.

Dear everybody,

Well I made it at last the promised land. However the plumbing is not all gold-plated and contrary to rumour there is not a PDP11 in every room. On my first day here I met a whole lot of people whom I am now in the process of re-meeting and getting their names straight. So many things have occurred in quick succession, it is hard to get them all into perspective or even a reasonable order.

First the next release of UNIX is planned for October. This will be an internal release for the Bell systems (over 300 of them) and they are in a position to call at least some of the tune since they are paying the piper. Whether a general outside release will follow almost immediately is not at all clear. (Incidentally it is likely that future academic UNIX license agreements will not contain such a strict interpretation of academic use as the PWB/UNIX agreement, since the latter is viewed as a special product with a different origin and market.)

The new release will be call UNIX/TS, to distinguish it from UNIX/RT (which is a new name for MERT). So in future there will be at least two flavours of UNIX. (PWB/UNIX will continue to be based on UNIX/TS). Many features of the new release are described fairly fully in the July-August issue of the Bell System Technical Journal (Vol. 57, No. 2, Part 2). A copy of this is being sent to each UNIX licensee so that a few copies will be around eventually. (I have already sent one copy to MWA.) This BSTJ constitutes a major addition to the available documentation, so that many of you may want private copies. They have apparently printed 5000 extra copies to start with. If you wish to order a batch I will try and help from this end.

The most obvious new feature is the new shell - elaborated along the lines already expected. The paper by Steve Bourne (p. 1971) is the place to look. Editor changes are relatively few, and a few ideas from outside could be usefully picked up. Some of the DMC features such as "!!" have not made it yet. The idea of changing directories within the editor has aroused some interest. The "wa" command used to be available as "W" but it will be replaced by a better idea which will go something like "w !cat >> filename". This is only a better idea when one realises that the editor file can be piped to any process started by an arbitrary shell command. There is a similar construct for "e" also.

On the subject of typesetting, the Computer centre here has three phototypesetters and is resisting pressure to acquire a fourth, for then they would have to hire another operator as well. Typeset copy is used regularly around here - for the weekly calendar of events, for address lists, - even for lecture notes. However there are bastions of conservatism: it is not used in the telephone directory (but a line printer listing is) and except for the most recent issue, not in the BSTJ. "troff" as it now stands is very closely wedded to the Graphics Systems phototypesetter - in particular only four fonts. I was talking to Brian Kernighan yesterday and he said that they were looking very seriously at a Morgenthaler CRT typesetter, with a much wider choice of fonts, point sizes and up to ten times as fast - its cost is now down to \$40,000 (compared to \$15,000 for the GSI dev-

ice). Once this new device is ordered, a new version of "nroff/troff" will have to be written. However I suppose it is another case of "don't hold your breath".

The UPM seems to have grown significantly. I counted 154 entries in Section 1 (section numbers are no longer in upper case Roman numerals) and the permuted KWIC index runs to 25 pages. New commands include:

filters for output to various terminals eg. Diablo
accounting routines
awk: perform operations on lines matching patterns (a rival for "sed")
bs: a compiler/interpreter for modest sized programs (replaces "bas")
cu: call Unix (cf "ll")
deroff: remove nroff etc. constructs
three versions of "diff"
three versions of "grep"
fsck: file system consistency check and interactive repair
"cut" and "paste"
shutdown
spell with a "-b" flag for Piers (and me); based on a dictionary with 25,000 words!

* or join
you spell
as to
chemist

A number of commands including "roff" seem to have disappeared for ever. New system calls include:

access: to files;
acct: turn accounting on or off;
chroot: change root directory on a per process basis (seems to be used for testing)
alarm & pause: replace "sleep" (new signal 14 SIGALRM)
syscall: indirect
umask: controls access permissions for newly created files on a per process basis
uname: returns name of current system version

There are a whole list of new routines in Section 3. All DHR's proposed I/O routines are now becoming standard and the old putchar, putc, printf, etc. are on the way out (vestiges remain). Access to the password file (still searched sequentially) is via "getpwent" etc. (Sixteen bit user ids are in, along with 16 bit group ids.) The remaining manual sections are not dramatically different. There is a new section 9 which documents the contents of a number of ".h" files (the number of which has grown dramatically).

Facilities here certainly are more lavish, eg. one office, two people, four phones (obviously one needs an extra phone for one's terminal). Up till now I have been using a tty 43 (this listing) which is pretty nice if you want hard copy but nothing fancy (apparently supply is having trouble keeping up with demand in the wide world). However being part of the telephone company tends to keep one if the fold as it were - everything is on a dial-up basis, limited to 1200 baud, and there is a noticeable preponderance of hard copy and lack of CRT terminals. Moreover there is no coffee room and people don't take coffee breaks. At lunch they tend to take the full hour and not to talk shop. On the other hand the whole building is air-conditioned which is highly appreciated right now as the weather is hot and very humid.

Widespread adoption of Unix throughout the Bell system may yet be the death of its reputation for solidity and reliability. There are now many, many people with their fingers in the pie and the new system is still in a state of flux. No one has yet called "enough" to changes and 'improvements'. There is a great long list of trouble reports from Bell installations of the kind which should be fixed and forgotten, eg.:

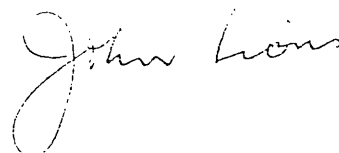
RMDIR can't remove a directory specified by a path name of more than 36 characters
SIZE cannot handle zero length files

A lot of problems are now arising because the new shell pays particular attention to the exit status returned by commands - apparently some commands have been doing it wrong for years. Regarding the resident code, I haven't been able to find out too much yet. Handling of text segments has been vastly improved to eliminate unnecessary swapping; they have picked up our suggested change for zombie processes; files can of course be bigger; method of getting more buffers without sacrificing segment five is being implemented; Keith Thompson says that up to 20% of cpu time may be going into "wakeup" in Level Six systems. The changes they have made to speed up the search for ready-to-run processes were definitely worthwhile. Some performance figures I have seen show "wakeup" still at the top of the list of most used procedures, but with only 8.9% of the time now.

The main new directions being pursued relate to portability. Bell doesn't want to be locked into one supplier for a variety of reasons. Quite a few references to the Interdata 8/32 appear in the UPM now. A VAX version of Unix is now running at Holmdel and performs 20% to 100% better, depending on the application. This is before any special advantage was taken of the VAX architecture, eg. for memory management. Code strings are about the same size, or a little less, than on the 11/70. so - if you really want to run large programs - the VAX now looks to be preferred. Further implementations on other equipment are certainly possible. In particular IBM has apparently looked very hard at Unix already. However there are more than a few legal problems inherent in that one, so again, don't hold your breath. More good news (?): the next release of UNIX will contain a Fortran 7 which bears comparison with Fortran IV+ for execution speeds. However the compiler is very large ... just makes it into "i" space.

Well that is about the lot for now. I've still got a lot of reading to do, and much material to digest. As a learning project, I am coding a modification to "login" and "passwd" to force people (where required by the administrator) to change their password at regular intervals - security is starting to become more important ever since somebody phoned up one of the systems, logged in as "ken" (which is an account on most systems) (without a password) wrote to somebody on the system late at night "Hi - it's me - what's the current root password" and got it! More to the point "ken" volunteered an interest to visit Australia .. possibly in conjunction with the next IFIP (i.e. world computer chess championship) so start saving your pennies, and putting together an official invitation ...

Cheers for now,



vpcheck
A Daemon to Balance Vector and Scalar Usage.

Frank Crawford

Aust. Supercomputing Tech., Private Mail Bag 1, Menai 2234
(frank@atom.ansto.gov.au)

1. Introduction

UNIX has long been recognised as the only operating system that runs on all classes of machines, from PC's to supercomputers. Although implementations are fairly standard across the range, there are different requirements for different types of systems, *e.g.* workstations are optimised to give good response to a GUI based interface and usually have only a single user, whereas large commercial systems need good I/O and efficient handling of a large number of users. Vector supercomputers have different requirements again, in general, they handle jobs that run for a long time and require large amounts of memory. Further, these jobs are a batch style job rather than interactive (who wants to wait five days to enter a number!).

Originally most supercomputers were operated in a batch mode, with a front-end used for general access, *e.g.* compilations, editing, viewing results, *etc.*. Recently there has been a trend to do away with the front-end and allow users direct access to the supercomputer. Because these systems have been optimised for long running batch jobs, the response for interactive jobs is often far less than expected, leading to users questioning the performance of such systems. In an effort to improve the interactive performance often some system tuning is done, but this has to be done in such a way that there is no negative impact on long running vector jobs.

2. Vector and Scalar Calculations

At this point it may be best to introduce how vector computers work and how they differ from other computers. Aside from conventional or scalar calculations, a vector computer has an additional *unit* which allows it to perform calculations on long lists, or *vectors*, of numbers with a significantly reduced overhead. Along with the *vector unit* there are additional vector instructions. For example, compare the number of instructions to process the loop (which is the main programming structure that vector compilers optimise) in Figures 1 and 2. The scalar loop must process ($7 \times 256 + 1 = 1793$) instructions, whereas the vectorised loop consists of only four instructions.

```
double x[256], y[256], z[256];
...
for (i = 0; i < 256; i++)
    x[i] = y[i] + z[i];
```

Figure 1. Simple Loop

	Scalar	Vector
	clr r1	vld vr1,y,#256
loop:	ld fr1,y[r1]	vld vr2,z,#256
	ld fr2,z[r1]	vad vr2,vr1,#256
	add fr2,fr1	vst vr2,x,#256
	st fr2,x[r1]	
	inc r1	
	cmp r1,#256	
	bne loop	

Figure 2. Assembler Code Generated

Unfortunately you cannot get something for nothing, and where you lose with a vector computer is that the vector instructions take much more time. In fact each instruction takes approximately $N \times \text{clock cycle}$ where N is the number of elements in the vector¹. Thus, for the loop in Figure 1 the *time* in scalar mode is ~ 1793

clock cycles, while in vector mode it is ~1024 clock cycles.

This is only one reason why vector computers are fast, other techniques such as instruction overlapping allow higher performance. For example, on the Fujitsu VP2200 there are two *load/store pipes* (see Figure 3), which means that both *vlds* can occur in parallel and, once data is available in the registers, the *vad* can commence. Unfortunately there is no way to run the *vst* in parallel as both *load/store pipes* are in use for the load².

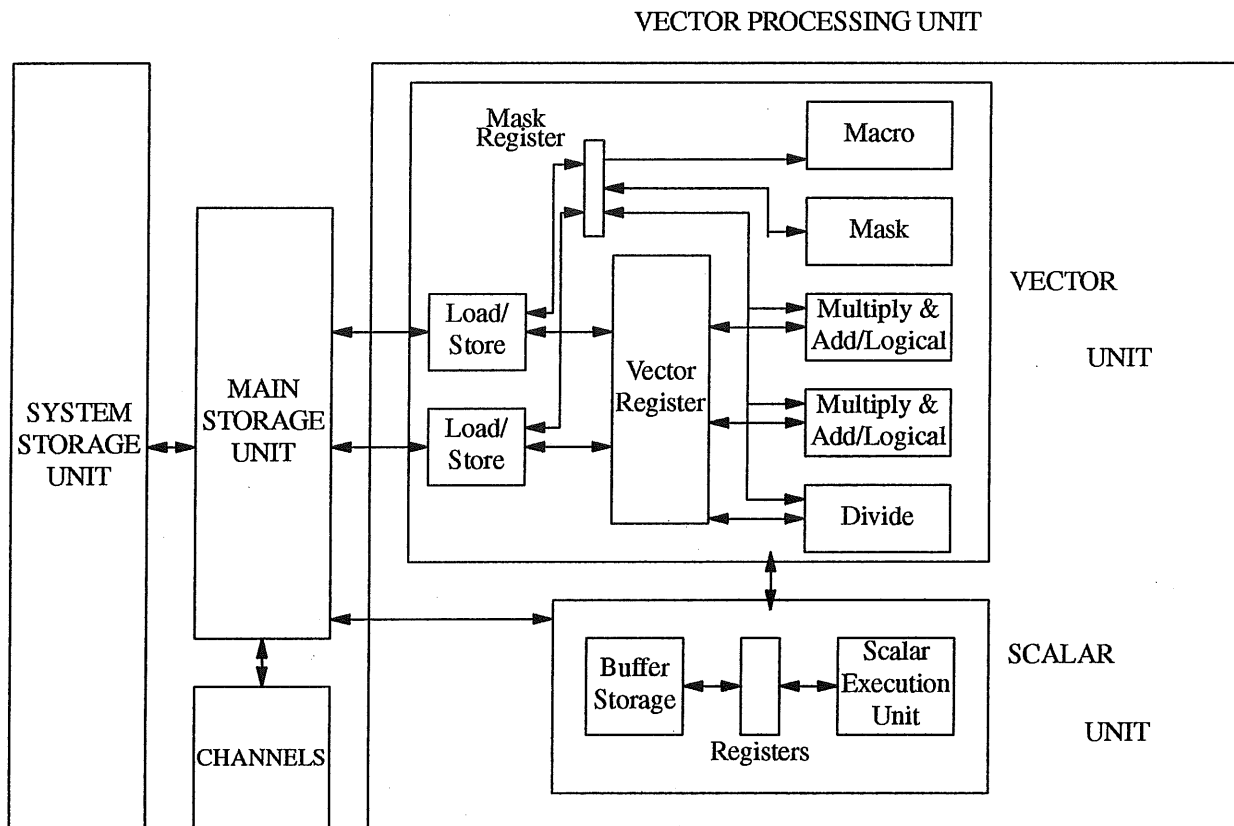


Figure 3. VP2000 Series Hardware Block Diagram.

This means that the loop can be processed in ~512 clock cycles, or about 30% of the scalar time. More complex calculations often make better use of the vector facilities and can achieve even higher performance.

One side-effect of this increased performance is that vector instructions have more context information associated with them and they exhibit much greater performance degradation from interrupts. For example, stopping the overlapping of the *vld* and the *vad* above will add an additional 256 clock cycles or ~50% to the time the loop takes.

3. System Tuning

One of the biggest causes of interrupts in a timesharing system is the system clock and the time slices given to individual processes. This has two effects on vector processes:

- i. Interrupts vector instructions and reduces performance, and
- ii. Causes the (larger) vector context information to be swapped.

1. This assumes long vectors and ignores startup times.

2. However a Cray has two load and one store pipes, so it can start a store in parallel as soon as results are available. The limitation of two *load/store pipes* on a VP2200 is not a drawback for more complex calculations.

These problems have been obvious for some time and the approach adopted by Fujitsu for UXP/M³ worked well in batch mode systems. The simple approach was to lengthen the time slice available to a vector process. A tunable parameter, **VPSLICE**, was added, which is used to multiply the normal time slice to give a longer value for vector processes. The recommended value for this is 150, *i.e.* a vector process gets a time slice 150 times longer than that for a scalar job⁴!

With the increasing use of such systems for interactive work and the explosion in the number of *system* processes (*e.g.* **nfsd**, **sendmail**, **inetd**, **lpsched**, **lpNet**, *etc*) such a scheme quickly runs into trouble. The first problem noticed is that the system becomes very slow to respond when vector jobs are running. What is worse is that compilations, one of the heaviest uses of the system by interactive users, take an excessive time, making even small changes to code difficult and ruining most users' development cycle. It is easy to see that the **VPSLICE** parameter acts like a multiplier of the load average for vector jobs, *i.e.* if **VPSLICE** is 20 and there are two vector jobs then the load to a scalar job appears to be $\sim 40!$ ⁵ A vector job with a negative priority can make the system appear to be dead.

There are also more subtle effects from extending the time slice, for example, most network activity depends on timers and large delays cause remote systems to time-out before a local daemon gets a chance to respond. This is most noticeable with **nfsd** and other *udp* based protocols.

Some experimentation has found that values of **VPSLICE** above 20 are unacceptable, while even values around five are very noticeable to users. Conversely, vector jobs suffer a marked performance degradation for values below 10 and some run up to 30% slower when **VPSLICE** is set to two. A small variation in performance can be noticed even for values of **VPSLICE** below 20. The results of these experiments precluded setting **VPSLICE** to a constant that suits both interactive users and vector jobs.

4. A Solution

One possible solution, adopted by Australian Supercomputing Technology (AST), is to set up a daemon to monitor the system load and automatically tune **VPSLICE** *accordingly*. One of the biggest problems with this approach is, what is "*accordingly*", *i.e.* what value do you select for **VPSLICE**.

4.1 *sysctl(VPSLICE)*

The first step in automating the procedure is to find some way to modify the current value of **VPSLICE**. Fortunately, UXP/M provides a function called *sysctl*, which, among other uses, can be used to get and to set **VPSLICE**. This function provides facilities to modify a number of UXP/M specific facilities, such as varying devices and CPU's online and offline.

Through the use of this function it is possible to query the system for the current value of **VPSLICE** or to set it to a new value. Although the value is an integer, it seems best to limit it to small values, *i.e.* < 256 , although even this leads to time slices of the order of one second for vector jobs.

4.2 */proc*

The second requirement is to obtain information about both scalar and vector processes. The */proc* file system provides one method to obtain this information. This *virtual* file system contains an entry for each process and through a number of *ioctl*s various statistics about each process can be obtained. In particular the most important information is to categorise a process as either scalar or vector.

One of the main reasons for the use of */proc* is that, under SVR4, a number of internal structures, including the process table, have been reorganised into lists. This makes it much more difficult to read the process table through */dev/kmem*, as there would be a number of separate memory references needed. The main drawback with the use of */proc* is the large number of system calls, such as *open*, *ioctl* and *close*, required for each process. As an example, under a BSD system only one *read* would be needed, versus one *readdir* and $N \times$ (*open*, *ioctl* and *close*), where N is the number of processes, required for SVR4.

3. UXP/M is Fujitsu's release of UNIX System V Release 4 (SVR4).

4. But with no change in how frequently it is scheduled to run!

5. This approximation only works for low values of **VPSLICE**.

4.3 *avenrun* vs *runque*

The final requirement is to obtain the current system load. For this there are two alternate mechanisms, the first is using the *avenrun* value. This is the *load average* which is reported by a number of BSD utilities, and gives the average number of runnable processes for the last one, five and fifteen minutes. Unfortunately, like most averages, it is very difficult to calculate the real number of processes available to run during the period.

The other system value that can be used is *runque*, which is incremented by the number of processes available to run, every clock tick. As this value is never reset it is possible to calculate how many processes were available to run during any period, just read the value at the start and end of the period.

Further, from observation, it appears that vector processes are always ready to run. This agrees with the fact that vector jobs should be predominately CPU bound and thus should only be interrupted by the system clock.

4.4 Algorithm

Given the above values, it is now possible to calculate a new value for **VPSLICE**. The prime desire for the new value is that **VPSLICE** should be small when there are a number of scalar jobs running. An additional desire is that the value increases rather quickly as the number of scalar jobs approaches zero.

After some experimentation the following formula was settled upon:

$$vp_slice_{new} = \left[\frac{1}{(\overline{runque} - nproc_{vec})} \right]^2$$

where:

vp_slice_{new} is the new value of **VPSLICE**,
 vp_slice_{cur} is the current value of **VPSLICE**,
 \overline{runque} is the average value of *runque* over the time, and
 $nproc_{vec}$ is the number of vector processes.

This formula also means that if there are no vector jobs and the total system load is low, then **VPSLICE** will drift to a high value, so that any new vector jobs will initially get a good run.

There are also a few limits that have to be considered, and these are:

$$vp_slice_{new} = \begin{cases} 2 \times vp_slice_{cur} & \text{if } \overline{runque} = nproc_{vec} \\ vp_slice_{cur} & \text{if } \overline{runque} < nproc_{vec} \end{cases}$$

If $\overline{runque} = nproc_{vec}$, then in effect there are no scalar jobs running, so this allows **VPSLICE** to quickly run up to large values. On the other hand $\overline{runque} < nproc_{vec}$ is most likely when a vector job begins or terminates during the period. Some account is taken of this within the sampling, but as it is assumed that vector jobs will run for long periods compared to the sampling interval, this is not a critical problem.

4.5 Restrictions

There are a number of restrictions placed on the value of **VPSLICE**. As discussed previously, if the value is greater than 20 there is a major impact on users, so if there are any users logged in, then the maximum value of **VPSLICE** is limited⁶ to 20. This is checked by reading */etc/utmp* for any users.

Much more of a problem is *nfsd* and related daemons. The only way to sample activity by these is to check for a change in CPU time for these processes. If any activity is seen from a selected list of programs then again **VPSLICE** is limited.

4.6 Sampling Frequency

As described above, some of the measurements can be expensive, and so cannot be run too frequently. On the other hand it cannot be run too infrequently or else users may still be inconvenienced by a system tuned

6. This is in fact an argument to the program, the default value is 20.

primarily for vector jobs. Again some experimentation has produced a value of five minutes as a suitable sampling time. This is not too long for a user to wait for a change in **VPSLICE**, and yet it does not appear to impact the system too greatly.

5. Impact on System

The program, *vpcheck*, has been in use by AST for nearly a year and during that time there have been very few comments on its use. Alternatively, there has also been little repetition of the original complaints that prompted its development.

