ISSN 1035-7521

AUUGN

AUUG Inc. Newsletter

Volume 15, Number 2

April 1994

Registered by Australia Post, Publication Number NBG6524

The AUUG Incorporated Newsletter

Volume 15 Number 2

April 1994

CONTENTS

AUUG General Information . . . 3

Editorial . . . 5

AUUG President's Page . . . 6

AUUG Corporate Sponsors - A New Level of Participation . . . 7

AUUG Institutional Members . . . 8

Letters to the Editor . . . 11

Announcements

- AUUG Appoints New Business Manager . . . 15
- AUUG Freephone Number . . . 16
- McKusick does Oz T-shirt . . . 17
- Call for Articles for the Australian . . . 18
- AUUG & Zircon Systems Reach Agreement . . . 19
- AUUG & the Express Book Store Reach Agreement . . . 20

AUUG Local Chapters

- AUUG Regional Contacts . . . 21
- AUUG Inc. - Victorian Chapter . . . 22
- Update on AUUG Inc. - Victorian Chapter Activities *Stephen Prince* . . . 23
- From the Western Front *Janet Jackson* . . . 24
- WAUG - Meeting Reviews *Adrian Booth* . . . 25
- 1994 Perth AUUG Summer Conference Overview *Adrian Booth* . . . 26
- Impressions of the 1994 Perth AUUG Summer Conference *Janet Jackson* . . . 28
- AUUG NSW Chapter *Julian Dryden* . . . 30

AARNet Mail Affiliation . . . 31

Book Reviews . . . 35

- Prentice Hall Book Order Form . . . 41
- WoodsLane Information . . . 42

Open System Publications . . . 43

!AUUGN - from AUUGN Volume 1, Number 1

- List of Licensed Subscribers . . . 44

The Electronic Interviews *Adrian Booth* . . . 45

Technical Reports

What's New in the 4.4BSD User Code . . . . *Kirk McKusick* 48
Testing Tools under UNIX . . . . *Peter Chubb* 54
An Overview of GCC Porting . . . . *Frank Crawford* . . . 64

From login: - Volume 19, Number 1

Review of BSD/386 72
Computers Could be Like Autos 73
An Update on UNIX-Related Standards Activities 74

From login: - Volume 19, Number 2

An Update on UNIX-Related Standards Activities 81
Unix Tricks & Traps . . . . *Adrian Booth* 91
Summary of Management Committee Minutes - 11th February 1994 93
AUUG Membership Categories 95
AUUG Forms 96

ADVERTISEMENTS

McKusick does Oz T-shirt 17
Advanced User Systems Pty Ltd 27
Prentice Hall Book Order Form 41
WoodsLane Information 42
ZIRCON Systems 43
Softway 47
UniForum NZ'94 53

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.

* UNIX is a registered trademark of X/Open in the United States and other countries.

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
Freephone:	1-800 625 655
Email:	auug@munnari.oz.au

AUUG Business Manager

Catrina Dwyer,
P.O. Box 366,
Kensington, N.S.W. 2033.
AUSTRALIA

Phone:	(02) 959 3656
Fax:	(02) 957 6706
Email:	catrina@sw.oz.au

AUUG Executive

President

Phil McCrea
pmc@atom.ansto.gov.au
ANSAMS
Private Mail Bag 1
Menai NSW 2234

Vice-President

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

Secretary

Peter Wishart
peter.wishart@csis.dit.csiro.au
CSIRO Div. of Information Technology
GPO Box 664
Canberra ACT 2601

Treasurer

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

Committee Members

Greg Birnie
greg@lna.oz.au
Leeds & Northrup Australia P/L
42 McKechnie Dr.
Brisbane Tech. Park
Eight Mile Plans QLD 4113

Stephen Boucher
stephen@mtiame.mtia.oz.au
MTIA
509 St. Kilda Rd.
Melbourne VIC 3004

Chris Maltby
chris@softway.sw.oz.au
Softway Pty. Ltd.
P.O. Box 305
Strawberry Hills NSW 2021

Michael Paddon
mwp@munnari.oz.au
Kodak
173 Elizabeth St
Coburg, Vic 3058

Rick Stevenson
rick@stallion.oz.au
Stallion Technologies Pty. Ltd.
56 Sylvan Rd.
Toowong, QLD 4066

AUUG General Information

Next AUUG Meeting

The AUUG'94 Conference and Exhibition will be held from the 7th to 9th September, 1994, at the World Congress Centre, Melbourne.

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 and AUUG'93 are still available, at $50 for members and $60 for non-members. 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. A copy of FrameMaker for use in the production of the newsletter has been provided by Platform Technologies.

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 15 Number 2. By now the Summer Conferences and Kirk's workshops are over, and they proved to be a huge success. Kirk managed to visit every AUUG chapter in a whirlwind couple of weeks, and to commemorate this, we will be printing a special T-shirt. Along with Kirk's tour, all the chapters have held their local conferences, which again, were very successful.