While *vpcheck* is running it logs all changes to **VPSLICE**. These indicate that, outside normal working hours, **VPSLICE** is set to reasonable values for vector jobs.

6. Future Enhancements

Although *vpcheck* is running successfully, there are a number of possible enhancements that can be made. These fall into three different categories:

- performance enhancements,
- changes to the basic algorithm, and
- additional functions.

6.1 Performance Enhancements

There are a number of areas that can be improved within *vpcheck*, however the biggest area for improvement is in the reading of the process table. This could be improved by the use of *mmap* on */dev/kmem*, as has been done in such programs as *svr4mon*⁷. With this technique */dev/kmem* is mapped into the current processes' memory and all kernel memory references can be processed as a normal reference.

This technique can be used to replace the use of */proc* and give a decrease in the number of system calls required. In fact, this technique would cause changes to most of the code, as the traditional method of reading kernel values is using *lseek* and *read*, rather than direct memory access.

6.2 Changes to the Algorithm

The current algorithm has been developed from experiments and experience with the system, in the future a more suitable algorithm may be found. Two obvious changes are to make it dependent on the previous value of **VPSLICE** and also on the ratio of vector to scalar processes.

Another possibility is to base the minimum value of **VPSLICE** on the number of vector jobs. Currently the minimum value is one⁸, however it may be better to set it to the number of vector jobs.

6.3 Additional Functions

Aside from tuning **VPSLICE** there are a number of other possible functions that *vpcheck* can perform. All of these are separate to its current role, and include:

- set appropriate *nice* on vector processes, and
- repartition memory between vector and scalar processes⁹.

These functions are appropriate to add to *vpcheck* as they rely on similar analysis of vector jobs, particular the use of */proc*.

7. *svr4mon* is a program written by Andreas Vogel (av@ssw.de) to display system information about SVR4 systems.

8. Again an argument to the program which defaults to 1.

9. Under UXP/M, memory can be partitioned for either vector or scalar usage. This can be modified through *sysctl*, but with a number of restrictions on its use.

7. Conclusion

UNIX on supercomputers provides new challenges, which can in general be met with the use of existing facilities. As has been shown here, by the use of */proc* and using facilities provided in the kernel, all standard in SVR4, extensions to UNIX can be easily implemented.

Vpcheck has been in use for nearly one year and has proved to be beneficial to all users of the system. Despite having a simple approach to balancing the system usage between scalar and vector jobs, it has proved very effective. There have been very few comments about system response, whereas previously complaints were common. Alternatively, there have been relatively few comments about any negative impact on vector jobs, which is a good sign. Ultimately, however, this is a stopgap measure, as the real problems lie within the kernel and have to be addressed by the system designers, Fujitsu.



Softway
Excellence in System Software

Moving to OPEN Systems?

Softway is Australia's largest UNIX systems software house.

We understand the needs of the Open Systems marketplace, and can provide services in these areas:

- Client/Server architectures**
- TCP/IP based networks**
- Network integration**
- Benchmarking and performance tuning**
- UNIX Training**
- Contract software development**

For more information contact:
Dr Philip McCrea, Managing Director on
(02) 698 2322, or phil@sw.oz.au

79 Myrtle Street
Chippendale NSW 2008

PO Box 305
Strawberry Hills NSW 2012

386BSD: A Look Under The Hood.

Andrew McRae

Megadata Pty Ltd.
2/37 Waterloo Rd
North Ryde
andrew@mega.com.au

386BSD is a freely available port of the BSD Networking Release 2 software to a 386 PC architecture. It is a fully functional kernel with associated user programs and tools, albeit not yet a commercially stable release.

Initially the history and background of this system will be explored, then some machine specific aspects of 386BSD will be discussed, such as system startup, I/O configuration, and the virtual memory subsystem. Some discussion of various performance and porting issues will follow, and some conclusions drawn concerning the strengths and weaknesses of this particular UNIX port, and how well it fits onto the 386 and the PC hardware architecture.

Disclaimer: *This paper does not represent the views of AUUG or Megadata, it contains solely my own (sometimes tongue-in-cheek) opinions.*

Introduction.

1992 saw the release of a freely available BSD port to the ubiquitous 386 PC architecture, based on the BSD Networking Release 2 and ported in the most part by William Jollitz. Whilst not considered a commercially supported UNIX release, it is a fully functional BSD system, with networking and multi-user support included. It has provided access for many people to a working kernel with source, which has in the past been unobtainable except to sites that have licenced the official AT&T sources.

In the course of researching some hardware profiling techniques, I used 386BSD as a case study to examine the performance of a typical kernel. As a result of this study, I delved somewhat into the internal structure of the kernel. This paper describes some of these internals, and also attempts to highlight interesting areas where improvement may be obtained.

History and Background.

A number of operating systems have been released for free use, among them Linux, Minix etc. Most of these are aimed at conforming to POSIX functionality, so that they follow a standard Application Programmer Interface and User Interface. They generally differ from UNIX in that they tend to be written from the ground up having no access to AT&T source code. On the other hand, the BSD networking releases have been developed as part of kernels incorporating portions of the official AT&T source code of UNIX, and there has been a degree of sharing of code between the two organisations, such as the networking software, the BSD Fast File System etc (some people feel that there has been *too* much sharing of code).

Historically, there have been several releases of software from the Computer Science Research Group (CSRG) at the University of California at Berkeley. These distributions were called Berkeley Software Distribution (BSD) releases, and provided major building blocks for vendor kernels in the area of networking and file systems, and were a key element in the growth and prevalence of UNIX in the workstation and distributed computing environment. A number of major vendors based their kernels on BSD 4.2, which itself was based originally on the 32V UNIX† VAX version from AT&T; thus all complete releases of

† UNIX is a trademark of Bell Laboratories.

working BSD kernels required at least a 32V UNIX source licence. More recent versions since 4.2 BSD have reworked the networking areas and added new ports to different architectures (4.3 and 4.3Tahoe).

A side effect of this continuing development by the CSRG has been the replacement of the AT&T portions with freely available unencumbered software, so that more and more sections of the kernel may be released as part of BSD software releases. The first of these releases (of a limited distribution) was 4.3Reno. The goal was to eventually release unencumbered complete kernel source and utilities for a range of architectures. The VAX (and CCI) architecture ports were to be deprecated, and newer ports to Motorola 68K and Intel 386 architectures were to be integrated and supported. Whilst the CSRG were not directly involved with some of these ports (such as the Sparc port, done by Chris Torek at Lawrence Berkeley Laboratories), it was planned to release ports for commonly available hardware. The CSRG supported the 68K port (originally from the University of Utah) on the HP 300 architecture; William Jolitz provided a port to the Intel 386, based on the IBM PC architecture.

Aside from porting, new functionality was added in the form of a NFS implementation (from the University of Guelph), implementation of the lower layer OSI protocols (from the University of Wisconsin), and other ISO upper layers and applications based on the ISODE distributions.

The next planned major release was to be BSD 4.4, but in the meantime an interim release was generated that contained the work to date, termed the BSD Networking Release/2 (commonly referred to as NET/2). NET/2 contained significant functionality, but with some key modules missing. It was becoming clear that the role of CSRG was changing, and it was seen that the release of 4.4 BSD would also spell the demise of that group.

The release of NET/2 in 1991 sparked a number of events; since it was a system that was *almost* there, people saw an opportunity to use NET/2 as the basis for a real operating system that was free of licensing restrictions. A company known as Berkeley Software Design Inc. (BSDI) was formed with the goal of taking the NET/2 release and creating a commercially viable and supported system that could be sold with source code. The initial product used the Intel 386 port based on the ubiquitous IBM PC architecture. This product is known as BSD/386. Latterly this company has obtained the SPARC port code, so in the future it is likely that other popular architectures will be supported.

The Legal Situation.

The exciting prospect of finally achieving truly open and available systems (the 'Prague Spring') came to a jarring halt when tanks rolled in driven by men in dark suits carrying loaded legal injunctions, declared war on BSDI for infringement of trademark, then for copyright infringements; the embattled BSDI unexpectedly gained a co-combatant when Unix Software Laboratories (USL) extended the suit to the Regents of the University of California at Berkeley. The members of the CSRG spent considerable time writing legal briefs instead of operating system software, delaying the expected release date of 4.4BSD, though at the time it was not clear whether there was going to *be* a release of 4.4BSD.

A major premise of the law suit was that the unencumbered portions of the software was written by people who were exposed to copyrighted code, and therefore were (in the minds of the lawyers) 'mentally contaminated' by the code, and any subsequent code they produced should be influenced by copyright. Many people have argued that the only way programmers could be mentally contaminated is by writing BASIC programs, and that no-one would want to copy the sort of code that is contained within the USL distribution anyway, but it is unclear what the courts will make of this.

There has been much made of the legal and political ramifications caused by the various suits, and it is likely that whilst battles may be won or lost, the war may go on for quite a while.

Recently BSDI won an injunction preventing distribution of a first release of their product, so BSDI is now allowed to sell non-beta versions of BSD/386. Now that the genie is out of the bottle, it is difficult to see how he can be forced back in.

The Role of 386BSD.

386BSD is another example of a system based on the NET/2 release. William Jolitz (who did the initial 386 port) filled in the missing pieces to create a 386 based BSD system, and initially released it to the world as a freely distributable system in early 1992 as version 0.0. This allowed access to a working 'real'

kernel for the great unwashed masses (such as myself), so that development, testing and learning can take place (generally known as 'kernel hacking'). William Jolitz documented a large portion of this work in a series of articles in *Dr Dobbs Journal*, and is planning a book to be released later this year called **386BSD From The Inside Out**.

The primary and stated goal of 386BSD was to provide a *research* tool for people to experiment with new operating system ideas, or to examine the internals of a functional system to see how it runs.

After an early phase of instability due to the immature state of the system, a much more reliable release was available in July 1992 (version 0.1). This contained many contributions such as a PC filesystem, CDROM support, SCSI support etc.

There is an emerging view of 386BSD in that it is a social experiment to see if everybody 'on the net out there' can actually support, contribute and improve what is in effect freely available common property. Already there is a growing cadre of core users who have contributed in major ways such as supporting the X Window system, coordinating contributions and fixes, and improving the basic functionality by implementing new features such as shared libraries etc. Time will tell just how effective this will be.

My Interest in 386BSD.

Having developed a hardware profiling technique, I wished to use this to closely examine the performance aspects of a real kernel, and with the advent of 386BSD this became possible. In doing so, I discovered some interesting things about the internals of 386BSD and the PC architecture, and this paper's goal is to shine light into some of the dark corners. I will examine some of these areas, and then discuss the profiling results.

System Startup.

Booting 386BSD is a two stage process. The BIOS code on a PC attempts to load a boot loader by reading the first sector on a floppy or hard drive attached. This code initialises enough of the 386 memory descriptor tables to allow the loading of the secondary 386BSD boot loader, which is contained on the next 15 blocks (7.5 Kbytes).

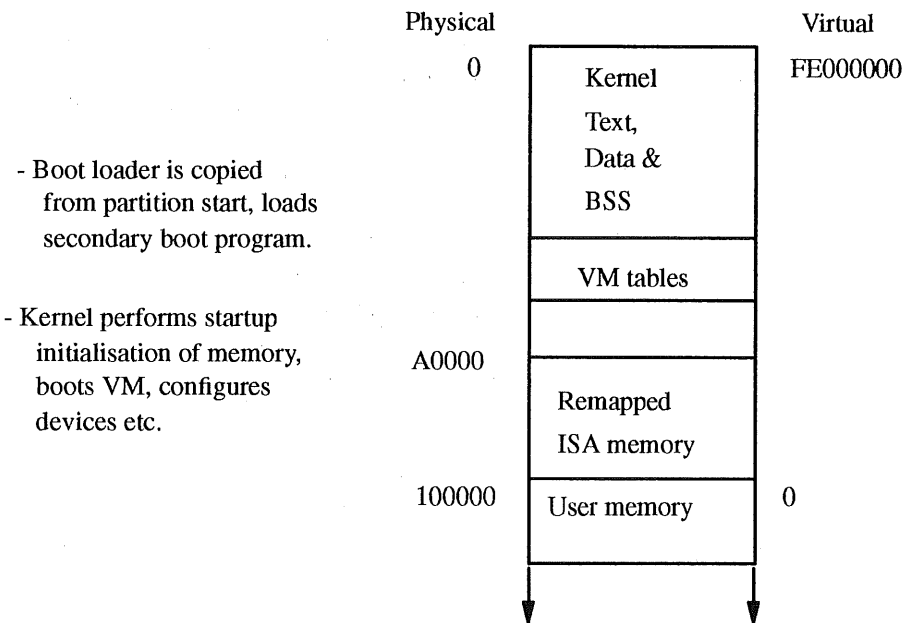


Figure 1 - Memory Arrangement

This secondary boot loader understands enough of the file system format to search for and read in the main kernel file, which is then stored in the lower 640 Kb section of the PC memory. After control is transferred to the 386bsd kernel, the physical memory addressing is remapped to new virtual locations as shown

in figure 1.

The default name searched for is *386bsd*, and the file is linked to run at an address of 0xFE000000. Currently the kernel is limited in size so that it fits into this lower portion of memory; future versions will remove this restriction.

In effect, the kernel is remapped to absolute location FE000000; the last location of the kernel is rounded to a page boundary, and a fixed number of pages are allocated for the kernel stack, a proto u-dot area and other virtual memory requirements. The ISA memory address space is then remapped to follow this kernel address space; the virtual address that this memory is mapped at may vary depending on the size of the kernel.

Once the VM and exception handling is bootstrapped, the system startup looks very much like other BSD based kernels, where configured devices are probed to determine the I/O configuration, and system initialisation takes place such as handcrafting the *init* process, and setting up the system page maps etc.

Virtual Memory on the 386.

The 386 has a modern paging virtual memory architecture that uses a two level page table scheme to map fixed size memory pages of 4 Kbytes each. Figure 2 shows the basic structure of this memory arrangement. In fact the 386 also allows extra segmentation descriptors to reference beyond the 32 bit address space, but quite rationally the 386BSD VM subsystem fixes the segment descriptors so that a flat 32 bit addressing mode is used.

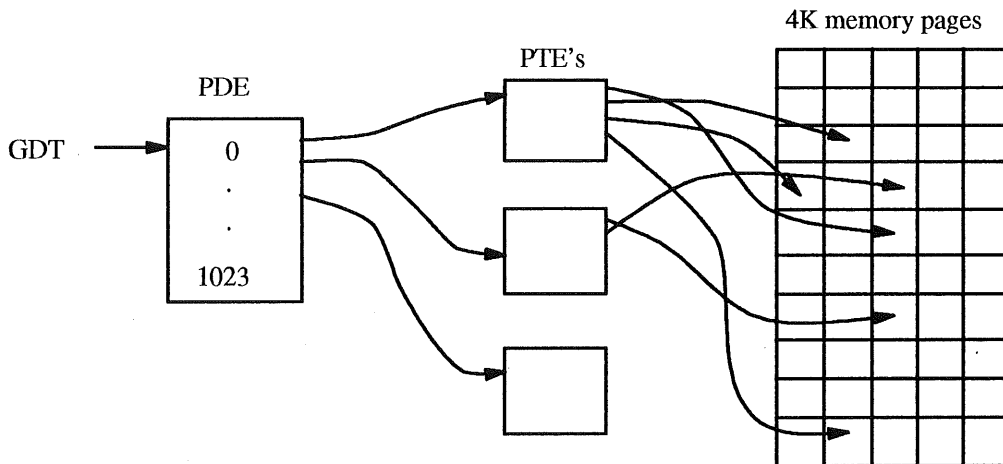


Figure 2 - 386 Virtual Memory

Allocated at the end of the kernel BSS space, the Page Directory Entry (PDE) table is a 4Kb page that has 1024 entries pointing to separate 4Kb Page Table Entry (PTE) pages, each of which holds up to 1024 Page Table Entries, each referencing one 4Kb page of memory. Each Page Table page itself is not required to be present in memory i.e it may be paged in as required, or not allocated to leave 'holes' in the VM address space. Thus the 32 bit flat address is divided into 10 bits of index into the Page Directory entries, the next 10 bits indexing the PTE within the page pointed to by the PDE; finally the remaining 12 bits reference the byte offset within the memory page referenced by the PTE.

To reduce the amount of information that must be obtained from memory to actually process a memory access through the VM subsystem, the 386 has a 32 entry Translation Lookaside Buffer (TLB) that caches PTE references.

Earlier BSD systems used a VM architecture based on the VAX virtual memory system; this was orientated around smaller and more expensive memory, and fast disc I/O. More modern systems are designed around larger memories and slower peripherals, so the Mach virtual memory subsystem (originating from Carnegie Mellon University) was grafted into the BSD kernel. This VM architecture centred around architecture independant page maps, where a system dependant portion would be implemented to map the desired actions onto the particular hardware VM architecture in use. This allowed sophisticated features to

--

be added such as copy-on-write, sharing of memory and mapping of files into memory, but without forsaking portability. The system dependant code was termed the *pmap* module for that architecture.

So to operate the Mach VM architecture a 386 specific *pmap* modules exists, which acts as the intermediary (when VM changes need to take place) between the standard VM code and the the physical manipulation of the 386 page tables. This module is a critical section of the 386BSD implementation, as it is a major part of the system dependant portion of the operating system. Since under a multi-tasking, multi-user paging operating system like 386BSD much of the processing of the system is taken up with the manipulation of the virtual address space, it is vital that this interface module be as fast and efficient as possible.

The kernel is linked to execute at the virtual address of (hex) FE000000, and the physical memory is mapped accordingly. User programs are linked to begin execution at virtual location 0, and memory allocated on a page by page basis as is required for the program. The linking of the kernel at high memory allows the kernel mode of each process to coexist within the same virtual address space as the user mode of the process, allowing direct access to the user data space by the kernel code. Transfer of data across the user/kernel boundary can then be simply a data move, allowing much higher transfer rates across the boundary, which in earlier systems was a major system bottleneck.

I/O Subsystem.

Currently only the ISA peripheral bus is supported; drivers exist for most commonly available boards, such as SCSI disc controllers, IDE controllers, a range of network cards, and serial cards. It is expected that a future release will incorporate support for EISA bus peripherals.

When the system is first initialised, the configured devices are probed to check their existence, and to allocate their interrupt vectors etc. Some devices are memory mapped in the so-called I/O memory physical address space from (hex) A0000 to FFFFF. When the kernel is initialized, this I/O memory is mapped to the end of the kernel space in virtual memory addressing, and the I/O memory address allocated at configuration time is adjusted accordingly.

One of the reasons why the PC architecture is so popular is because the hardware is plug compatible, and motherboard support for DMA, interrupts etc. has not changed since their introduction. This is a double edged sword; since the hardware bus architecture is the same as 10 years ago, drivers can be assured of compatibility, but that architecture has not scaled well with faster memories and processors. Compatibility is assured, but with devices that were once discrete ICs; whereas now nearly all functionality can be contained within a small number of fast VLSI chips, but these must still obey the slow timing rules of the original design.

One example of this is the interrupt structure. Originally the 8086 processor only allowed one interrupt line, and there was a single bit mask to disable this interrupt. Separate Interrupt Controller Units (ICUs) were used to provide multiple sources and vectoring, and the processor used I/O commands to program the ICUs with the desired interrupt masks. This has been carried through into the very latest processor designs, and the same interface is required to program the ICUs to allow/disallow different classes of interrupts. This adds overhead and complexity to the handling of driver interrupts.

Another example is the use of DMA over the ISA bus. Since the address bits used on the bus are limited, systems that have large amounts of memory cannot DMA into high memory; this requires the use of a *bounce buffer* in low memory, and CPU intervention to copy the results of the I/O into the desired final location.

Results of Profiling.

After profiling a number of the key areas of the kernel, some impressions emerged concerning the kernel performance. These fall into three main categories; CPU performance, I/O performance and virtual memory management. The platform was a 40 MHz 386 with 64 Kb cache and 8 Megabytes of main memory.

Firstly, I was pleasantly surprised to note the oft maligned Intel architecture did indeed run fast, especially at a clock speed at 40 Megahertz and employing 64 Kb of external cache. Moving data through the kernel to user space was faster than expected, and it was clear that function call and return was also speedy.

Undoubtedly memory speed and cache effects have a major impact on performance, as data throughput dropped markedly whenever memory was accessed on the ISA bus as opposed to main memory. More on this later. Profiling the interrupt code showed that the regular clock tick interrupt took on average 94 microseconds to execute; unfortunately the hardware architecture does not provide for Asynchronous System Traps (commonly known as software interrupts), so the interrupt code has to work extra hard to emulate this facility. The interrupt code overhead to do this is around 24 microseconds per interrupt; it is hard to judge whether this has a significant impact on system performance.

Due to the interrupt architecture of the bus and the processor, it was evident that more time was spent ensuring correct synchronisation and interrupt lockouts than would normally be required on a multi-priority interrupt level processor such as 680x0; on the average it took 11 microseconds per *splnet* call, which may not seem a long time, but the *spl** routines get called a great deal, and it all adds up to a significant amount. In one test, 9% of the total CPU time was spent in *splnet*, *splx*, *splhigh* and *spl0*. Unfortunately it is hard to see how this could be improved, given the nature of the interrupt architecture.

When some tests were performed where input/output activity was heavy, it was clear that a major bottleneck in system performance is the use of the ISA bus. This was especially noticeable on the Ethernet adaptor, which is a 8 bit wide controller. To transfer similar amounts of data, the ISA bus is up to 20 times slower than main memory transfers.

It would be instructive to profile different controller cards to determine where each performed best; when support for EISA cards is available it would be interesting to see what performance gain would be obtained using the higher bandwidth bus.

Whilst the CPU performs reasonably well, overall performance is crippled by the poor I/O bandwidth, and the interrupt architecture of the 386 and the ISA bus also contributes to reduced performance.

The virtual memory management subsystem of 386BSD was derived from the Mach memory management code. Following code path traces of various virtual memory functions indicates that the VM subsystem is definitely non-optimal. Some functions seem to run surprisingly fast; the routine that handles page faults and enables new pages to be accessed (*vm_fault*) takes about 400 microseconds, which seems reasonably low overhead. On the other hand, an excessive number of page faults seem to occur at times. Where the real performance problems lie is in creating new VM contexts for new processes, as explained in the next section.

Fork/exec Profiling.

A common operation of UNIX is to *fork* a process and create a child copy of the process, which then *execs* a new process image. For UNIX to perform well, these two operations must be reasonably fast, since some UNIX operations rely on a low cost of process creation; shell scripts for example rely on fast execution of processes to achieve a reasonable performance level. Due to the portable nature of shell scripts, it is becoming more and more common to employ shell scripts instead of compiled binary programs.

The current situation looks fairly abysmal; it takes some 24 milliseconds to perform a *vfork* operation, and it takes about 28 milliseconds to perform an *execve* system call. This adds to about 52 milliseconds to perform a combined fork/exec operation. Note that these times do not include any disk activity, as the process image was already cached. Where is this time being used? In figure 3 a summary of the highest cost subroutines is shown.

Most of the CPU time occurs within a small number of routines; it is clear that the *pmap* module is a bottleneck when manipulation of the virtual memory is required (the *bcopyb* call relates to scrolling of the console screen, so it should be ignored for the purpose of the exercise). Over 50% of the time is being spent in the virtual memory routines shown above. Examination of the code path trace shows that *pmap_pte* is called 1053 times when a *fork* is executed, and a similar amount when an *exec* is done. Further analysis of the code path shows the exact progress of the fork operation, and each subsection can be examined in detail to see the amount of time it is taking, and whether significant optimisation can take place. There is a major amount of cross-calling between the *pmap* module, and the rest of the virtual memory subsystem, so it is envisaged that a major performance benefit would occur if some of that glue could be trimmed back and some sculpting of the interface performed.

Elapsed	Net	# calls	(max/avg/min)	% real	% net	name
77603	58913	67	(14061/879/2)	5.02%	28.22%	pmap_remove
22283	22148	5549	(66/3/2)	1.89%	10.61%	pmap_pte
12938	12938	1215	(13/10/9)	1.10%	6.20%	splnet
10912	10874	3	(3634/3624/3613)	0.93%	5.21%	bcopyb
33435	10134	453	(40/22/21)	0.86%	4.85%	spl0
15963	7876	8	(3862/984/3)	0.67%	3.77%	pmap_protect
5657	5657	77	(244/73/3)	0.48%	2.71%	bcopy
47723	4889	115	(64/42/27)	0.42%	2.34%	vm_fault
4759	4759	1349	(5/3/3)	0.41%	2.28%	splx
7836	4361	236	(29/18/13)	0.37%	2.09%	vm_page_lookup
7320	3489	119	(39/29/12)	0.30%	1.67%	pmap_enter
3457	3457	38	(132/90/2)	0.29%	1.66%	bzero

Figure 3 - Fork/Exec Summary

Network Performance.

Profiling was performed on the TCP/IP and socket code by running a program that listened on a socket and when another host connected, read and discard the data. A Sun Sparcstation 2 was used as the host to send the data, as I was sure it could fill the available network bandwidth to the PC over an ethernet.

This was the only test that caused the PC to be totally CPU bound, so that essentially the CPU was busy 100% of the time. It was obvious that the PC could not process the data from the network at anywhere near Ethernet speed. Examining the code path trace and function summary showed that 33.6% of the time was spent in *bcopy*, and that 30.8% of the time was spent in *in_cksum*. Again, *splnet*, *splx* and *spl0* contributed around 9% of the time.

Delving further into the code path trace, it was clear that a major bottleneck occurs because the Ethernet driver for the card must copy that data from the onboard controller memory across the bus; each TCP data packet that was received (i.e a full Ethernet packet) took about 1045 microseconds to process at the driver level. This alone is only about 20% more data throughput than Ethernet itself, so it is unlikely that Ethernet data rates through to the network applications can be achieved using this 8 bit controller card, unless the rest of the software has been tuned for minimum overhead.

The other major CPU user was the checksum routine itself, which was almost a big an overhead as the driver packet copy. This was surprising at first, as the packet was now in main memory, and the checksumming should be close to memory-to-memory copying speeds. To checksum a 1 Kbyte packet was taking 843 microseconds. It was discovered that the *in_cksum* routine has not been optimally coded (e.g like other architectures where it is done in assembler), and recoding this routine should provide a reduction in packet processing from 2000 microseconds to perhaps 1200 microseconds; this would provide a major improvement in network performance, and the limiting factor would become the memory bandwidth available to the network controller across the ISA bus.

Another conclusion that can drawn is that a much faster I/O architecture is required before serious data throughput can be expected, but I think we all knew that.

Filesystems.

Separate profiling studies have been performed on the BSD Fast File System (FFS) code and the Network File System code. Due to the network performance problems discussed in the previous section, any performance issues in the actual NFS implementation are totally swamped by the I/O bandwidth limitations. An interesting situation arises due to the fact that UDP checksums are usually turned off with NFS; since the checksum routine contributed a large proportion to the CPU overhead, NFS actually provides less overhead and better throughput than an FTP style connection!

The disc controller used in the target PC was an IDE controller on a Seagate ST3144 disc. The FFS profiling showed how disc seek times impact the I/O throughput. Each read of the disc varied from 18 milliseconds up to 26 milliseconds. Each write interrupt took about 200 microseconds in total, with about 149

microseconds of that being actual transfer time of the data to the controller. Interrupts seemed to be close together most of the time (< 100 microseconds), so the disc driver may well be improved by waiting a short time after transferring the data to see if the controller is ready to accept another block straight away.

Overall, the CPU was only busy for 28% of the time when doing a large number of writes, so the disc seek times are still the major influence in determining disc throughput. It was interesting to see that out of that 28%, at least 6% was spent in the *spl** routines. It would be interesting to use a different type of controller (maybe one with DMA) and see what difference it makes.

Where Do You Get It?

There are a number of sites which contain the official distribution, as well as maintaining mirrors of the unofficial contributions, drivers etc. In Australia, kirk.bu.oz.au is an official mirror of the semi-official central repository of 386BSD software agate.berkeley.edu, run by Chris Demetriou (cgd@cs.berkeley.edu). Jordan Hubbard (jkh@whisker.lotus.ie) is the current co-ordinator of the 386BSD patchkit; patchkit is an automated method of applying bug fixes and patches, as well as the accumulated patches. It is highly recommended that these patches are applied.

The Usenix news groups **comp.os.386bsd.{apps, questions, development, bugs, misc}** cover the usual range of discussions, arguments, complaints and misunderstandings. Compared to the usual level of support that users get for operating systems that they pay for, the Net actually seems to provide fast and accurate results (some of the time, anyway).

It is expected the the official 0.2 version will be released sometime Real Soon Now; it is likely that this release will integrate the patches and bugs already reported, as well as providing new features such as EISA support and shared libraries.

Conclusions.

The major conclusion about the performance of 386BSD is that there are a small number of areas that need addressing, that when fixed should improve the performance considerably. The hardest area to address is the virtual memory subsystem. The easiest area would be the IP checksumming. The grossest area of mismatch between the hardware architecture and UNIX is the interrupt priority control and lack of software interrupts.

It was also clear that the hardware I/O performance is a major factor, and that the platform the profiling was performed on is crippled in I/O bandwidth.

How well does UNIX fit on a PC architecture? There are a number of areas where the fit is a bit rough; this is especially true in the I/O area, where the lack of a decent interrupt structure adds considerable overhead, and the poor performance of the peripheral bus limits the bandwidth of networking and mass storage data transfer. This begs the question, what is the difference between a (real) workstation and a PC? The answer seems to be I/O bandwidth and overhead, since the memory capacity and CPU performance of PCs seem to be reaching similar levels. The biggest strength of the PC architecture, that of hardware and binary compatibility, has also seen it chained to an architecture that is over a decade old, and is showing its age.

The future bodes well, however, for 386BSD. It is likely to provide a reasonable platform for teaching, research and experimentation for quite a while; at least until perhaps Plan 9 is freely available.

Load Sharing in a Distributed[†] Environment

Andy Bond

Department of Computer Science
Victoria University of Wellington

Email: Andy.Bond@comp.vuw.ac.nz

A network of UNIX [Ritchie, 1974] workstations is becoming a common sight in modern computing environments. Each workstation provides powerful computing resources which are periodically in strong demand by the local user. However even in busy environments, a significant proportion of these machines will be idle or underutilized at any one time. By supplementing the local computing resources through offloading tasks to idle workstations, better utilization can be made of the distributed computing environment. Such a strategy is commonly known as **load sharing** or **load balancing**. This paper discusses an experimental task allocation system called STARS. Using a distributed database of resource usage measurements, STARS allocates tasks to systems based on the availability of resources within the network. We will look at how such a system can be integrated into a users computing environment and some results based on our experience with its use.

Introduction

Today's research computing environments often comprise a variety of workstations designed to provide computing resources to their local user. These resources include CPU processing, IO bandwidth, memory, and network access. While this local computing capacity manages to fulfil the requirements of most users, there are users who require either more or less of their local computing capacity. For those on the lower end of the scale, their workstations can abound with idle computing resources which are going to waste. In contrast, there are those users who could quite easily soak up any extra computing resources if a simple means of access was available.

The interconnection technology that is now common place in the UNIX environment provides users with just such a simple means of using extra remote computing resources. Combined with a shared file system, the user is able to perform tasks remotely with little more effort than is required to compute locally. The allocation of spare computing resources to those users requiring more computing capacity is provided by a "load sharing" environment. By striving to balance the computing load on our network of workstations, we share the extra computing requirements of some users amongst the remaining workstations with spare capacity. However we must ensure that no local workstation user loses access to required local resources by a task allocation from some external workstation. This situation can lead to a load sharing policy being ostracized by the local computing community.

This paper looks at load sharing mechanisms and how they intergrate into a user community. Any such service should pose little over head on the general computing environment and make access to remote computing resources as transparent as possible. Initially, we shall look at some common traits to load sharing mechanisms including how such facilities are provided. The following sections will look at two implementations of load sharing mechanisms. CMU's Butler provides an off-the-shelf set of programs that allow users to gain to idle workstations. This is a good example of how a simple load sharing mechanism can be eased into a standard UNIX com-

[†] This paper was presented at UniForum NZ '92

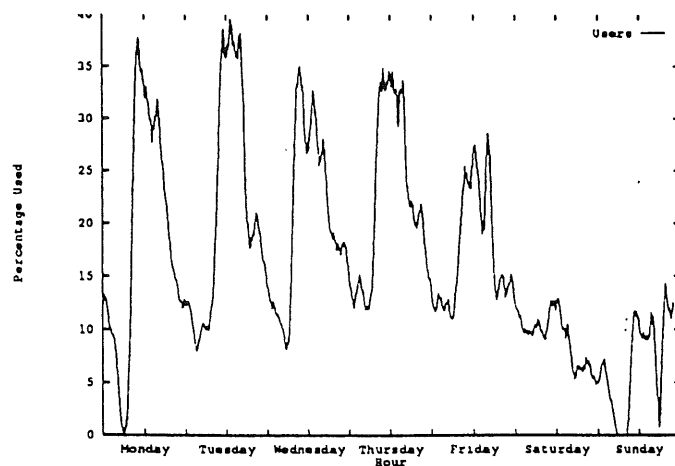
puting environment. Condor is the second example we will look at. It is somewhat more complex than Butler as it provides task mobility features that transparently move tasks amongst idle workstations until the task completes. This provides a useful comparison between a "strap on" task sharing paradigm and one that requires modification to the computing environment (i.e. the kernel).

The Load Sharing Paradigm

Load sharing in a UNIX environment seems to be something that should be there by default. As we have noted, the simple remote execution mechanism and the now common shared file system seem to beg the load sharing question. Why are such mechanisms still only provided in research environments? Probably because commercial organizations have not yet entered into workstation environments with force. However, the load sharing paradigm is applicable to any environment employing a number of UNIX hosts not just those with a one-to-one assignment of workstations to users.

There is extensive motivation for the use of load sharing mechanisms apart from simple implementation. In the Computer Science Department at Victoria University, we have seen that even at the busiest of times more than 50% of workstations are idle (see FIGURE 1 on page 2). As we would expect, the use of the computing environ-

FIGURE 1 Percentage of active workstations over an eight week period.

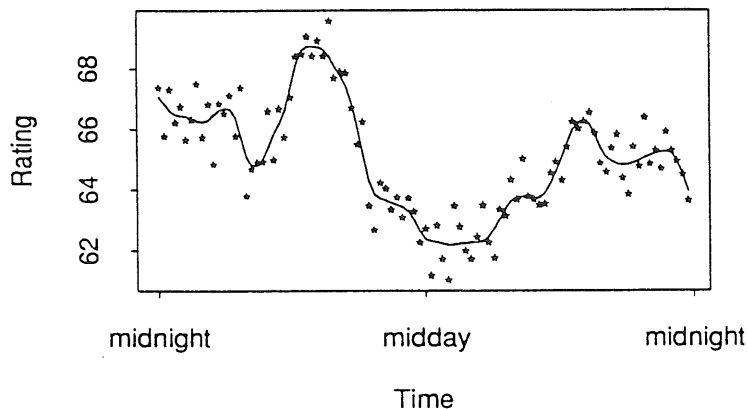


ment is busiest during the working hours with a steady drop off as night approaches. By ranking each host out of 100 (where 100 is no users with little load and 0 is many users with a high load), we can see from FIGURE 2 on page 3 the expected daily trend in workload. The load increases as the workday commences and slows down as the 5pm approaches¹. With such extensive idle computing resources available consistently over the day, it seems fateful that load sharing mechanisms will emerge from the woodwork.

The problem of optimally allocating a set of tasks to a set of hosts while minimizing response time is NP-complete². There are several successful allocation algorithms [Merkenscoff and Liaw, 1986] which approximate solutions to the problem but in real life we usually try and share load while disregarding the need for any strict minimization. Our solutions will attempt to reduce response time by automatically and transparently assigning tasks to processors. At the extremes, this can be as optimistic as assigning a task to a number of idle workstations or as pessimistic as being confronted with a set of very busy machines. Most allocation situations will occur

-
1. Various glitches in the curve indicate periods where coordinated system activity takes place (such as file system checks at 3am).
 2. An NP-complete problem is hard.

FIGURE 2 Average host ratings over the 24 hour weekday period for 6 weeks.



somewhere within these extremes but at all times we should ensure that our mechanism provides no worse response time than would be present without the intervention of load sharing.

Load sharing requires some criteria on which the allocation of tasks to hosts must be based. Some load sharing mechanisms are based on **static** allocation techniques, i.e. separate the tasks so all tasks of type α get sent to host β , etc. Essentially we are dividing the host set into a collection of task servers. A **dynamic** load sharing mechanism bases the allocation on some changing criteria using current system performance as a metric. Commonly, this metric turns out to be something like the load average. The next job to be allocated is sent to the host with the lowest load average. In addition, we can introduce another dimension into the equation by using a different allocation policy for different types of jobs. A system that changes the load sharing policy may be thought of as adaptive.

Sometimes the allocation of a task to a host turns out to be less than desirable. The system load may increase due to some unforeseen local task or the allocation mechanism may make a misinformed allocation choice. An allocation mechanism may have the ability to pre-empt a task and migrate it to a more suitable site. Instead of load sharing, we now have an optimal method for load balancing at the extra cost of task migration. This usually requires modifications to the UNIX kernel or the application to support such moves [Freedman, 1991]. Without a migration mechanism, the load sharing system may choose to terminate any offending allocated tasks and restart them elsewhere or simply ignore the situation and hope it goes away. The system implementations described later include examples of each of these choices.

A dynamic load sharing method needs access to some sort of performance data from all the hosts that are potential remote execution sites. By simple extension, each host cooperating in the load sharing paradigm must both have access to load information from each other host as well as provide load data to these hosts. As is common in most distributed systems, this information may be provided by a central server or may be distributed amongst all contributing hosts. If such servers are also responsible for allocation decisions, a central server approach will ensure that global allocation decisions are available at minimal cost (i.e. no messages need passed between cooperating hosts). However should the central server be unavailable, no load sharing is possible. In contrast, a distributed approach benefits from increased reliability as no single node failure will restrict load sharing at anywhere other than the point of failure. This reliability comes at an increased cost of communication between the cooperating server sites.

Load sharing mechanisms can come in many forms. The decision criteria or policy for making the load sharing decisions can range from static table driven to dynamic with adaptive task dependent decision criteria. By intro-

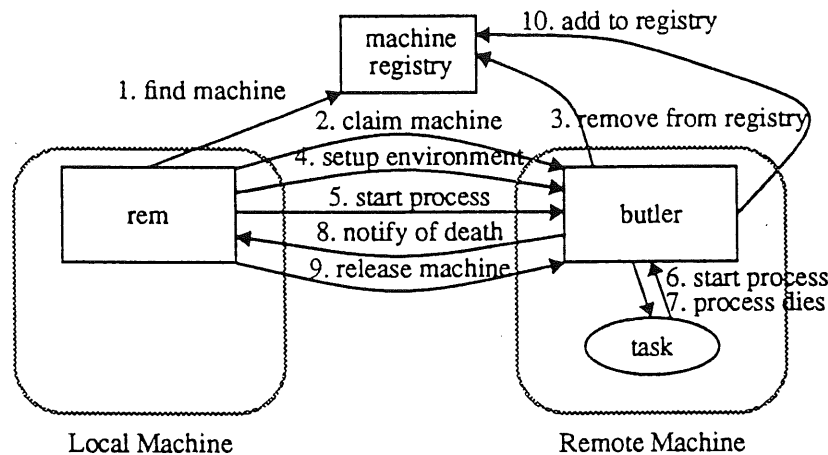
ducing a pre-emption and migration scheme paradigm we are able to turn our load sharing tool into a optimal load balancing mechanism. However this usually requires reduced integration flexibility (through kernel or application modifications) and migration costs. The reliability of any distributed mechanism is only as good as it's weakest link. A distributed server approach decreases the failure probability by increasing the required points of failure for total system loss. In contrast, the more traditional centralized server approach reduces communication overhead but at the same time decreases total load sharing reliability.

A Butler for Idle Workstations

Butler [Nichols, 1987] is a system for using idle workstations that was developed at CMU. It requires some form of networked file system (such as NFS [Lyon et al., 1985]) and a windowing environment (like X Windows [Scheifler and Gettys, 1986]). The system is completely detached from the kernel and runs off-the-shelf with neither kernel nor application modifications. Butler is only used for the allocation of idle machines and does not interest itself in spare capacity on underutilized workstations.

FIGURE 3 on page 4 shows a diagram of the three mechanisms making up the Butler system and how they inter-

FIGURE 3 Process invocation using Butler.



act in the invocation of a process.

A pool of idle hosts is maintained in a global machine registry. By running a Butler server on a workstation, the workstation is added or removed from this machine registry as remote jobs are executed on the host. Originally, the machine registry was simply implemented as a shared directory where the Butler servers added or removed file entries from the directory. Later as this became a bottleneck (with up to 350 workstations accessing it), the machine registry was cached in registry servers that migrated amongst idle hosts (gypsy servers).

To invoke a process using Butler, a local Butler client (`rem`) is run requesting an idle host. This client contacts a machine registry server and then attempts to contact the Butler server on the provided host. If the idle host has already been allocated or the machine has been reclaimed by it's local user then the client tries for another idle host from the machine registry server. Upon successful allocation, the remote Butler removes itself from the machine registry and exchanges execution information with the client. If the local user returns while the host executes the requested process, the client is informed of the process's impending termination and a short time later the process is terminated. No attempt is made at either restarting nor migrating the process. Following pre-emption and the user once again logging out, a Butler server is run on the workstation. Initially, these servers were manually started by the departing user but were then automatically invoked when the user count reached zero.

Butler is very useful for implementing gypsy servers. These servers roam idle workstations performing their assigned task (such as machine registry or help servers) while only using spare capacity within the distributed environment. Such mobile servers are located via a shared directory structure as was the early implementation of the machine registry.

The average start-up time for remote executions using the Butler system is seven seconds. It was found to be long enough to discourage people from using the mechanism for small tasks but was popular for long interactive tasks (such as remote shells). Butler is a good example of a simple, non-intrusive load sharing system which provides an elegant solution to remote execution in a distributed UNIX environment. The scheduler uses simple dynamic allocation information (a boolean host idle metric) as the load sharing policy and has sidetracked the pre-emption/migration option by a warn and kill approach. By using Butler servers on each host, the only point of failure for the entire load sharing system is the machine registry server set. The machine registry servers increase the reliability of this point of failure by duplicating themselves. Each machine registry server monitors other servers, starting another when the number below a specified minimum.

The Condor Scheduling System

Condor [Litzkow et al., 1988] was developed at the University of Wisconsin-Madison on top of a specialized remote execution mechanism¹ for UNIX. The system is designed to maximize host utilization while minimizing the interference between the workstation users and the allocated tasks. By incorporating a pre-emption and migration mechanism, interference to returning workstation users is minimized. Condor is essentially an elaborate background mechanism which supports execution on remote hosts. All these features come at a minimal overhead with less than 1% of the local CPU used.

The underutilization of workstation resources and abundance of large computing tasks led to the development of Condor. The long idle periods that characterize workstation processing cycles make the workstations an ideal environment for the scheduling of long running jobs. By guaranteeing the sanctity of the local user, a checkpointed remote execution system can ensure that pre-empted tasks will always run to completion. The Condor subsystem has solved several issues associated with load distribution:

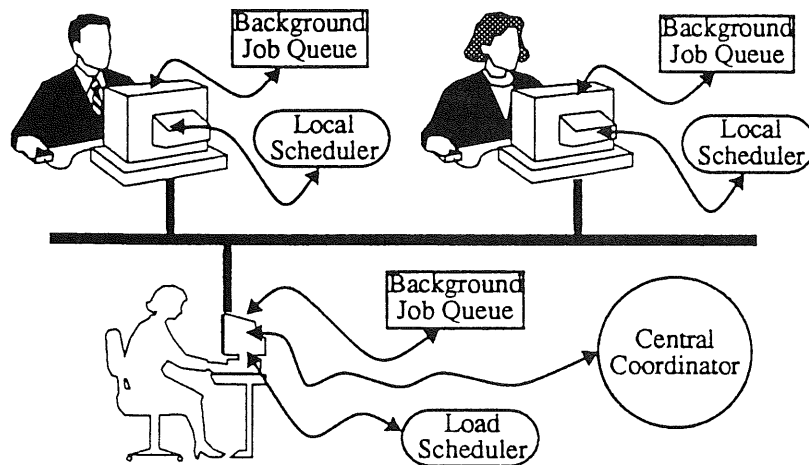
- task placement is transparent
- on remote site failure, a task is automatically restarted with no user interaction
- there is fair and equal access to available idle remote sites
- scheduling overhead is minimal, ensuring wide user acceptance

FIGURE 4 on page 6 shows how task scheduling is affected from within Condor. Each workstation maintains a local job scheduler and background job list. In addition, one site maintains a central coordinator. A user submits background jobs to their local job queue which is in turn periodically polled by the central coordinator. The central coordinator collects these background jobs and also the site states from the local schedulers. A site state is determined by the local scheduler and is set according to the availability of the site for remote processing. The central coordinator then allocates the collected jobs amongst the available local schedulers. While a job is running, the local scheduler monitors whether the local user has returned and if so, pre-empted the executing job. The pre-empted task is then placed in the background job queue again, ready to be picked up by the central coordinator on the next poll. The job will then be restarted at another available site.

The Condor dynamic load sharing system uses an idle metric that is provided by the distributed local schedulers. The system relies on an underlying checkpointed remote execution system to provide migration. Once the local user returns to an idle workstation, any executing allocated tasks are pre-empted and migrated to another idle host. This underlying remote execution relies upon kernel and application changes to provide the checkpointing. Even though the local schedulers are fully distributed, the central coordinator is required to make the final allocation.

1. Remote Unix (RU) has a checkpointing feature which allows a program to recover its state and be restarted on another machine. As was mentioned in the migration discussion, this feature requires extensive system support.

FIGURE 4 Scheduling within Condor.



tion decisions. This leaves a single point of failure through which the load sharing scheme can fail. This problem is minimized as only yet to be allocated tasks are effected. Since the central coordinator has a rather simple job, it can be restarted at another site if required.

STARS¹ — A Distributed Allocation Environment

STARS is a mechanism that provides distributed task allocation based on the demand for and supply of resources in a heterogeneous UNIX workstation environment. The system has been in use (in various forms) at the Computer Science Department, Victoria University, for over three years providing allocation for a variety of clients. In the previous sections, we have described various forms of allocation for idle workstations. The STARS allocation philosophy is somewhat different. Our method attempts to model more closely the economics of resources in a typical distributed workstation environment. By matching available host resources and task resource requirements, we hope to make more informed decisions on which hosts are more suitable for different tasks.

Before embarking on a tour of the allocation scheme that we use in STARS, we look at an overview of the subsystems that make up the allocation system. Following a brief description of the allocation scheme, some examples are given of how a distributed allocation mechanism such as STARS fits into a typical users environment.

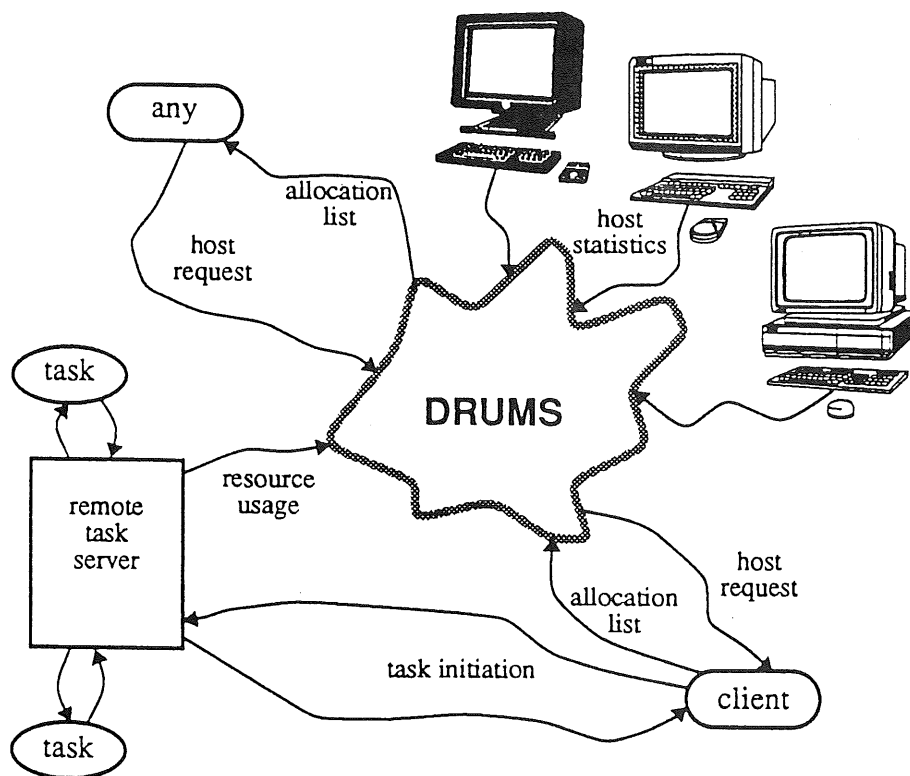
An Overview of STARS

STARS is a multifaceted system that supports distributed task allocation through a set of cooperating servers. FIGURE 5 on page 7 shows how the various parts interact to provide this allocation service.

- **DRUMS** is a robust and adaptive server set that roams the network providing access to statistical information about hosts and tasks. Two sets of servers are used, repositories of replicated data and collectors of statistical measurements. The servers minimize their impact on the computing environment by moving from busy hosts to those that have available resources². As the servers move, they are also able to increase or decrease their numbers according to their current processing loads. By a complex

-
1. Shrewd Task Allocation via Resource Scheduling
 2. In fact, both sets of servers use STARS to allocate new server instances.

FIGURE 5 The STARS task allocation environment.



set of checks and balances, each server set is robust to all but the most catastrophic multiple host failures.

DRUMS performs two main tasks.

1. The duplicated collector set is responsible for gathering the resource statistics from each host in the distributed environment. This data is then passed on to the fully replicated set of databases. Databases store this host resource information as well as task resource usage data passed on by the remote execution unit (described below). The databases also maintain a list of currently executing tasks that have been invoked through STARS.
2. The most important task of DRUMS is to respond to client allocation queries. The client provides a host requirement description and is given back a ranked list of hosts which best fulfil this criteria (this allocation scheme is described in "The Adaptive Host Allocation Scheme" on page 8).

- **Statistical daemons** are present on each host in the networked environment. These provide the statistical information gathered by DRUM's collectors. The resource statistics are obtained by periodically gathering data from the local UNIX kernel.
- **Any** is an example of a primitive interface to DRUMS. It packages client queries and ships them off to a DRUMS database for processing. The database returns a list of ranked hosts ordered by how well they fulfil the query. Options to **any** specify the order and quantity of hosts to be printed on the standard output. Typically, **any** is used to select a host for remote execution using the remote task interface. More explanation of this interface is given in "Incorporating STARS into a Distributed UNIX Environment" on page 9.
- A **remote task server** runs on each host in the networked environment. A client front end provides user access to remote computing facilities through these remote task servers. The mechanism performs a job analogous to `rsh` with a few minor additions:

- I. The client front end uses an alias file to find
 - a. the exact location of the executable file on the remote host
 - b. the environment variables that need transportation to the remote host
 - c. which streams of I/O should be transported from the client to server and back

This elaborate process means no shell need be invoked on the remote host, thus saving on remote task setup time.

- II. On connection, the remote task server registers the initiation of all tasks with the DRUMS databases. The databases then have a list of all tasks currently running in the STARS environment. Upon task completion, the resource usage information is sent to the databases who maintain this task resource usage history for future use by the task allocation algorithm (see "The Adaptive Host Allocation Scheme" on page 8).

The failure of a node within STARS is handled easily. If a DRUMS database was running on the failed host, the other databases will notice an increase in query load and start a new database. Since the database contains only replicated information, the loss is not vital. However, the unique information in collectors is lost if their host fails. The loss of host statistic updates is eventually noticed by the databases and consequently, a database begins to re-register the hosts with out of date statistics. This re-registration process will lead to the creation of a new DRUMS collector (see [Bond and Hine, 1991] for more details). Obviously remote processes executing on the failed node will be lost (but no Condor-like restarting is attempted). The databases will notice that tasks have stopped on the failed host when no task resource usage information is returned and consequently, their execution entries will be deleted from the databases.

The Adaptive Host Allocation Scheme

The task allocation scheme used in STARS is based on dynamic resource statistics gathered by DRUMS collectors. This information is updated every few minutes and when new tasks are started. As well as dynamic resource availability information, the allocation is based on dynamic task resource requirement data. This feature is unique to STARS.

A client to the allocation scheme passes a task alias to a DRUMS database requesting a ranked host list. This alias indexes a table listing characteristics of hosts able to run that task. For example,

```
latex
COMPSCI && (SUN4 || HP300) && hostname != embassy
```

specifies that latex must run on a computer science machine which is a sun4 or hp300 but isn't called embassy. In addition to the characteristics mask to select hosts, the allocation scheme requires a method for ranking hosts. This is accomplished via a resource weight vector specifying how important each resource is to the task. If DRUMS has not seen another task like this before it uses a default set specified in the alias table. For example,

```
max free_virtmem 0.2 , min load_1 0.3, min num_users 0.2,
max mips 0.3
```

Here we name each resource and specify how much relative emphasis is to be placed on it's value. In this example, 20% emphasis on free virtual memory, 20% on the number of users, 30% on the 1 minute load average, and 30% on the relative processor speed¹ of the host.

Each time a task completes, DRUMS adds it to its task resource history. Unfortunately, the resource usage values for tasks and the resource availability measurements available from hosts are not easily comparable. To solve this comparison problem, we introduce an intermediate metric vector that each measurement set can be translated to. The ranked host list for a task is found by producing a metric vector for the task (either from the default weights or from task resource usage history) and ranking how well it fits into the metric vectors computed from host re-

1. MIPS do not refer to Millions of Instructions per Second but rather a totally arbitrary speed rating guessed at by the author.

source availability. As more tasks are run, DRUMS uses the task resource usage feedback loop to learn a tasks resource requirements and eventually makes better allocation decisions for this task.

Work is currently being done to more precisely estimate task resource requirements by not only using the task alias as resource usage selection criteria but also the user running the task, their user group, time of day, host run from, etc. Each task may have several different subsets of resource usage estimates. This extra information should lead to more accurate resource usage predictions and finally better task allocations.

Incorporating STARS Into a Distributed UNIX Environment

The introduction of load sharing mechanisms into user environments is sometimes regarded with suspicion. By competing for idle resources, the user feels that "their" workstation is under siege. STARS has been introduced slowly by creating simple interfaces to the allocation scheme. As mentioned before, any is used as a simple host allocation interface for users who want to add remote host allocation to their current working environment. Currently, a more complex interface incorporating the remote execution feedback to DRUMS is hidden behind shell scripts.

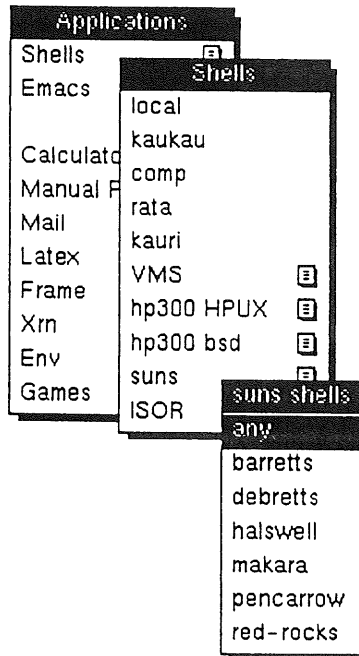
Many users have introduced any into shell aliases and popup menu items. A common use is an alias of the form

```
$ alias emacs "rsh `any emacs` emacs </dev/null &"
```

or to use the remote execution form with resource usage feedback

```
$ alias emacs "re-client -h `any emacs` emacs"
```

This might appear in a menu as



In addition to an alias table for allocation information, we have previously described an analogous table for remote execution information that is used by the remote execution client. An entry for emacs may look like:

```
emacs /usr/local/bin/emacs PWD;HOSTTYPE NONE
```

In column order these are:

- the alias name

- the remote location of the executable
- the environment variables that need to be set up on the remote host
- which I/O streams need be sent to and received from the remote task¹

If this information is unavailable for an alias then the default is to set up a remote shell and follow the standard `rsh` procedure for remote process execution. However, if the alias is found in the table, the remote execution is much quicker (approximately 30% of `rsh` executable time) as no shell need be invoked on the remote host.

STARS has also been used in conjunction with various applications where multiple subtasks can benefit from running in parallel. This has included work on a distributed ray tracer which breaks the workload up and distributes server ray tracers to run on hosts allocated by STARS. Another simple application is a parallel make. GNU make supports background processing of compilations in parallel. For example,

```
$ gmake -j 5
```

will execute at most five background compiles in parallel. By assigning the default compiler to be a task allocated through STARS, we can benefit from parallel compilations on separate hosts. We might accomplish this by setting the internal `CC` variable in `gmake` to

```
CC = re-client -h `any gcc`
```

As an example, we have compiled GNU make using this parallel compilation method. GNU make comprises over 13,000 lines of code in 23 C source files. The tests were conducted on a population of 17 HP300 type machines running `gcc` version 1.40. Each result is the average of four runs.

TABLE 1 Comparison of make times for compilation configurations.

Test #	# Machines	# Parallel Compiles	Distributed/Local	Make time (sec)
1	1	1	Local	568
2	1	1	Distributed	492
3	1	5	Local	297
4	5	5	Distributed	239
5	10	10	Distributed	119

TABLE 1 on page 10 shows the results of five separate tests. Tests 1 and 2 compare sequential local compilation versus sequential distributed compilation using STARS. We only see a modest 13% decrease in execution time for the entire make since most hosts, including the local site, were relatively idle. Test 3 executes 5 parallel compilations on the local host. This is almost 50% faster than the local sequential compile indicating that the machine was not using all its resources when compiling sequentially. Eventually a bottleneck will be reached, limiting the number of parallel local processes that can be executed². Test 4 also performed 5 parallel compiles but this time they were allocated using STARS. We only see a 20% decrease in total make time from the 5 local parallel compiles but a 58% decrease in total time from the original local sequential make is still impressive. This lack of significant decrease over the parallel local compilation shows that a significant local bottleneck has probably not been stressed yet. Finally, test 5 performs 10 parallel distributed compilations allocated using STARS. Here we see an 80% decrease in processing time over the initial local sequential compile and an impressive 50% decrease over the parallel make using 5 machines — by doubling the number of processors, we have halved the execution time. You really couldn't ask for much more than this!

1. NONE effectively isolates the remote execution and the remote execution client returns immediately.
2. In fact, it was not possible to perform 10 parallel compiles on the local host as it ran out of virtual memory.

Conclusion

Load sharing is a simple and effective means of maximizing computational resource usage in a distributed computing environment. By applying simple allocation strategies, the average response time is reduced. A load sharing mechanism may be implemented as a centralized scheduling server as in Condor but a single host failure can effectively cripple this load sharing scheme. By distributing the server as in Butler or STARS, the mechanism becomes more robust to such host failures. Butler, Condor, and STARS all use dynamic system information to make effective allocation decisions. The first two are only concerned with idle host allocation while STARS attempts to allocate hosts matching task resource requirements and host resource availability. Condor goes beyond the other two systems in providing preemption and migration facilities at the cost of extensive kernel modifications to support this feature. The other two systems are able to slot into conventional UNIX implementations with no such kernel nor application enhancements.

Users are able to incorporate load sharing mechanisms into their working environment with little effort. By adding appropriate aliases, menu options, and shell scripts the details of load sharing calls are hidden from the user. As well as decreasing job response time, the allocation mechanism is also able to enhance any inherent parallelism in application subtasks. This feature can increase effective processing power by as much as 450% over traditional sequential methods on a single host. In conclusion, a load sharing environment increases global system performance for little overhead or user effort.

References

- Bond, Andy; Hine, John (January 1991) "DRUMS: A Distributed Statistical Server for STARS", Proceedings of the Winter 1991 USENIX Conference, Dallas, Texas, pp. 335-348.
- Freedman, Dan (January 1991) "Experience Building a Process Migration Subsystem for UNIX", Proceedings of the Winter 1991 USENIX Conference, Dallas, Texas, pp. 349-356.
- Litzkow, J. Michael; Livny, Miron; Mutka, Matt W. (June 1988) "Condor — A Hunter of Idle Workstations". Proceedings of the 8th International Conference on Distributed Computing Systems, San Jose, California, pp. 104-111.
- Lyon, B.; Sagar, G.; Chang, J. M.; Goldberg, D.; Kleiman, S.; Lyon, T.; Sandberg, R.; Walsh, D.; Weiss, P. (January 1985) "Overview of the SUN Network File System", Proceedings of the Summer 1985 USENIX Conference, Dallas, Texas, pp. 1-8.
- Markenscoff, Pauline; Liaw, Weikuo (August 1986) "Task Allocation Problems in Distributed Computer Systems", Proceedings of the 1986 International Conference on Parallel Processing", pp. 1037-1039.
- Nichols, David A. (1987) "Using Idle Workstations in a Shared Computing Environment". Proceedings of the Eleventh ACM Symposium on Operating System Principles, pp. 5-12.
- Ritchie, D. M.; Thompson, K. L. (1974) "The Unix Time-Sharing System". Communications of the ACM, 17:7, pp. 365-375.
- Scheifler, Robert W.; Gettys, Jim (April 1986) "The X Window System", ACM Transactions on Graphics, volume 5, number 2, pp. 79-109.

by Stephen Walli

Report Editor

<stephe@usenix.org>

USENIX Standards Watchdog Committee

POSIX – Caving In Under Its Own Weight

“Standards are commercial and international politics dressed up as technical arguments.”

I think POSIX is caving in under its own weight. All of the hard nuts-and-bolts work effort of defining a technical programming specification is slowly being mired in the mud. POSIX exists to “... support application portability at the source-code level. It is intended to be used by both application developers and system implementors.”¹

It has been floundering for some time in a mess of its own making. I want to look at this mess, describing it and its historical context, and offer up a few possibilities for solutions. This article is long, but there is a lot of context that needs to be understood to see what’s happening to an otherwise useful standards effort. The article ends with a list of e-mail addresses to which you may wish to send any questions and concerns. In fact, I encourage it, and hope that you’ll be convinced by the end of the article.

The Problem

There are two sets of people doing work in the POSIX working groups. The first set sit in the individual working groups, distilling historical practice and experience into a technical specification “for application developers and systems implementors.”

The second set of people have typically been involved at the working group level for quite some time. They are often chairs of the groups or other officers. These people have begun to have coordination meetings and form steering committees outside the working group structure. All of the pieces of POSIX are related to one another, and there is a genuine need to coordinate between the different groups of heads-down-over-the-specification-technicians. The bureaucracy has grown because of need rather than

1. ISO/IEC IS 9945-1:1990,1.1 Scope, p.1, lines:2-3.

desire to hold extra meetings. Most of the people involved can think of more enjoyable ways to spend their time.

I wander in these steering committees, sub-committees, and the hallways of POSIX. It quickly became apparent to me that this is where the politics that drives POSIX is most on display. I was eventually around long enough to get involved in some of these committees. (Fool me.)

There has been a strange tension in these rooms for quite some time, coupled with a terrible confusion and sense of apathy. This is not noticeable in the working groups themselves. Heads down and oblivious to the politics of POSIX, the working groups are buried in the religious wars and politics of their own technical specification.

A couple of POSIX meetings back, it began. First in one steering committee, then another, and another. The group would hit a crisis point, and throw up its hands. Despite the fact that each room contained people with a long history and knowledge of POSIX, they would reach a point of apparent confusion as to how to coordinate with another steering committee or sub-committee. (The running joke is that we need a steering committee steering committee, but it really isn’t seriously contemplated.)

Finally, someone would suggest we need to define the problem. I offered to go away and write it up. (More fool me.) Then the next sub-committee meeting. The same process. Tension, confusion, “let’s define the problem.” It started in the Project Management Committee. I later saw it in the Steering Committee for Conformance Testing, then the System Interface Coordination Committee. These are all really fundamental sub-committees, with a lot of POSIX history in their membership.

The coordination complexity is amazing. The major areas of POSIX requiring coordination are the base documents themselves, their test methods, and their structure with respect to language independent specifications (LIS) and programming language bindings. (This complexity has spawned profiles, about which I’ve yelled enough for now.)

† This is a re-print from ;login, the USENIX Association Newsletter, Volume 18 Number 1

Steering committees were thought to be a way out of the mire. If we just communicate with one another, the problems will all become apparent, sort themselves out, and go away. But ultimately this falls down. POSIX is too big. The steering committees have no authority to impose their collective will. POSIX is a volunteer effort. There are no sticks and there are no carrots.

If it becomes too much trouble to build the standards, then the volunteers will cease to arrive at the meetings. The POSIX standards effort will fail. Or worse yet, they will continue to be defined by fewer and fewer people with sound technical background and a proper perspective on the subject. This will cast doubt on the good work which has already been done.

Test Method Madness

To ensure that implementations of the POSIX.1 standard could somehow be tested and certified in a uniform way, the POSIX.3 standard (Test Methods) was created. This work was heavily supported and resources provided by the United States government, along with the testing agencies that were supporting the actual testing requirements.

The POSIX.3 standard is not a bad thing. It defines a methodology by which test methods and results of test cases written to these methods can be uniformly described.

If you are creating a standard it's a useful tool to ask yourself "how would I test this functionality or feature" as you write the specification. It ensures you read and possibly rewrite the specification properly. You may wish to deliberately not be complete in the definition, but these areas in a standard specification should be intentional.

This "testing" tool has even been proven. Several working groups have written test methods for their specifications, with some help from people historically involved in the original POSIX.3 effort. Many of these POSIX.3 members have formed the Steering Committee on Conformance Testing (SCCT) that oversees how test methods are applied and created in the working groups. The SCCT has been too busy to review these test methods in depth, but without judging whether the new test methods are good or bad, the working groups that have gone to the trouble of creating them have all felt that their base specifications are better defined for the effort. It seems that the tool works!

Now for the problem. Some time ago, the SCCT recommended to the Sponsor Executive Commit-

tee (SEC) that all POSIX standards must have associated test methods. These test methods would be standards as well. They convinced the SEC to make this a requirement.

Now, a standard cannot officially exit balloting without having a test method specification that is also a standard. This instantly sets up a directly competing body of text to the original standard. This is not a competing functional standard like IEEE 802.n LAN standards. This is a competing body of text. (Note: ALL discussions of formal testing languages and formal specifications are red herrings here. Anyone wishing to hear my three Canadian cents worth on the subject can email me.)

Test methods standards will become the appointed specification for the test suite to demonstrate conformance by organisations with the funds or market presence to demand as much. Implementations can hit the narrower mark of the test suite (embodying the standard test methods) to naively certify rather than hit the standard itself. If you don't realise the subtle and nasty differences that can appear, spend some time with the POSIX.1 standard (IEEE Std 1003.1-1990), and with its newly declared standard test methods (IEEE Std 1003.3.1-1992).

And what happens when there are holes in the test methods? Some things cannot be tested. The standard still has requirements on these areas of behaviour, but they may not translate nicely. And there are some places where the test methods simply aren't complete. A reasonably recent draft of the POSIX.3.1 test methods had test methods for the POSIX.1 environment variables required by U.S. FIPS PUB 151-1 (the U.S. government profile of POSIX.1), but none for the other environment variables. The international community might wish to take note of this oversight on all LC_ environment variables, should the POSIX.3.1 standard get to ISO. What other holes are there?

There is a terrible balloting problem. Balloting apathy or overload is striking many places. The test methods documents are as big as the standards they repeat. Fewer people care about the test methods, they've seen the original specification and the job is done, right? We run the terrible risk of passing bad test methods documents if these documents are quickly processed through balloting groups whose members have little time on their hands. In the current commercial climate for standards, this is dangerous.

Then, of course, there is the maintenance problem. All useful standards have the same problem as all useful software. They need to be main-

tained. It's just slower and more tedious. A level of complexity has been added to the administration of the interpretations.

POSIX.1 has the fun little contradiction that `PATH_MAX` is the length of the pathname both explicitly including and excluding the terminating null byte. An interpretation was requested, and came back that it was an inconsistency and that both can be right.² Now what happens when someone requests an interpretation of a standard with its test methods?

If the request is leveled against the base, what guarantees are there that the test methods, i.e., a separate standard, will be kept synchronized? If it's against an inconsistency between the base and its test method standard, which one wins? If the `PATH_MAX` argument holds, then both are correct. Since one of them is implemented as a test suite to demonstrate conformance, which one wins in the real world?

Do test methods need to be standards? Who wins by forcing working groups to completely re-specify their work as test methods? Testing is expensive, but the market ultimately protects itself. What has been done in the TCP/IP space? (If you don't think TCP/IP is a successful widely implemented specification, stop reading now.) What about the C language? No one specified a set of test methods for the ANSI C standard. People in the know wanted to see how to test the C standard, and through a lot of hard work built the Plum-Hall test suite. The U.S. government created a FIPS for C and chose an available suite. There were no test methods for this work. No added burden on the volunteer standards community to respecify itself.

A great tool; but only a tool!

LIS – The Great Experiment

Language Independent Specification (LIS) is burden Number #2 on working group members. Two working groups have been operating in the POSIX space for quite some time in programming languages other than C. One is the POSIX.5 Ada Bindings group, which has re-cast the POSIX.1 standard into Ada, and is now working on POSIX.4 (Real-time Extensions). The second is POSIX.9 which has similarly cast POSIX.1 into FORTRAN 77, and is now considering what to do with Fortran 90. The two groups have finished their work. Two real standards exist within the IEEE standards realm:

2. IEEE Std 1003.1-1988/INT, 1992 Edition, Interpretation Number: 15, p. 36.

IEEE Std 1003.5-1992 (Ada Bindings to IEEE Std 1003.1-1990.)

IEEE Std 1003.9-1992 (F77 Bindings to IEEE Std 1003.1-1990.)

A small digression is required on ISO POSIX. Along the way, IEEE POSIX entered the international community and an ISO Working Group (WG15) was created as its home in the Subcommittee on Programming Languages (SC22). WG15 is not a standards development group per se, in that it does no drafting of specifications. Its job is to review the draft IEEE documents and make recommendations to the IEEE, through the ANSI sponsored U.S. Technical Advisory Group (TAG) on POSIX, back to the POSIX Sponsor Executive Committee.

Do not be fooled. There is a substantial overlap in the key personnel of the IEEE working groups and people sitting in the WG15 meetings as individual technical specialists from their respective national POSIX standards groups.

ISO began trying to specify programming interface standards in programming language independent ways, such that the functional specification appears once, with multiple bindings. It seems expensive to continually re-specify a standard from one language into a standard in another language. There is the feeling that there is twice the work effort, plus the coordination effort.

A different international group, WG11, is working at defining abstract data types and such. All programmatic interfaces could eventually be described in some abstract functional way and each individual language binding would just "fall out" once the mapping from the abstract types to program language types had been established. Because of early experiments in specifying standards this way, language independence was inflicted on POSIX as a requirement from WG15. POSIX the Guinea Pig. WG11 had never been faced with POSIX.

All this means every standard becomes two standards. There is a book describing the functional specification in abstract data types, and a book specifying a mapping to a real programming language's syntax, along with additional required semantics. Try re-reading each of the last few paragraphs, and after each repeat, "It is intended to be used by both application developers and system implementors." Ideally, ISO WG members believed that the functional specification would be a "thick" book, and that the language binding would be "thin."

The Ada group, POSIX.5, chose not to split their work. They argued it was too late in their project and that a sufficiently mature POSIX.1 LIS did not exist. They further argued that they had to produce a "thick" language binding reproducing much of the semantic content of the POSIX.1 book, recast into Ada-speak, in-line. Programmer usability was very high on their list of priorities. Think about that for a minute.

I work in an environment where we regularly refer to the POSIX.1 standard. We write code that needs to be portable to many non-Unix based architectures that provide POSIX.1 interfaces. All of our many copies of POSIX.1 are very dog-eared and marked up. We use our copies daily. It is a useful book from which to program. It is not a tutorial. It is a programmer's reference.

I recently had to go through the POSIX.5 and POSIX.9 standards. I am not an Ada programmer, but still found the information I needed to find, in an easily understandable form. The POSIX.5 group did their job well. Yes, it is a thick binding repeating the semantic functional material of POSIX.1. And yes, even though the POSIX.5 standard is supposed to exactly mirror the POSIX.1 standard, I found a bug (or at least something about which to request an interpretation). But I found the information, clearly laid out; even the bug!

The POSIX.9 (FORTRAN 77) working group chose to attempt a thin language binding to POSIX.1. They were very tight for resources and they wanted to do the right thing with respect to the ISO WG15 requirements. Through no fault of their efforts, I found it to be a difficult book to use, and I was a Fortran programmer in a previous incarnation.

First, you immediately run into the two book issue. Look up the syntax in POSIX.9 which immediately punts you to the semantics in POSIX.1. So you jockey about two books in your lap, continually cross referencing.

Second, you continually switch frames of reference. In one book, there is a solid real world line of language syntax; in the other book, a description of that syntax's semantics in a different specification language (C).

In balloting the POSIX.1 Language Independent Specification (LIS), I ran into the same problems. Two books, two frames of reference. At least POSIX.1 Classic (IEEE Std 1003.1-1990 == ISO/IEC IS 9945.1:1990) stands as an existing reference against which to compare these models. When we begin balloting drafts of API standards as LIS

and attendant bindings in at least one language, will we be able to catch all the holes?

The IEEE paid to have the initial drafts of POSIX.1 LIS and its C binding (POSIX.16) produced. They couldn't get the work done any other way. Paul Rabin worked long and hard to produce guidelines for writing LIS and language bindings. This work was done within the IEEE POSIX realm, although Paul liaised closely with ISO WG11 and WG15. The few IEEE POSIX working groups that have attempted partial or complete drafts of their work using these guidelines, have immediately started finding problems in their previous C language specific descriptions. Just like test methods, prodding the text by attempting to recast it into a different form made a better specification.

Again, one has to ask if this is a good way to define standards. A tool to test the specification, yes. The specification itself? One has to assume that the standard has an audience, and that usability is an important factor. One should assume that the standard is based on existing practice for the most part. That existing practice is in a particular programming language for API type standards. Those will be the first people to come forward to develop the standard. (There has to be a need to standardize.)

If others with a different programming language background participate, this would be ideal. If the experience with the functionality exists in more than one language, and they all want to come to the table, this is even better. But we do not live in an ideal world. Specifying the functionality in a hard to use (2 document/2 context) format is error prone, especially when the document is being balloted. Until formal methods become a common method of expression, we are stuck with English descriptions, and the exacting programming language syntax of the existing body of experience in that area of functionality.

Language Independent Test Methods

Yes, you read the title correctly. If the functionality can be abstracted, described exactly, then bound in various programming language syntaxes, so to can the test methods of that functionality. Think about how you would test an Ada run-time implementation of POSIX.1.

And each is a standard. So there is a base programming language independent functional specification (LIS) standard, a programming language binding standard, the LIS test methods standard, and the language binding standard for those test methods. Balloting will kill us. We will produce unusable junk if we continue.

Simple economics says we're doomed. The IEEE is being forced to pay up into ANSI for its international standards efforts. To cover the costs of simply balloting the quantity of paper, the IEEE has been forced to start charging \$25 US to join balloting groups. To cover the international participation, they've considered raising this to \$50 US. That means it will cost the individual professional programming member of the IEEE \$200 to join the balloting groups for a set of standards that represent a simple piece of functionality in which they are interested.

One might argue that a programmer will only join two balloting groups, for the LIS and language binding. Because the test methods (LIS and language binding) are a competing body of text, however, they will need to check the test methods to confirm they are accurate. Because of government procurement policies here and abroad, the test methods will be important!

An Architect's Nightmare

LIS, language bindings, LIS test methods, and their bindings. Now imagine that we start amending the four standards at once. POSIX.6 (Security Extensions to POSIX.1 and POSIX.2) will amend POSIX.1 and POSIX.2 somehow at some point in the not too distant future. So will POSIX.4 (Real-time Extensions), POSIX.8 (Transparent File Access), and POSIX.12 (Sockets/XTI).

The original POSIX.6 document, which did contain all the information they could put together on POSIX security has just needed to be split SIX ways:

- The API as an LIS, to amend POSIX.1/LIS,
- The API as a C-binding, to amend POSIX.16,
- The API test methods in LIS form, to amend POSIX.3.1 (which currently isn't in LIS form),
- The API test methods as a C-binding, to amend POSIX.3.1 (in its current C form?),
- The utilities, to amend POSIX.2,
- The utility test methods, to amend POSIX.3.2.

Can't wait.

The Problem Revisited

If POSIX continues on its current course, one of two things will happen.

ONE – They will succeed. The useful standards which do exist will be amended to a user unfriendly form. An ugly unusable set of stan-

dards will eventually be born. Because of the lack of use, they will fail. People will not use them. It will be too easy to ignore them. Programmers will not be able to rely on a certain portability model. The vendors will continue to sell completely proprietary implementations.

TWO – They will fail. Under its own weight, it will collapse. If not with a bang, then with a slow sickening crunching sound. The people with the knowledge will get tired, or lose support (as they obviously aren't producing anything to show their management in recessionary times). POSIX.1 will become unusable as it is amended and amended and almost amended. ("If we wait for another 6 months, we'll be able to get all the wizzy features in POSIX.42....")

ONE AND A HALF – Life isn't this black or white. The ugly truth will lay in the middle. We're talking about several thousands of pages of functional specification. We're talking several hundred people in working groups, plus hundreds more in balloting groups, plus the unsuspecting time-delayed purchasing public. The death will be long and painful. Senility will set in first.

Solutions!

OK. Let's stop the gloom and doom. Let's take an optimistic pro-active view! What to do about the problems of POSIX? Let's put it on a diet.

Remove the continued requirement on balloting the test methods as standards. The Steering Committee of Conformance testing would no longer have a function. Its members could go do real work in the POSIX.3 update effort, adding to a useful document which provides a tool for testing the specifications developed in working groups.

These working groups would immediately cease worrying about developing complete test methods documents. Those that cared, would, when occasionally confronted with ugly passages in their drafts, have a useful tool (POSIX.3) to use to try answering the question, "how would I test this?"

Ballot groups could concentrate on the real specification in front of them. Repeat again: Bad test methods standards will be dangerous in the marketplace.

Individual technical members in working groups could stop worrying about completely re-specifying their document. Possibly some that cared, with the newly found time, might actually write

some real honest-to-god test cases. These would surface, instead of everyone waiting to see which way the testing wind was going to blow by large governmental agencies here and abroad. These test cases might even be used, therefore useful.

Should these large governmental testing concerns wish to compare the merits of test suites, they could require that they are documented, and record results according to POSIX.3. Render unto the standards community that which is the standards community's, and render unto the marketplace that which is the marketplace's.

Who can act on this recommendation? The IEEE POSIX Sponsor Executive Committee can. They are made up of the working group chairs, the steering committee chairs, and institutional representatives. There is a list of these at the end of the article, with email addresses. Send them e-mail. It really only takes a minute. It will save you a lot of future grief to take the minute to ask questions NOW!

There is also a list of some important heads of delegations within the ISO POSIX WG15. WG15 is considering forwarding IEEE test methods documents as standards at the international level. Then we can all live with any mistakes in the U.S. government procurement policies! E-mail soon! E-mail often!

Let's continue the POSIX diet. Programming Language Independent Specifications should be stopped for the time being. The IEEE has put forward an incredible good faith attempt. The experiment should be considered a success! We have demonstrated that we don't yet know enough about specifying API standards in this abstract way. We should cease to hold up the working process.

Once the problem is better understood, and our methods of describing things in an LIS improve, we can begin exploring the possibilities. Notice that I didn't say retrofit or recast. I said explore the possibilities. Until we actually add a few of the large amendments to the base standard, changing its format midstream just opens things up for abuse and error. Let's do it a few times in languages that many of us understand, i.e. C, Fortran, Ada, before tackling the problem with little understood methods, which have been untried at this scale.

What would happen? Working groups would spend less time trying to recast their work (again!) into LIS. They would spend more time on the real specification, making it usable "for application developers and systems implementors."

When the existing working groups want to bind something in more than one language, they arrange to attend one another's meetings, and they work together. This sometimes takes the form of the complex strained negotiations that are the consensus process. This process is already in place in POSIX and has been for some time. It works. The LIS has not been required in producing the usable standards documents to date.

Who can act on this recommendation? Once again, the IEEE POSIX Sponsor Executive Committee can. This one is harder, however, as ISO WG15 is also involved.

First, the SEC has to be willing to say "no". This is not a surly uncooperative "no". A huge work effort has gone into the LIS experiment. There is real experience in the IEEE POSIX projects with this. The SEC can say "no" with confidence based on experience. ISO cannot claim the same experience. (If they could, they would have been helping us a long time ago.)

Second, ISO WG15 has to be willing to say "no." Remember that there is a sizable overlap in the small membership of WG15, and members of the SEC. The IEEE POSIX working groups have many international members who show up in the Canadian, UK, American, and German delegations. Education is certainly not the problem here, however, communication might be.

Other special working groups within ISO may be concerned with this approach, but again the experience lies within the IEEE POSIX working groups, which overlap with ISO WG15. Other ISO concerns should be acknowledged and put to rest. Once again I say: E-mail soon! E-mail often!

Ultimately, in a worst case scenario some level within ISO could refuse to accept IEEE POSIX drafts for ISO balloting. I believe even this case should not be of concern, based on the following examples:

ISO WG15 has not accepted the perfectly useful IEEE POSIX.5 for international standardization, since it did not fit the ISO requirements. ISO WG9 (ISO Ada Working Group) has been very concerned by this action and is attempting to fast track the IEEE POSIX document.

A representative from AFNOR (France's National standards organization) voiced strong support for the IEEE POSIX groups to continue to bring forward the standards as LIS at the last ISO WG15 meeting. He then immediately expressed grave concerns that POSIX.4 be brought forward

as quickly as possible in its current C-based form to the Draft International Standard (DIS) state. You see, the French government can procure against a DIS.

Ultimately, if the IEEE POSIX working groups do their job right and produce useful and usable standards, the market will demand their use, even if they have to be fast-tracked into the back door to make them international standards for the international market place. Twisting the standardization process away from defining detailed

specifications towards suiting procurement processes from organizations that are too big to change is wrong!

POSIX has market momentum. It will affect the way you do things. The working groups have produced useful standards, but that is now in jeopardy. You can affect the process. If you can't get directly involved, e-mail the appropriate people below and ask questions! Explain your concerns! Otherwise, you'll have to live with their decisions.

Who Ya Gonna Call?

Position	Name	E-mail
IEEE Concerns		
Chair SEC	Jim Isaak	isaak@decvax.dec.com
Vice Chair Interpretations	Andrew Twigger	att@root.co.uk
Vice Chair Balloting	Lorraine Kevra	l.kevra@att.com
Chair Steering Committee on Conf Testing	Roger Martin	rmartin@swe.ncsl.nist.gov
Chair Project Management Committee	Shane McCarron	s.mccarron@ui.org
Chair POSIX.1	Paul Rabin	rabin@osf.org
Chair POSIX.2	Hal Jespersen	hlj@posix.com
Chair POSIX.3	Lowell Johnson	3lgj@rsvl.unisys.com
Chair POSIX.4	Bill Corwin	wmc@littlei.intel.com
Chair POSIX.5	Jim Lonjers	lonjers@prc.unisys.com
Chair POSIX.6	Ron Elliot	elliott%aixsm@uunet.uu.net
Chair POSIX.7	Martin Kirk	m.kirk@xopen.co.uk
Chair POSIX.8	Jason Zions	jason@cnd.hp.com
Chair POSIX.9	Michael Hannah	mjhanna@sandia.gov
Chair POSIX.12	Bob Durst	durst@mitre.org
USENIX Institutional Rep	Jeff Haemer	jsh@canary.com
EurOpen IR	Stephen Walli	stephe@mks.com
Uniforum IR	Ralph Barker	ralph@uniforum.org
DECUS IR	Loren Buhle	buhle@xrt.upenn.edu
OSF IR	John Morris	jsm@osf.org
Unix International IR	Shane McCarron	s.mccarron@ui.org
X/Open IR	Derek Kaufman	d.kaufman@xopen.co.uk
WG15 Concerns		
Convenor WG15	Jim Isaak	isaak@decvax.dec.com
US Head of Delegation	John Hill	hill@prc.unisys.com
Canadian HoD	Arnie Powell	arniep@canvm2.vnet.ibm.com
UK HoD	David Cannon	cannon@exeter.ac.uk
German HoD	Ron Elliot	elliott%aixsm@uunet.uu.net
Dutch HoD	Herman Wegenaar	(phone: +31 50 637052)
Japanese HoD	Yasushi Nakahara	ynk@ome.toshiba.co.jp
French HoD	Jean-Michel Cornu	jean-michel.cornu@afuu.fr
Danish HoD	Keld Simenson	keld@dkuug.dk

Report on POSIX.0: The POSIX Guide

Kevin Lewis <klewis@gucci.enet.dec.com> reports on the October 19-23, 1992 meeting in Utrecht, NL

The ballot submission period for POSIX.0 closed on September 15, 1992. Below are the ballot statistics:

86 ballot group individuals	
81 ballot group formal members	
<hr/>	
69 ballots submitted = 85% returned	
<hr/>	
11 abstentions	
30 negative	
28 affirmative = 48% returned	
(16 affirmative w/ no comments)	
<hr/>	
1127 comments/objections (approximate)	

POSIX.0 dedicated all of the October meeting towards ballot resolution. The section leaders are serving as the technical reviewers for ballot resolution. They received 30 ballots via e-mail approximately three weeks prior to the meeting. (Three of the 30 were from individuals not on the ballot list. The group decided to treat them as "parties of interest.") Fifteen were received by the ballot coordinator on the Friday before the meeting, so the technical reviewers saw these for the first time in Utrecht.

The group focused on identifying those objections felt to be substantive, key, or "show stoppers." The areas that these fell into include profiles, the reference model, and public specifications.

Let me note at this point that just about everyone in the group, including Yours Truly, demonstrated a clear case of memory shutdown, i.e., forgetting how we dealt with process and disposition issues during mock ballot. I attribute that to this last quarter requiring no working group activity aside from individuals' submitting their ballots. So it took the group about a day to "reboot."

In parallel, the guide is also in the review and comment process within WG15 and SC22. As of this writing, no comments have yet been received.

The TCOS SEC approved a resolution to forward the next draft of the guide, which will be the first recirculation draft, to SC22 for CD registration.

The group established the goal of completing ballot resolution within 7-10 days after the January

93 meeting. A tentative first recirculation meeting has been identified within the April 1993 time frame. This will be confirmed before the January meeting.

Overall, the guide is in good shape. The big question, implicit as it may be, is how well we will fare beyond the 75% requirement for affirmative votes before the guide can be published. It is too early to say. I'll have a much better feel after the January meeting.

Report on POSIX.2: Shell and Utilities

David Rowley <david@mks.com> reports on the October 19-23 meeting in Utrecht, NL

Summary

The grand moment has arrived, we have a final POSIX.2 Standard:

IEEE Std 1003.2-1992

Approved by the IEEE Standards Board on September the 17, 1992, POSIX.2-1992 is the culmination of over five years of the working group's efforts. The standard consists of both the "Dot 2 Classic" and "Dot 2a" components, previously balloted as separate standards. The IEEE Standard (based on the new Draft 12) is identical (at least from a technical standpoint) to the draft ISO standard, ISO/IEC DIS 9945-2:1992.

NIST continues to work on the draft of a new FIPS (Federal Information Processing Standard) for POSIX.2, expected in final form by early 1993.

POSIX.2b work continues to proceed, incorporating symbolic link support within a number of utilities, a new PAX archive format, and addresses a number of international concerns regarding locales. The PAX format is still based on the old but standard ISO 1001 tape format.

Test assertion work nears completion. The POSIX.2 assertions have almost full coverage, and will go to ballot again in December. The POSIX.2a test assertion work is going well, with almost all assertions complete, including vi. These will be folded in to the next draft of the POSIX.2 test assertions.

The test assertion work for POSIX.2 will be renamed P2003.2 instead of the current P1003.3.2.

Background

A brief POSIX.2 project description:

- The base utilities of the POSIX.2 standard deal with the basic shell programming language and a set of utilities required for the portability of shell scripts. It excludes most features that might be considered interactive. POSIX.2 also standardizes command-line and function interfaces related to certain POSIX.2 utilities (e.g., `popen()`, regular expressions, etc.). This part of POSIX.2, which was developed first, is sometimes known as "Dot 2 Classic."
- The User Portability Utilities Option or UPUO, is an option in the base standard (previously known as POSIX.2a). It standardizes commands, such as `vi`, that might not appear in shell scripts, but are important enough that users must learn them on any real system.
- Some utilities have both interactive and non-interactive features. In such cases, the UPUO defines extensions from the base POSIX.2 utility. Features used both interactively and in scripts tend to be defined in the base utility.
- POSIX.2b is a project which covers extensions and new requests from other groups, such as a new file format for PAX and extensions for symbolic links. It also includes resolution of items arising from comments by ISO Working Group 15.

POSIX.2 is equivalent to the International Standards Organization's ISO DIS 9945-2 – the second volume of the proposed ISO three-volume POSIX standard.

Publishing

Now that the Standard has been approved by the IEEE, everyone is anxiously awaiting the final published volumes. They will be printed on A4 paper in two volumes: the core standard (Sections 1-7), and the annexes. The set should be available from the IEEE sometime in the March 1993 time frame at a total page count of around 1300 pages.

POSIX.2 FIPS

NIST (National Institute of Standards and Technology) is still preparing the draft FIPS (Federal Information Processing Standard) for POSIX.2. The goal of the FIPS is to directly adopt, rather than adapt, POSIX.2 as a procurement standard. The selection of options and extensions will be left to the procurement officer. This will lead to even greater use of the standard, due to the flexibility this offers the agencies wishing to reference POSIX.2.

NIST Draft Request for Test Technology

NIST has issued a draft of a Request for Test Technology. NIST is seeking candidate test suites from which to select one test suite to measure conformance to the proposed POSIX.2 FIPS. It must be based on TET (Test Environment Toolkit from OSF-UI-X/Open), cover all assertions from POSIX.3.2, and satisfy the general test method requirements specified in POSIX.3. The suite must also be commercially available (either now or in the future). The full RFTT is due out early in the new year.

X/Open Request for Proposal

X/Open is in the final stages of signing the contract for the Integrator they have chosen for their combined POSIX.2/XPG4 Commands and Utilities test suite, to be integrated into a future release of VSX (Validation Suite for XPG). The Integrator will likely be publicly announced before the end of the year. Work is to start early in 1993, and should result in a suite being publicly available early in 1994.

Test Assertion Group Name Change

The IEEE is in the process of renaming the test suite working groups to a parallel numbering system to P1003. As of March 1993, the POSIX.2 test methods work will be numbered P2003.2. This should ease the confusion of many similar sounding working groups containing numerous dots and digits.

The ballot for Draft 8 of the POSIX.2 test assertions starts December 6th and ends January 6th. Some ballot resolution will be attempted at the January POSIX in New Orleans (the 11th to the 15th). Draft 8 includes assertions for all utilities except those from Section 5 of POSIX.2 (the User Portability Utilities Option, formerly POSIX.2a). These missing assertions will be included for the full re-ballot, Draft 9, expected sometime in March 1993.

POSIX.2b

The current draft of POSIX.2b, Draft 4 – August 1992, includes a number of extensions and additional utilities. The BASE64 encoding from MIME (Multipurpose Internet Mail Extensions, RFC 1341) has been incorporated into `uuencode/uudecode`. The "iconv" utility for character set conversion has been added from XPG4. Print field widths have been added to the "date" command. Support for symbolic links has also been added to a number of utilities.

Locales

A proposal from Thomas Plum regarding a new locale specification format from P. J. Plauger was discussed. Although the format has some interesting features, the codeset specific nature of the format limits its usefulness, and was deemed dangerous in a POSIX environment. A liaison statement to WG14(C), WG20 (Internationalization) and WG21 (C++) will be drafted by the Chair.

Yoichi Suehiro (DEC Japan) made a proposal to extend LC_TYPE to handle user-definable character conversions and user-definable character classes. These were both felt not to be within the scope of POSIX.2, but may be reconsidered at a later date.

Extensions to LC_TYPE were approved to specify the display/print widths of characters in the locale. This information will be specified by using the keywords "width1", "width2", etc. There will also be a "default_width" keyword which specifies the default width occupied by all characters not specifically mentioned in one of the "width" classes.

"era_d_t_fmt" had accidentally been left out of the LC_CTIME category. This will be corrected through POSIX.2b.

There was a long discussion on multibyte and stateful encodings and the need for coordination between ISO 9945-1 and ISO 9945-2. This will be discussed further in subsequent meetings.

New PAX File Format

The request for alternate PAX format proposals generated only a few pointers to other file formats, particularly the MIME standard (RFC 1341). Mark Brown has volunteered to write up a rough draft of a MIME-based PAX format to be discussed in New Orleans. Other than that, the group continues to work with ISO 1001. The group has also agreed to adopt Gary Miller's (IBM Austin) new File System Safe UTF (UCS Transformation Format) which specifically stays away from the codepoints representing the ASCII "/" character and null bytes.

Character set conversions issues within the PAX format can now be handled in a generic, system-wide manner given that the "iconv" utility has been added to the standard. This should result in a much more useful and consistent system for the user.

Report on POSIX.4, POSIX.4a, POSIX.4b, POSIX.13 (Real-time POSIX)

Bill O. Gallmeister <bog@lynx.com> reports on the October 19-23, 1992 meeting in Utrecht, NL

Summary

Well, for all those of you who've been breathlessly following the progress of the real-time POSIX proposals these last few months, you may have noticed a dearth of USENIX updates on the subject. Blame the snitch. He's a slug, and forgot to do the last report. This report will cover the last two meetings - July (Chicago) and October (Utrecht).

The real-time working groups are making quiet, steady progress on POSIX.4 and POSIX.13, which are two of our proposals that are out to ballot. In fact, we fully expect to turn POSIX.4 into a real live standard on or about January, 1993. (It depends more on when the high muckety-mucks of IEEE get around to it than on anything else, in my opinion.)

POSIX.13 is our profile document, which calls out what parts of POSIX you need in order to run POSIX on your Cray or your cruise missile, depending on what you may have. The situation with POSIX.13 is really pretty interesting, so we'll end with that to give you something to look forward to.

Rounding out our picture, we have POSIX.4a - threads - which seems to have completely vanished into the hands of the technical editors. Those of us who actually would like a useful threads standard sometime in this century are getting a little impatient. We have rather little recourse, however, since documents in ballot are not really the province of the working group anymore. Threads is a grown-up standard now and it'll just have to look out for itself.

And, finally, the Yet More Real-Time additions in POSIX.4b are proceeding apace in the working group.

POSIX.4: Real-Time Basics

Good news here. POSIX.4 is actually approaching finalization! After a couple of changes that had us a little worried (the addition of mmap(), and the change to semaphores from binary to counting), we found the balloting group basically agreed whole-heartedly with the way things were going.

That's not to say they didn't have plenty of other things to kvetch about, but then that's what bal-loters are for.

But at this point, we have passed Draft 13 through a recirculation, and from what I am told, the initial results look quite promising. Basically, very little of the POSIX.4 document is open to comment at this point, and the next circulation should be small, fast, and quickly resolved. That done, we can take POSIX.4 to the IEEE standards board at their June meeting. It is already in the Committee Document registration phase at the ISO WG15 level, on its way to international standardization.

POSIX.4 is one of the last standards that was allowed to pass without a language-independent specification and test methods. One of our next jobs is to produce a version of POSIX.4 in LI form, with test methods. A group of volunteers has been formed to start on that work, and should have some progress to report at the January meeting (but not much, given the holidays between now and then).

POSIX.4a: (The Long-Lost) Threads

What's going on with threads? Don't ask us. We're just the working group. As far as I've been able to tell, everyone involved in moving the threads chapters through their ballot has either lost interest, had children, gotten out of school and started making the big bucks, moved to France, or been involved up to their eyeballs in justifying their own continued existence at their various companies.

I'm told that threads needs to be kick-started a little bit. In Utrecht, we had a serious contingent of angry natives wanting to know what was up with threads. My prediction (and take it for what it's worth) is that the threads technical reviewers have until the January meeting to make some visible progress on their standard, or we might get some new technical reviewers who are less strapped for time.

POSIX.4b: Extra Real-Time Interfaces

This is a proposal that not many people know too much about, so I'll give a fast introduction to it. POSIX.4 was started to extend POSIX.1 for real-time. POSIX.4 settled on a subset of functionality for real-time – things we thought were absolutely crucial, and most importantly, things we could actually make some progress on. The more contentious items were left behind for a "future standardization" effort. That effort is POSIX.4b.

The facilities of POSIX.4b are more esoteric and less widely applicable, although they are absolutely essential for certain real-time applications. POSIX.4b has chapters for:

- direct application access to interrupts,
- device control (a.k.a. *ioctl()*, although we had to change the name to protect the existing),
- *spawn()* (a combined fork-and-exec which can be more easily performed than fork/exec on an MMU-less architecture),
- Sporadic Server scheduling (a scheduling discipline used in conjunction with Rate Monotonic Analysis to support, fittingly enough, sporadically-interrupting devices and other things that take unpredictable amounts of time),
- and CPU time monitoring (the POSIX.4 version of *times()*, essentially allowing one thread to monitor the execution time of another).

There is also work ongoing on extended memory management, something to allow one to allocate from distinct, special "pools" of address space (memory attached to a particular bus or device, in particular.) This chapter is up in the air and might go away.

The POSIX.4b proposal is proceeding along rather fast. It's a little terrifying to see a proposal that aims to allow an application to manhandle an interrupt vector, coming at you full speed ahead. Luckily, we have the (I hesitate to say it) stabilizing influence of people from POSIX.1 (who are interested in *spawn*) and sundry large, entrenched camps of UNIX aficionados in the group on an intermittent basis. Hopefully this influence will help produce something that is appropriate for standardization. It would certainly help, in my opinion, if more mainstream UNIX types were to give us a hand at UNIX-ifying the POSIX.4b proposal before it hits balloting. Maybe some of you nice people can drop in on the working group in New Orleans in January.

POSIX.13: Real-Time Profiles

This is the fun one.

POSIX.13 was the first profile proposal to hit balloting. We played by the rules. We produced our document. We formed our balloting group. We went to ballot. We got substantial approval, enough that very little of POSIX.13 should be open to comment on the next recirculation.

Oh, did I mention how POSIX.13 breaks just about every rule of how a profile document should be built? This unfortunate fact has led to some hand-wringing among the POSIX powers-that-be. The Powers would probably like for POSIX.13 to withdraw itself from ballot (despite the fact that it's mostly approved by the balloting group) and just go away until it can be reformed as a good POSIX citizen.

What are POSIX.13's crimes? Well, it's four profiles, not one. That's a problem, but not a big one. We could split the document with only minimal impact on the Spotted Owl population (and the lumberjacks would love us).

A bigger problem is that POSIX.13 calls for subsets of POSIX.1. Like, a POSIX without the ability to *fork()* (can't do it on an embedded, MMUless target), or *exit* (what sense does that make if you can't *fork()*?).

The smaller profiles of POSIX.13 are undoubtedly useful to people building embedded applications, however, there's a lot of consternation that something without a small modicum of UNIX-ness could possibly be allowed to call itself POSIX. So, lately, compromise wording was adopted in the committee whose job it is to make rules about profiles. That wording would allow the minimal profiles to be called "Authorized POSIX Subset Standardized Profiles," whereas something with a real POSIX.1 would be called a "POSIX System." And, of course, we would still need to convince POSIX.1 to subset itself.

Meanwhile, the POSIX.13 proposed standards are in the hands of – gasp! – people who are interested in doing real work. And it is clear that POSIX.13 would be useful for those doing real work, even if it is confusing and nasty by POSIX standards. [*ed. – Nasty pun, Bill.*]

I predict we'll see an essentially-approved version of POSIX.13 in a year, which will then have to wait for POSIX.4a to be finalized before the profiles really mean anything (you can't call out threads support when there is no threads standard). I further predict that the POSIX powers that be will declare POSIX.13 out-of-bounds, and that people will continue to use POSIX.13 anyway.

Report on POSIX.7b: Software Administration

Esti Koen <emk@cray.com> reports on the October 19-23, 1992 meeting in Utrecht, NL

I attended the POSIX.7b meeting in Utrecht, never having been previously exposed to POSIX. Lacking the historical perspective, it was difficult for me to identify when the discussion was a clarifi-

cation of an already agreed upon point versus a major shift in emphasis or direction. If this report seems somewhat lacking in detail or introductory, it reflects my own level of involvement to date.

For the purpose of this report, I assume readers are mainly interested in broad decisions concerning the content of the standard or a shift in direction and expected balloting dates.

Early attempts to standardize the nonexistent "common practice" of software administration seemed doomed to failure. (I don't envy those early pioneers.) POSIX.7 finally adopted the network view of a managed system. Forging ahead in areas where they feel they can make consensus based progress, POSIX.7 is now split into two documents called POSIX.7a (print queue administration) and POSIX.7b (software administration).

Recognizing the need for information describing existing practice in the area of network wide system management, the Open Software Foundation (OSF) solicited technologies from industry that could be integrated to simplify system management in heterogeneous computing environments. In October, 1991, OSF announced that they had chosen Hewlett Packard's Software Distribution Utilities to provide the basis for the OSF Distributed Management Environment (DME). The current draft of POSIX.7b is a roughly one year old descendant of the External Specification that describes the HP Software Distribution Utilities.

The original HP implementation suggested an object orientation but it was not developed using a rigorous object oriented specification language. In one year of POSIX meetings the group has made significant progress in further defining the attributes of the managed objects, but the specification is still incomplete and at times ambiguous. There is much discussion concerning object behavior.

Open issues include the question of allowing multiple Management Information Bases (MIB), and which attributes of a software object can be used, and how they are used as a selection mechanism.

Although invention by a standards committee is not advisable, it seems unavoidable when the base design is incomplete for the purposes of the standard.

Several decisions regarding general content were finalized. There will be no API included in the standard. An informative annex which provides information on how one implementation communicates between the manager, source, and tar-

get roles will be included. A rationale section which informs the reader as to the intent and history of the standard will also be included.

The serial media format was previously specified as `tar`, but will now be specified as being readable and writable by `pax` (POSIX.2-1992). Locking mechanisms are considered to be an implementation detail and outside the scope of the standard. A command line option will be provided to permit interaction sufficient to handle multi-volume media.

The group discussed rewriting part of the document using the ISO Guidelines for the Definition of Managed Objects (GDMO). The process of rewriting using GDMO would have the beneficial side effect of highlighting inconsistencies, omissions, and redundancies. In fact, it was advised that the draft would not be adopted by ISO unless GDMO was used.

The active participants did not embrace the idea wholeheartedly because a drastic structure change could further delay the balloting schedule. Mock ballot is planned to occur after the January meeting. Budget constraints may impose a time limit on the standards activity, and active participants fear having the POSIX.7b standards activity permanently interrupted before going to ballot. Refinement of the existing object definitions and behaviors continues at a fast pace.

Report on POSIX.14: Multiprocessor Profile

Rick Greer <rick@ivy.isc.com> reports on the October 19-23, 1992 meeting in Utrecht, NL

The big news in the POSIX.14 working group is that we have inherited the POSIX.18 draft from Donn Terry and are now responsible for seeing it through balloting. POSIX.18 is the Platform Environment Profile, more commonly known as a profile to describe the traditional multi-user Unix platform.

Having been assured that the POSIX.18 document was "practically ready for balloting," we traded POSIX.14's March 1993 balloting slot to POSIX.18. Remember that this year there are so many documents in ballot that a strict timetable is being used to control the potential administrative overload. Our document's ballot slot had been allocated as a purely defensive measure anyway -- see below. We also decided to keep the balloting group open right up to the last minute, so those interested in paying \$25.00 for the privilege of complaining may still do so. [Ed.-- This may be raised to \$50.00 in the new year!]

We made one major change to the POSIX.18 draft: The C language feature is now required. It had been optional. Our reasoning for this was two-fold. First, we realized that because there was no requirement that a given implementation provide a specific language feature, people could write POSIX.18 compliant applications that would not run on POSIX.18 compliant implementations! By requiring C at a minimum, vendors can guarantee portability of other languages, in particular FORTRAN and ADA, to all POSIX.18 compliant implementations by writing their runtime libraries in C.

Secondly, given that POSIX.18 is supposed to codify "classic UNIX," and since classic UNIX has always included a C compiler, albeit the "classic" K&R compiler, not `c89`, we felt it appropriate to require C language support in POSIX.18.

The working group also made a number of minor editorial changes to the document, mostly removing redundant text, which brought it down to less than half its original size!

As for POSIX.14's real purpose, the POSIX multiprocessor profile, we decided not to ballot the current draft after all. We had originally decided to put POSIX.14 out to ballot in March in an attempt to be in ballot by the time the Profile Steering Committee (PSC) finalized its rules for "Standard Posix Profiles." We reasoned that if profile groups that were in ballot at the time the rules were adopted were grandfathered in such a way as to allow them to ignore said rules, POSIX.14 might be the only profile to which the rules applied. This seemed a bit unfair.

It now appears, however, that all profiles will have to follow the PSC rules before they can come out of ballot.

So we're back to proposing new MP interfaces for POSIX.1 and POSIX.2 that would fill various semantic gaps in MP systems that will be noted in the POSIX.14 draft. This includes describing parallel behavior for a number of common utilities (e.g., `make`, `find`, `grep`, `xargs`,) as well as describing special MP features of system administration functions such as `ps(1)` and `times(2)`. We also continue to argue about processor binding: can we specify enough of this in an architecture-independent manner to make it worthwhile?

One interesting point made at the October meeting was that many of the participants in our working group feel that our major contribution will not be the MP profile, so much as our monitoring of other POSIX work to make sure that any new interfaces do not cause major headaches for

MP implementations (e.g., the work that we've already done with respect to pthreads). With this in mind, we have proposed a new name for the group: POSIX.14 – the POSIX reentrancy police!

Report on POSIX.17 - Directory Services API

Mark Hazzard <markh@rsvl.unisys.com> reports on the October 19-23, 1992 meeting in Utrecht, the Netherlands

Summary

A recirculation ballot of Draft 4.0 of POSIX.17 completed just prior to the Utrecht meeting and the group met primarily as a ballot resolution team. All but one of the outstanding comments and objections were resolved.

The next draft (Draft 5.0) will contain editorial changes and two minor technical changes. The changes will require another recirculation ballot. Only the pages affected by the technical changes will be distributed and can be balloted upon.

We expect to produce a Draft 5.0, do the "mini" recirculation, process and incorporate changes (if any) in time for the March 1993 IEEE RevCom meeting. Given this schedule, you can expect publication of our approved specifications in the middle of 1993.

The US TAG to ISO/IEC JTC1 has stated their intention to forward our specification to ISO for fast tracking (direct ISO ballot) when approved as an ANSI/IEEE standard.

Introduction

The POSIX.17 group has generated and is currently balloting a user to directory services API (e.g., API to an X.500 DUA – Directory User Agent). We used APIA – X/Open's XDS specification as a basis for work. XDS is included in XPG4 and has been adopted as part of both OSF's DCE and UI's Atlas.

XDS is an object oriented interface and requires a companion specification (XOM) for object management. XOM is a stand-alone specification with general applicability beyond the API to directory services. It will be used by IEEE 1224.1 – X.400 API (and possibly other POSIX groups) and is being standardized by POSIX/TCOS as P1224. A draft of P1224 is already in ballot.

POSIX.17 is one of five "networking" groups that currently make up the IEEE TCOS/POSIX Distributed Services and as such, POSIX.17 comes

under the purview of the Distributed Services Steering Committee (DSSC). --

Status

Draft 4.0 of POSIX.17, which included all the technical, editorial, and format changes identified in the July Chicago meeting, completed a recirculation ballot prior to the Utrecht meeting. POSIX.17 was recirculated as four separate specifications:

- P1224.2 Directory Services API– Language Independent Specification
- P1326.2 Test Methods for P1224.2
- P1327.2 C Language Binding for P1224.2
- P1328.2 Test Methods for P1327.2

NOTE: During a special ad hoc meeting of the US TAG to JTC1, POSIX.17 was one of three TCOS APIs recommended for fast track to ISO. In order to accommodate the ISO format, POSIX.17 was required to be split into four separate parts (documents), hence the four specifications.

The group spent a majority of the meeting processing the results of that ballot and planning for another "mini" recirculation and final submission to IEEE RevCom for approval. Most of the comments were editorial in nature. However, two minor technical corrections were suggested and accepted by the committee, which (in the opinion of the IEEE) will require another (mini) recirculation.

All but one of the outstanding comments and objections were resolved for Draft 4.0. These results exceed the level of consensus (75%) required by the IEEE for approval as a standard and we don't expect much change in Draft 5.0. We plan to complete this recirculation ballot, clean up the draft and submit it to IEEE RevCom for final approval in time for their March 1993 quarterly review meeting. Based on this schedule, I would expect to see it approved and published by the IEEE mid-year 1993.

It is still my understanding that when P1224.2 and P1327.2 are approved by the IEEE, the US TAG to ISO/IEC JTC1 will propose that they be accepted by ISO as Draft International Standards (DIS) and balloted directly (fast tracked).

In Closing ...

There's quite a bit of work remaining, such as coordinating the recirculation and wrapping up loose ends for submission to IEEE RevCom. The group is not planning to meet in New Orleans in January.

Report on The Distributed Security Study Group

Dave Rogers <drogers@datlog.co.uk> reports on the October 19-23, 1992 meeting in Utrecht, NL

The POSIX Distributed Security Study Group (DSSG) met for the third and last time in Utrecht. This is the end of the six month lifetime of the study group. The group continued to be well supported and the Utrecht meeting brought a few new faces into the group, particularly European, but also a Canadian.

The DSSG made progress with the approach of defining a security framework for POSIX by mapping the ECMA "Open Systems Security - A Security Framework" onto a POSIX environment with encouraging results. The draft framework produced has been used to make an initial identification of the services requiring Application Program Interfaces and has mapped known existing or emerging implementations onto the APIs identified. Other standards activities in this area have also been identified.

A white paper titled "A Distributed Security Framework For POSIX" has been published presenting the work done to date with the specific objective of stimulating discussion and comment.

The DSSG has recommended the formation of a new POSIX working group to produce a "Guide to Security within Distributed POSIX Systems" using the white paper produced by the DSSG as the base document. A project authorization request (PAR) for this work has been submitted and will be considered by the POSIX SEC at the January meeting. An objective of this Guide is to produce a definition of the security services and APIs required throughout POSIX so that the adequacy of future PARs on meeting the defined security requirements can be assessed.

If anyone is interested in obtaining a copy of the white paper or wants more information then contact the DSSG Chair:

David Rogers
+44 81-863-0383 or
+44 256-59222 x4083
Data Logic Ltd
Queens House
Kymbereley Road
Harrow,
Middx HA1 1YD
UK
email: drogers@datlog.co.uk

Report on IEEE Standards Board

Mary Lynne Nielsen <m.nielsen@ieee.org> reports on the September, 1992 meeting in New York, NY

September's meeting was unusually busy for TCOS, with lots of new project authorization requests (PARs) due to mirving activities and, at last, approval of one of the key components of POSIX. Decisions in the area of JTC1 funding will also have an impact on TCOS work.

[Ed. - TCOS is the technical committee within the IEEE responsible for developing the POSIX standards.]

At Long Last...

The IEEE Standards Board Review Committee (RevCom) approved P1003.2 and P1003.2a as IEEE standards at this meeting. POSIX.2 covers the shell and utilities for a POSIX system, while 1003.2a covers the user portability extensions for the shell. This pile of over 1300 pages of material is now in the publication process and should be available in the spring of 1993. Congratulations to all involved!

Note to any who actively work with this committee: as of the March 1993 meeting, only the new RevCom submittal forms will be accepted. Make sure you're using the form dated 9/92.

NesCom Actions Everywhere

The IEEE Standards Board New Standards Committee (NesCom) dealt with a whopping 14 Project Authorization Requests (PARs) from TCOS at this meeting. Twelve of these 14 came from the mirving, or splitting up, of three existing TCOS projects. Why did that have to happen? Well, mostly from a lot of resolutions either from various committees of ISO/IEC JTC1 (all of which basically translates to "the international group working on TCOS projects") or from TCOS itself. These resolutions say that it's better to have this work, have it all completed at the same time, and have it in bite-sized, somewhat digestible chunks, rather than receiving one huge document that takes a great deal of time to prepare. (For example, POSIX.2 was in ballot for three years).

What that means is that the various PARs in TCOS will often have to split into at least two and usually four parts: a base standard that is language independent, its test methods, a related language binding (usually C at first), and its test methods. While there is some debate as to the merits of this method, this practice is now being put into force in TCOS.

The first documents to be produced in this manner will be the 1224 series of standards (which is now the 1224, 1326, 1327, and 1328 standards). There is a strong indication that these standards will make the March 1993 RevCom meeting for approval, and the PARs for their mirving were approved at this meeting.

Also approved was a PAR for the revision of IEEE Std 1003.3-1991, the base standard on how to describe test methods for POSIX. Due to the expansion of testing to all TCOS (not just POSIX) standards and the need for test methods for new types of documents like profiles, the committee felt that it was time to start work on a revision of this standard.

In an attempt to control the bewildering expansion of 'dot' projects, a new numbering system will be employed for the POSIX testing standards. They will be numbered 2003.x, in parallel with the base standard they are testing. This revision is therefore numbered P2003.

Finally, P1003.19 was finally approved at this meeting, when NesCom at last received the reassurances they wanted that this work was not an infringement on the X3 work on the Fortran language itself. As such, the PAR for work on a Fortran 90 binding to POSIX.1 has at last gained clearance to go ahead.

Is It TransCom. . . or Isn't It?

TransCom, the IEEE Standards Board Transnational Committee, has voted to changed its name to IntCom, the IEEE Standards Board International Committee, an action that was also approved by the Board. It seems that the term "transnational," while used in the IEEE bylaws to define the scope of the IEEE, is very confusing to the members of this committee and to the people they speak to about their work. (My understanding is that the term means "without borders.") They feel that the word "international" far better suits the activities they undertake, which is to coordinate IEEE standards activities with non-US standards organizations.

In addition, Trans/IntCom continued to work on a guide for synchronizing work with ISO/IEC JTC1, a plan that recognizes the methods used by POSIX to move its standards forward in this arena. This guide should hopefully be approved in December.

IT Funding

As mentioned in earlier snitch reports, the Standards Board has been wrestling with an action

from ANSI that proposes having the groups involved in JTC1 activities support the secretariats of JTC1 that ANSI maintains. The IEEE Standards Board, representing one of the major groups involved, created an ad hoc committee to explore resolutions to this issue. TCOS supplied information to this committee in the form of a resolution expressing their position, while the committee examined the financial and legal aspects of this question. They also examined if this funding conflicted with the expressed goals of the IEEE Standards Board Strategic Objectives.

The committee submitted its final report at this Board meeting. In it, they felt that these funds could be collected without any negative impact on the legal aspects, financial aspects, or stated objectives of the IEEE Standards Board. The report recommended that IEEE staff work with the standards committees in designing and implementing procedures for the collection and administration of participation fees assessed to IEEE participants for these secretariats. The report also stated that each standards committee should decide on its own procedures for fund collection, but they should be encouraged very strongly to standardize on one or two methods for collecting fees.

One note on this: TCOS discussed this situation at its October meeting in Utrecht, and the following methods for collecting funds were approved by the TCOS Standards Executive Committee (SEC): an increase in balloting fees; an increase in NAPS mailing costs; a reduction in meeting services (such as Friday lunches); and a fee imposition for meetings held independently of the regular TCOS meetings. It was felt that this system would distribute the burden of raising these funds equitably among those who attend meetings and those who do not but who participate in the process through mailings.

Awards and Recognition

Three TCOS members received awards from the IEEE Standards Board, called IEEE Standards Medallions, in recognition of their contributions to standards development. They are Donn Terry, the former chair of POSIX.1, Hal Jespersen, the chair of POSIX.2, and Roger Martin, the chair of the TCOS Steering Committee on Conformance Testing (SCCT) and the former chair of the POSIX.3 (Test Methods) working group. Congratulations to them all.

NesCom Approvals

New PARs

P1003.19 Standard for Information Technology – POSIX Fortran 90 Language Interfaces – Part 1: Binding for System Application Program Interface (API)

P2003 Standard for Information Technology – Test Methods for Measuring Conformance to POSIX

Revised PARs

P1224 Standard for Information Technology – Open Systems Interconnection (OSI) Abstract Data Manipulation – Application Program Interface (API) [Language Independent]

P1224.1 Standard for Information Technology – X.400 Based Electronic Messaging Application Program Interfaces (APIs) [Language Independent]

P1224.2 Standard for Information Technology – Directory Services Application Program Interface (API) – Language Independent Specification

P1326 Standard for Information Technology – Test Methods for Measuring Conformance to Open Systems Interconnection (OSI) Abstract Data Manipulation – Application Program Interface (API) [Language Independent]

P1326.1 Standard for Information Technology – Test Methods for Measuring Conformance to X.400 Based Electronic Messaging Application Program Interface (API) [Language Independent]

P1326.2 Standard for Information Technology – Test Methods for Directory Services Application Program Interface (API) – Language Independent Specification

P1327 Standard for Information Technology – Open Systems Interconnection (OSI) Abstract Data Manipulation C Language Interfaces – Binding for Application Program Interface (API)

P1327.1 Standard for Information Technology – X.400 Based Electronic Messaging C Language Interfaces-Binding for Application Program Interface (API)

P1327.2 Standard for Information Technology – Directory Services Application Program Interface (API) – C Language Specification

P1328 Standard for Information Technology – Test Methods for Measuring Conformance to Open Systems Interconnection (OSI) Abstract

Data Manipulation C Language Interfaces – Binding for Application Program Interface (API) --

P1328.1 Standard for Information Technology – Test Methods for Measuring Conformance to X.400 Based Electronic Messaging C Language Interfaces – Binding for Application Program Interface (API)

P1328.2 Standard for Information Technology – Test Methods for Directory Services Application Program Interface (API) – C Language Specification

RevCom Approvals

P1003.2 Standard for Information Technology – Portable Operating System Interface (POSIX) – Part 2: Shell and Utilities

P1003.2a Standard for Information Technology – Portable Operating System Interface (POSIX) – Part 2: Shell and Utilities, User Portability Extension

Report on ISO WG15 (POSIX) Rapporteur Group on Co-ordination of Profile Activities

Kevin Lewis <klewis@gucci.enet.dec.com> reports on the October 23-24, 1992 meeting in Utrecht, NL

The IEEE Technical Committee on Operating Systems – Standards Subcommittee (TCOS-SS) forwards POSIX documents through an ANSI technical advisory group to ISO Working Group 15 (WG15) for approval as international standards. WG15 has a number of rapporteur groups, which are small groups of experts on various ISO POSIX related topics.

This was the third meeting of the Rapporteur Group on Coordination of Profile Activities (RGCPA). It was my first. The meeting lasted a day and a half. There were actually more observers in the room than members. About 15-18 people attended, of which 75% were IEEE POSIX attendees. Seeing all the familiar faces from a week of IEEE POSIX meetings underscored the high percentage of overlap between the IEEE and ISO POSIX working groups.

The work of this Rapporteur group is to co-ordinate profiling activities that would be of interest to WG15 as follows:

- the process of addressing user requirements for profile harmonization,
- the development of the appropriate approach to sub-setting WG15 standards within profiles,

- the treatment within profiles of the options that exist within a standard that is part of the profile.

Recognizing that there are other organizations dealing with profile issues, the group put forward a resolution to WG15 that the TCOS Profile Steering Committee and X/Open be encouraged to establish Category C liaison with WG15 RGCPA.

Reflecting back on this meeting, it seemed to me that the real purpose of this group is to serve as a radar, seeking out any and all profile activities anywhere in the globe that would be pertinent to the work of WG15 and SC22. From my own vantage point, it appeared to be accomplishing his purpose.

The next meeting of this group will be on May 10-11, 1993 in Heidelberg, Germany.

The Elusive JTC1

John Hill <hill@prc.unisys.com>

Quite often in reading articles concerning standards for information technology the term "JTC1" is encountered. This article defines the term, describes its activities, and puts JTC1 in context.

Until late 1988 there were multiple confusing processes for developing worldwide standards for information technology. Some standards, such as those for equipment and electrotechnical matters, were developed by the IEC. IEC is the acronym for the French equivalent of the "International Electro-technical Commission." Other standards, such as those for media and programming languages, were developed under the auspices of ISO. ISO is the commonly used name for the French "international standards organization."

The source of the confusion about ISO and IEC was largely at the detailed level of standards development, and stemmed from the fact that there was overlap of the work of the two organizations.

In the middle 1980s, thanks largely to the efforts of Ed Lohse, late of Burroughs Corporation, activities to rationalize the situation were started in earnest. The product of these activities is the ISO/IEC Joint Technical Committee 1, or JTC1.

JTC1 is the first, and currently the only, technical committee that is jointly managed by ISO and IEC. Devising the scheme for joint management of JTC1 was a formidable task. Here were two organizations whose generalized aims were similar

but operated in dissimilar fashions in key procedural areas. --

The situation was sufficiently complex that they decided that separate procedures for JTC1 were to be developed and approved. This document is known as the JTC1 Directives. (The JTC1 Directives can be obtained from the JTC1 secretariat, ANSI.)

So much for the framework. Now for the current organization and program of work of JTC1 and its subgroups.

First, you must understand that the members of JTC1 are referred to as member bodies. There are two types of member bodies: national bodies, and liaisons. There are 42 national member bodies. (24 are primary, and 18 are observer). As an example, the USA, as represented by ANSI, is a national body member of JTC1. There are others, including France (AFNOR), Germany (DIN), and Sweden (SII).

The matter of liaison members is a bit more complicated. There are 14 internal liaisons. These are subgroups of ISO or IEC that have interest in the work of JTC1. There are also 19 external liaisons. ECMA, the European Computer Manufacturers Association, is a representative example of a liaison member of JTC1. One interesting sidelight to this is that most nations have some sort of umbrella-like standards organization that can be designated as the country's representative in JTC1. These national umbrella standards organizations operate within their own countries according to their own rules and procedures. JTC1, while insulated from member countries' internal operations, is nonetheless aware of them.

So the membership of JTC1 is either national (i.e., by country) or notified liaison. There is no concept of "organizational" or corporate membership. Similarly, there are no individual members. Many national bodies operate internally on the basis of organizational membership. Some operate on the basis of individual membership. The umbrella organization in the USA, ANSI, accredits organizations and committees to develop standards for it. Membership in some of these is organizational, such as X3. In some it is individual, such as the IEEE, the Institute of Electrical and Electronic Engineers.

For the most part the work of JTC1 itself is managerial in nature. JTC1 focuses on matters like:

- project initiation,
- subgroup establishment (and disposal),
- document approval.

The technical work of JTC1 is really accomplished by its subgroups. Broadly speaking, there are three types of JTC1 subgroup. These are special working groups (SWG), study groups (SG), and subcommittees (SC).

SWGs are typically established to perform some specific task and are often non-technical in nature. Examples include the SWG-P that deals with JTC1 procedures, and SWG-FS that deals with functional standards (often called international standardized profiles, or ISPs). [*Ed.- SWG-FS is sometimes referred to simply as SGFS.*] SWG-FS has developed Technical Report (TR) 10000 that describes procedures for the development of open systems interconnection (OSI) ISPs. SWG-FS is currently revising TR10000 in order that it incorporate procedures for managing the development of ISPs for open systems environments.

There have been two special study groups established by JTC1. Each was given a specific charter and assigned specific deliverables. Neither exists today since they completed their assignments. The two study groups were MSG-1 (management study group) and TSG-1 (technical study group). TSG-1 focused on interfaces for application portability.

The most enduring subgroup type is the subcommittee (SC). SCs tend to be organized around functional topics. An SC's typically focuses on a single technical subject area. The detailed standards development work of an SC takes place within the working groups (WG) within an SC.

One way to better grasp the activities of JTC1 is to group the SCs. There are four convenient groupings:

- application elements,
- systems,
- equipment and media,
- systems support.

A complete list of these SCs follows the article, grouped according to the above list.

The scope of JTC1 is extensive. Virtually all standards used in modern information technology systems receive their worldwide endorsement by JTC1. This has simply been an overview. There are a multitude of detailed projects that collectively specify the full depth of the technical program of JTC1.

ISO/IEC JTC1 Subcommittees:

Application Elements

SC1 (Vocabulary): To collect and coordinate the usage of terminology by all groups within JTC1. The Dictionary Group!

SC7 (Software Engineering): To define standardized tools to development software.

SC14 (Representation of Data Elements): To codify data elements such that their common definitions can be used to exchange data.

SC22 (Languages and Application Environments): Programming Language and Operating Environment standards.

Systems

SC6 (Telecommunications and Information Exchange): Standards for telecommunications and OSI, (systems functions, procedures and parameters, as well as the conditions for their use) for the four OSI layers that support the transport service. Done in effective cooperation with CCITT.

SC18 (Text and Office Systems): Standardization of functionality that simplifies text editing and other office related subjects.

SC21 (OSI Information Retrieval, Transfer and Management): Development of standards for the upper layers of the Open Systems Interconnection (OSI) model. Also included are database management systems, information resource management systems (IRDS), and open distributed processing standards (ODP).

SC26 (Microprocessor Systems): Development of standards used in microprocessor systems including basic hardware, bus and allied interfaces.

Equipment and Media

SC11 (Flexible Magnetic Media for Digital Data Interchange): Development of standards for diskettes and cartridges. The unrecorded (raw media) as well as the recording standards are both included.

SC15 (Labeling and File Structure): Standardization of file allocation and directory information used for all types of recorded media.

SC17 (Identification Cards and Related Devices): Standards for cards such as credit

and debit cards including the physical, electrical, and magnetic properties. Intelligent (IC) cards are also covered.

SC23 (Optical Digital Data Disks): Development of optical media standards including the unrecorded (raw) media as well as the recording onto and reading from those media. Both write once (WORM) and rewritable media are included.

SC28 (Office Equipment): Standardization of equipment commonly used in office settings. This includes printers and the quality of their output.

Systems Support

SC2 (Character Sets and Information Coding): Standards for the bit and byte coded representation of elements of various identified types of information, for interchange mainly at the application level, i.e., all aspects of sets of graphic and control characters.

sC27 (Security Techniques): Development of standards for security, such as encryption and verification.

SC29 (Coded Representation of Picture, Audio and Multimedia/Hypermedia Information): Standardization of complex (i.e., more difficult than characters) data representation. Data compression without the loss of information is also handled here.

Unix Tricks 'n' Traps

This is a brief quiz on the security of your UNIX environment.

Question: Do you have any shell scripts that are *setuid* to root on any of your systems?

If you answered No, then you have passed the Quiz. If you answered "Yes" or "I don't know", then you have *failed*.

Bourne shell scripts that are *setuid* to root are one of the biggest security holes in UNIX. This is because there are at least three well-known ways that one of these scripts can be coerced into *executing any arbitrary commands as the superuser*.

Some of these holes have been fixed in later versions of UNIX. However, it is very likely that at least one of them has not been fixed on *your* version. It is also so easy to avoid the problem in the first place that there is no reason to take the risk.

The best known ways of compromising shell scripts rely on the very fact that they *are* shell scripts - they are interpreted by */bin/sh* at the time the script is run. I don't intend to give any lessons on how you can get *setuid* root shell scripts to execute anything you want. Instead, I will demonstrate a simple framework for rewriting them in C, in a fashion that greatly enhances their security (because the shell is no longer involved).

A common example of a *setuid* root shell script is one that mounts a DOS floppy on a workstation (I see this a lot on Sun sites). This script usually contains something simple like:

```
#!/bin/sh
/etc/mount -t pcfs /dev/fd0 /pcfs
```

(The script may also do things like check that the mount point exists, that a *pcfs* filesystem mount appears in */etc/fstab*, and so on - but these are not important to our discussion).

We can close the security holes associated with shell scripts like this one by replacing it with the executable generated from a simple C program.

Essentially all the C program has to do is execute the *mount* command directly, instead of having the shell do it. This precludes the use of the *system()* library call - *system()* starts a shell and passes the command line to it! Instead, we use one of the *exec()* family to run *mount*.

I have included the code for a simple C program to mount a DOS floppy on a Sun. For space reasons, this program leaves all error checking to the *mount* program itself. It should be fairly simple to see what the C program does, and also to see how to translate a shell script like the one above into a similar C program.

Some points worth making:

- We do *not* use either *execvp* or *execlp* to invoke *mount* - both of these calls search the PATH for the program. A malicious user could set their PATH to their own *bin* directory and put any commands of their choice in a file called *mount*, which would then be executed by our "secure" version. Instead, we use another variant that invokes */etc/mount* directly.
- There is a very, very dumb feature of *mount* in SunOS that if the filesystem to be mounted is not a normal UNIX ("4.2") filesystem, it invokes another program (*mount_pcfs* in this case) to perform the mount - by searching the user's PATH for this program (which we explicitly said not to do in the previous point)! To prevent this hole from being exploited, we reset the PATH to only include the directory that contains these *mount* variants before we invoke */etc/mount*.
- We should install the executable with an owner of *root* and 4711 (*setuid*, and *rwX--X--X*) permissions. You do not need read access to an executable binary to be able to execute it, and why give away more access than is


```

needed?
#define      MOUNTPOINT  "/pcfs"

main()
{
    static char *mountenv[] = {
        "PATH=/usr/etc",
        NULL
    };

    /*
     *   Attempt the mount. We invoke mount with the full pathname
     *   and supply a restricted environment for security reasons.
     */
    execl("/etc/mount", "mount", "-t", "pcfs", "/dev/fd0", MOUNTPOINT,
        NULL, mountenv);

    /*
     *   If we get here, the execl failed. Report the error and abort.
     */
    perror("execl");
    exit(1);
}

```

Adrian Booth, Adrian Booth Computing Consultants <abcc@dialix.oz.au>, (09) 354 4936
From WAUG, the WA Chapter of AUUG

User Support Mailbox

Dear Unixsupport,

How can I sort a file in order of line length?

Frustrated With Awk.

Dear Frustrated With Awk,

Time to learn Perl!

```

#!/usr/bin/perl
# slurps all of input into an array, sorts it and prints it

sub lencomp
{
    return length($a) - length($b);
}

print sort lencomp <>;
exit 0;

```

Newsgroups: cs.unix

From: janet@cs.uwa.edu.au (Janet Jackson)

Subject: something to help you log in when your favourite NFS server is down

This is for those of you who want to have a directory from an unusual server, such as a subdirectory of /proj/robvis or a directory in your Cellar, in your search path and/or MANPATH.

As you may have noticed ;-) if you try to log in when the machine that serves your special directory is down, your session will hang. (This happens because the shell tries to look through all the directories in the PATH to set up a hash table for faster searching.) To avoid the hanging, you may like to change your .login or .cshrc file (whichever sets your PATH) to include a version of following code.

The idea is to add the special directory to the paths only if its server is up. The example uses /proj/robvis/ansi_vip/bin, served from mardo, so change it if you want something else.

If you're not using csh, you'll have to write the equivalent in whatever shell you do use.

Janet

```
# first set your path to include everything but the robvis directory

alias addpath 'setenv PATH "${PATH}:"; rehash'      # very handy!

set MY_SERVER = mardo          # in case it changes someday
set TIMEOUT = 2               # how many seconds ping will try for

/usr/etc/ping $MY_SERVER $TIMEOUT >/dev/null
if ($status == 0) then
    addpath /proj/robvis/ansi_vip/bin
    setenv MANPATH "${MANPATH}:/proj/robvis/ansi_vip/man"
else
    if ( $?prompt ) then      # this is an interactive shell
        echo "Warning: can't talk to $MY_SERVER"
    endif
endif
```

Janet Jackson <janet@cs.uwa.edu.au>
From WAUG, the WA Chapter of AUUG

AUUG MANAGEMENT COMMITTEE

SUMMARY OF MINUTES OF MEETING 18 February 1993

Present: Frank Crawford, Glenn Huxtable, Chris Maltby, Phil McCrea, John O'Brien, Michael Paddon, Greg Rose, Peter Wishart, and Liz Fraumann.

Guests: Wael Foda, Lachie Hill, Piers Lauder

Apologies from Rolf Jester

1. AUUG '93

The Programme Chair for AUUG '93 (Piers Lauder) reviewed the current projected programme and papers received to date (5 papers so far).

1.1. Tutorials

Tutorials: currently have at least 80% or more commitment from the following to present:

- (1) Rob Kolstad
- (2) Greg Rose - TCL/TK Windows
- (3) Tom Christenson - perl
- (4) Ian Hoyle - Internet

It was suggested that we solicit tutorials on DCE and TCP/IP.

1.2. Sponsorship

The following sponsors have been secured to date:

- (1) IBM Australia Conference Brochure
- (2) CMP Marketing Dinner
- (3) McDonnell Douglas IS Conference Folder
- (4) Cognos Pty. Ltd. Lapel Badges
- (5) Sun Microsystems Carry Bags

1.3. Public Relations

In March an awareness campaign will begin with an overview of AUUG. Many of the major speakers will be released. June - September will encompass features on most of the general session speakers. Diary entries will be posted on a regular basis from April onwards.

It was decided to secure a clipping service for 7 months to monitor the effectiveness of the PR and advertising. It will be used in AUUG '94 as a selling point and to provide a barometer as the conference draws near.

The top 5 publications as reported in the Roy Morgan study will be targeted in the publicity campaign. LF cautioned the group it will be an expensive endeavour, but reminded the group of the cuts last year which harmed the conference in the long run. LF will prepare a complete advertising budget and recommendations to the group. The campaign would commence in June/July time frame. It was also noted this campaign would need to work together with the public relations and features to maximise impact.

1.4. Budget

LF reviewed the overall conference and budget. Comparison information from the UniForum, GTE, and Open Software Symposium were brought forth to discuss the registration fees for AUUG '93. The following fees were approved:

- (1) Members \$350.
- (2) Non-Members \$495.
- (3) Day Members \$150.
- (4) Day Non-Members \$200.
- (5) Student \$100.
- (6) Late Fee (all above categories) \$70.
- (7) Special Event \$ 30. (Late fee \$20)

We will solicit "Early Bird" registration to assist with deposits. It was recommended that this be available to MEMBERS only, \$20 discount (\$330.) from 1 May to 31 May. Cheque Bankcard information MUST accompany registration. A registration form will be ready for distribution by 1 May.

1.5. Conference Events

Discussion on a GO/NOGO for the reception on 28 Sept. took place. Facility space and room to move around were greatest concern. If "spill" can take place into foyer proceed with Harbour View room at the conference centre. Wine and cheese platter will be served @\$20.pp.

Dinner was discussed and options of the Power House Museum (PHM) and the convention banquet hall reviewed. The flexibility of the PHM and "things to do," was a clear win with this facility. Dinner will consist of "substantial hors d'orves" and beer/wine/soda. Cost will be \$55@pp. Supplying our own wine was discussed, however, due to the \$10 per bottle corkage fee, it was determined a selection of 2 red and 2 white wines would be made from the catering list. Guest tickets for dinner would be made available for a \$55@ fee.

Partners Programme: Activities would be tours of the area (i.e. harbour cruise, Sydney Explorer, Blue Mountains, etc). Cost would be the participants only (no cost to AUUG).

NCR will be covering total costs for the special event in exchange for advertising in the conference registration brochure, and introduction of Cliff Stoll at the special event.

2. TREASURER'S REPORT

The Treasurer reported we should anticipate approximately a \$30,000 loss this year. He distributed the current and projected budget. There is funding in the bank. The deficit is due to AUUG '92 and deposits required for AUUG '93. This is the first year AUUG is having to pay the holding deposits for the conference due to the separation of the conference and exhibition.

3. CD Rom Exchange

GH shared the CD with the group from Prime Time Freeware. It was decided GH would do a survey of interest via email. It was felt the local chapters should hold the CDs for distribution to members.

4. Election Procedures

The Returning Officer submitted an updated version of the procedures. Procedures to be distributed with call for nominations.

5. Non Financial Members

The Secretariat advised that we had 165 non-financial members. Primarily institutional members. It was felt by the group this was likely due to industry movement in personnel. PL offered to send email to the listing. LF is to draft letter and PL will distribute via email. All non-financial members will be cut from listing by 31 March '93.

6. Summer Conferences

Summer Conference attendance is generally up from last year. Tas. - 60+, Darwin - 80, Canberra - 160 (others not held yet). There had been a press release for Darwin, as per their request. It seemed to help very much with the attendance. GH asked the same be done for WA.

7. AUUGN Advertising Rates

AUUGN Advertising rates have been established by FC/JC. FC reported the following:

- (1) Full Page - \$300 Back Cover - \$750
- (2) Half Page - \$180 Quarter Pg. - \$120

8. Next Meeting

23rd April 1993 at Softway.

AUUG Membership Categories

Once again a reminder for all "members" of AUUG to check that you are, in fact, a member, and that you still will be for the next two months.

There are 4 membership types, plus a newsletter subscription, any of which might be just right for you.

The membership categories are:

Institutional Member
Ordinary Member
Student Member
Honorary Life Member

Institutional memberships are primarily intended for university departments, companies, etc. This is a voting membership (one vote), which receives two copies of the newsletter. Institutional members can also delegate 2 representatives to attend AUUG meetings at members rates. AUUG is also keeping track of the licence status of institutional members. If, at some future date, we are able to offer a software tape distribution service, this would be available only to institutional members, whose relevant licences can be verified.

If your institution is not an institutional member, isn't it about time it became one?

Ordinary memberships are for individuals. This is also a voting membership (one vote), which receives a single copy of the newsletter. A primary difference from Institutional Membership is that the benefits of Ordinary Membership apply to the named member only. That is, only the member can obtain discounts an attendance at AUUG meetings, etc. Sending a representative isn't permitted.

Are you an AUUG member?

Student Memberships are for full time students at recognised academic institutions. This is a non voting membership which receives a single copy of the newsletter. Otherwise the benefits are as for Ordinary Members.

Honorary Life Membership is not a membership you can apply for, you must be elected to it. What's more, you must have been a member for at least 5 years before being elected.

It's also possible to subscribe to the newsletter without being an AUUG member. This saves you nothing financially, that is, the subscription price is greater than the membership dues. However, it might be appropriate for libraries, etc, which simply want copies of AUUGN to help fill their shelves, and have no actual interest in the contents, or the association.

Subscriptions are also available to members who have a need for more copies of AUUGN than their membership provides.

To find out your membership type, examine your membership card or the mailing label of this AUUGN. Both of these contain information about your current membership status. The first letter is your membership type code, M for regular members, S for students, and I for institutions, or R for newsletter subscription. Membership falls due in January or July, as appropriate. You will be invoiced prior to the expiry of your membership.

Check that your membership isn't about to expire and always keep your address up-to-date. Ask your colleagues if they received this issue of AUUGN, tell them that if not, it probably means that their membership has lapsed, or perhaps, they were never a member at all! Feel free to copy the membership forms, give one to everyone that you know.

If you want to join AUUG, or renew your membership, you will find forms in this issue of AUUGN. Send the appropriate form (with remittance) to the address indicated on it, and your membership will (re-)commence.

As a service to members, AUUG has arranged to accept payments via credit card. You can use your Bankcard (within Australia only), or your Visa or Mastercard by simply completing the authorisation on the application form.

AUUG Incorporated

Application for Institutional Membership

AUUG Inc.

To apply for institutional membership of the AUUG, complete this form, and return it with payment in Australian Dollars, or credit card authorisation, to:

AUUG Membership Secretary
 P O Box 366
 Kensington NSW 2033
 Australia

• Foreign applicants please send a bank draft drawn on an Australian bank, or credit card authorisation, and remember to select either surface or air mail.

This form is valid only until 31st May, 1994

..... does hereby apply for

- New/Renewal* Institutional Membership of AUUG \$350.00
- International Surface Mail \$ 40.00
- International Air Mail \$120.00

Total remitted AUD\$ _____
 (cheque, money order, credit card)

* Delete one.

I/We agree that this membership will be subject to the rules and by-laws of the AUUG as in force from time to time, and that this membership will run for 12 consecutive months and becomes renewable on the following January or July, as appropriate.

I/We understand that I/we will receive two copies of the AUUG newsletter, and may send two representatives to AUUG sponsored events at member rates, though I/we will have only one vote in AUUG elections, and other ballots as required.

Date: ___ / ___ / ___ Signed: _____
 Title: _____

Tick this box if you wish your name & address withheld from mailing lists made available to vendors.

For our mailing database - please type or print clearly:

Administrative contact, and formal representative:

Name: Phone: (bh)
 Address: (ah)

 Net Address:

 Write "Unchanged" if details have not altered and this is a renewal.

Please charge \$ _____ to my/our Bankcard Visa Mastercard.
 Account number: _____ Expiry date: ___ / ___ .
 Name on card: _____ Signed: _____

Office use only: Please complete the other side.
 Chq: bank _____ bsb _____ - a/c _____ # _____
 Date: ___ / ___ / ___ \$ _____ CC type _____ V# _____
 Who: _____ Member# _____

Please send newsletters to the following addresses:

Name: Phone: (bh)
Address: (ah)
.....
..... Net Address:
.....
.....

Name: Phone: (bh)
Address: (ah)
.....
..... Net Address:
.....
.....

Write "unchanged" if this is a renewal, and details are not to be altered.

Please indicate which Unix licences you hold, and include copies of the title and signature pages of each, if these have not been sent previously.

Note: Recent licences usually revoke earlier ones, please indicate only licences which are current, and indicate any which have been revoked since your last membership form was submitted.

Note: Most binary licensees will have a System III or System V (of one variant or another) binary licence, even if the system supplied by your vendor is based upon V7 or 4BSD. There is no such thing as a BSD binary licence, and V7 binary licences were very rare, and expensive.

- | | |
|----------------------------------------------------------------|--------------------------------------------|
| <input type="checkbox"/> System V.3 source | <input type="checkbox"/> System V.3 binary |
| <input type="checkbox"/> System V.2 source | <input type="checkbox"/> System V.2 binary |
| <input type="checkbox"/> System V source | <input type="checkbox"/> System V binary |
| <input type="checkbox"/> System III source | <input type="checkbox"/> System III binary |
| <input type="checkbox"/> 4.2 or 4.3 BSD source | |
| <input type="checkbox"/> 4.1 BSD source | |
| <input type="checkbox"/> V7 source | |
| <input type="checkbox"/> Other (<i>Indicate which</i>) | |

AUUG Incorporated

Application for Ordinary, or Student, Membership

AUUG Inc.

To apply for membership of the AUUG, complete this form, and return it with payment in Australian Dollars, or credit card authorisation, to:

AUUG Membership Secretary
P O Box 366
Kensington NSW 2033
Australia

- Please don't send purchase orders — perhaps your purchasing department will consider this form to be an invoice.
- Foreign applicants please send a bank draft drawn on an Australian bank, or credit card authorisation, and remember to select either surface or air mail.

This form is valid only until 31st May, 1994

I, do hereby apply for

- Renewal/New* Membership of the AUUG \$90.00
- Renewal/New* Student Membership \$25.00 (note certification on other side)
- International Surface Mail \$20.00
- International Air Mail \$60.00 (note local zone rate available)

Total remitted AUD\$ _____
(cheque, money order, credit card)

* Delete one.

I agree that this membership will be subject to the rules and by-laws of the AUUG as in force from time to time, and that this membership will run for 12 consecutive months and becomes renewable on the following January or July, as appropriate.

Date: ___ / ___ / ___ Signed: _____

Tick this box if you wish your name & address withheld from mailing lists made available to vendors.

For our mailing database - please type or print clearly:

Name: Phone: (bh)

Address: (ah)

.....

..... Net Address:

.....

..... Write "Unchanged" if details have not altered and this is a renewal.

.....

Please charge \$_____ to my Bankcard Visa Mastercard.

Account number: _____ Expiry date: ___ / ___ .

Name on card: _____ Signed: _____

Office use only:

Chq: bank _____ bsb _____ - _____ a/c _____ # _____

Date: ___ / ___ / ___ \$ _____ CC type ___ V# _____

Who: _____ Member# _____

Student Member Certification *(to be completed by a member of the academic staff)*

I, certify that
..... *(name)*
is a full time student at *(institution)*
and is expected to graduate approximately ___ / ___ / ___.

Title: _____

Signature: _____

AUUG Incorporated

Application for Newsletter Subscription

AUUG Inc.

Non members who wish to apply for a subscription to the Australian UNIX systems User Group Newsletter, or members who desire additional subscriptions, should complete this form and return it to:

AUUG Membership Secretary
 PO Box 366
 Kensington NSW 2033
 Australia

- Please don't send purchase orders — perhaps your purchasing department will consider this form to be an invoice.
- Foreign applicants please send a bank draft drawn on an Australian bank, or credit card authorisation, and remember to select either surface or air mail.
- Use multiple copies of this form if copies of AUUGN are to be dispatched to differing addresses.

This form is valid only until 31st May, 1994

Please *enter / renew* my subscription for the Australian UNIX systems User Group Newsletter, as follows:

Name: Phone: (bh)
 Address: (ah)

 Net Address:

 Write "Unchanged" if address has
 not altered and this is a renewal.

For each copy requested, I enclose:

- Subscription to AUUGN \$ 90.00
- International Surface Mail \$ 20.00
- International Air Mail \$ 60.00

Copies requested (to above address) _____

Total remitted AUD\$ _____

(cheque, money order, credit card)

Tick this box if you wish your name & address withheld from mailing lists made available to vendors.

Please charge \$ _____ to my Bankcard Visa Mastercard.

Account number: _____ Expiry date: ____ / ____ .

Name on card: _____ Signed: _____

Office use only:

Chq: bank _____ bsb _____ - a/c _____ # _____

Date: ____ / ____ / ____ \$ _____ CC type ____ V# _____

Who: _____ Subscr# _____

AUUG

Notification of Change of Address AUUG Inc.

If you have changed your mailing address, please complete this form, and return it to:

AUUG Membership Secretary Fax: (02) 332 4066
PO Box 366
Kensington NSW 2033
Australia

Please allow at least 4 weeks for the change of address to take effect.

Old address (or attach a mailing label)

Name: Phone: (bh)
Address: (ah)
.....
..... Net Address:
.....
.....

New address (leave unaltered details blank)

Name: Phone: (bh)
Address: (ah)
.....
..... Net Address:
.....
.....

Office use only:

Date: ___ / ___ / ___

Who: _____

Memb# _____