Looking to the future, we have a number of events coming up. The first is the *early bird registration* for the Winter Conference, which should be included as an onsert to this edition, and secondly, the upcoming Management Committee election, with all positions being contested, including 13 people for the general committee! Keep an eye on your mail for the election material.

Getting back to this edition of AUUGN, we have a number of reports from the chapters on their conferences, followed by papers from the Sydney conference (other conference papers will be published in following issues). We also have a copy of Kirk's slides from his presentation at the conference on new features in 4.4BSD, which have been printed with his permission.

We also have another of Adrian Booth's Electronic Interviews, this time with Piers Lauder. I believe he is planning to present something at the Winter Conference on AUUG's history. To go along with this I've published the list of the original subscribers to AUUGN, printed in Volume 1, Number 1. Are you in this list?

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

The AUUGN book review editor is Frank Crawford. Anyone interested in reviewing books or with book reviews to submit for publishing in AUUGN please contact Frank. His address can be found on page two of this issue. Remember, that any books you review, you keep.

Contributions

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

Volume 15 No 3	Friday 27th May
Volume 15 No 4	Friday 29th July
Volume 15 No 5	Friday 23th September
Volume 15 No 6	Friday 25th 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 President's Page

I'd like to fill you in on two activities that I have been involved with recently.

1. A Milestone in UNIX's history

On March 24 I had the pleasure of launching Berny Goodheart's book, *The Magic Garden explained*, in my capacity as AUUG President. The venue was the Chinese Garden at Darling Harbour in Sydney, which was certainly appropriate, given the title of the book. The actual event was organised and sponsored by Prentice Hall, and they have hopes of having a global best seller on their hands. The title of the book belies its contents, which is concerned with the internals of SVR4!

I was particularly pleased to launch the book, because in a very real sense it has re-instated Australia's position on the world stage as a centre of UNIX expertise. Those of you who are relatively new to UNIX may be interested to know that Australia was a real hot-bed of UNIX activity in the mid to late 1970s. UNIX arrived in the country in 1975, and in the ensuing years, the Universities of NSW, Wollongong, Sydney and Melbourne were at the forefront of UNIX development. We missed a glorious opportunity at that time, because the University of California at Berkeley set itself up as a clearing house for University produced UNIX code, and managed to get Universities round the world to sign their rights away!

But I digress! One of the important documents to come out of the Australian UNIX development efforts in the 70s was John Lions' *Commentary on the UNIX Operating system*. This was the first definitive monograph on UNIX, and is still a revered volume amongst UNIX affictionados the world over.

During the 80s two major monographs on UNIX were produced: *The Design of the UNIX operating system* by Bach in 1986, and *The Design and Implementation of the 4.3BSD Operating System*, by Leffler, McKuisick et al. Both these books are American.

The point with all this historical reflection is that the 4th, and most recent, major UNIX monograph is again Australian – or at least partly Australian, since Berny has agreed to finally become an Australian citizen! Berny is (of course) a member of AUUG, and this book is something we as AUUG members can feel proud of. I commend it to you.

2. Corporate Sponsorships

AUUG has embarked on a campaign of procuring Corporate Sponsorships. The benefits of Corporate Sponsorships are spelt out elsewhere in this journal, and our aim for this coming year is to enlist all the major hardware providers as Sponsors.

Thus far IBM and Sun have become Sponsors, for which we thank them. I envisage that the benefits of Sponsorship will be mutual: AUUG benefits financially, and our Corporate Sponsors benefit through avenues such as access to AUUG members, and the potential endorsement by AUUG of certain technologies and products which further the cause of UNIX: WABI, for instance, is one such area. I have described WABI before as 'liberating technology', as it allows most of the popular Windows packages to run on a host of UNIX platforms.

Phil McCrea

CORPORATE SPONSOR

A New Level of Participation

Since it's inception in 1974, AUUG Inc has been a keystone of the computing industry in Australia offering community services and education to Open Systems users. For the first time, AUUG is offering a new and prestigious level of participation to its long term supporters; the Corporate Sponsorship Scheme.

The scheme is a mutually beneficial one. In return for an annual donation of $5,000 to AUUG's ongoing promotion of Open Systems, Corporate Sponsors will receive:

- complimentary unlimited access to the AUUG mailing list each year which reaches some 1200 of the most influential Open Systems users in Australia
- prominent listing on all AUUG literature
- public acknowledgment by AUUG of the Sponsor
- an attractive wall plaque for prominent display in their office

All of these will continue to be augmented by the benefits currently enjoyed by AUUG Institutional Members:

- two nominated representatives to receive full membership benefits
- a nominated representative to become the AUUG voting member
- access to AARNet at substantially discounted rates

If you would like to know more about the AUUG Corporate Sponsorship Scheme, please contact:

Either:

Philip McCrea
AUUG Inc. President
tel: (02) 717 9401
email: pmc@atom.ansto.gov.au

Catrina Dwyer
AUUG - Business Manager
tel: (02) 959 3656
email: catrina@sw.oz.au

AUUG is pleased to announce the participation of the following companies in the Corporate Sponsorship Scheme.

IBM

Sun

AUUG Institutional Members as at 14/04/1994

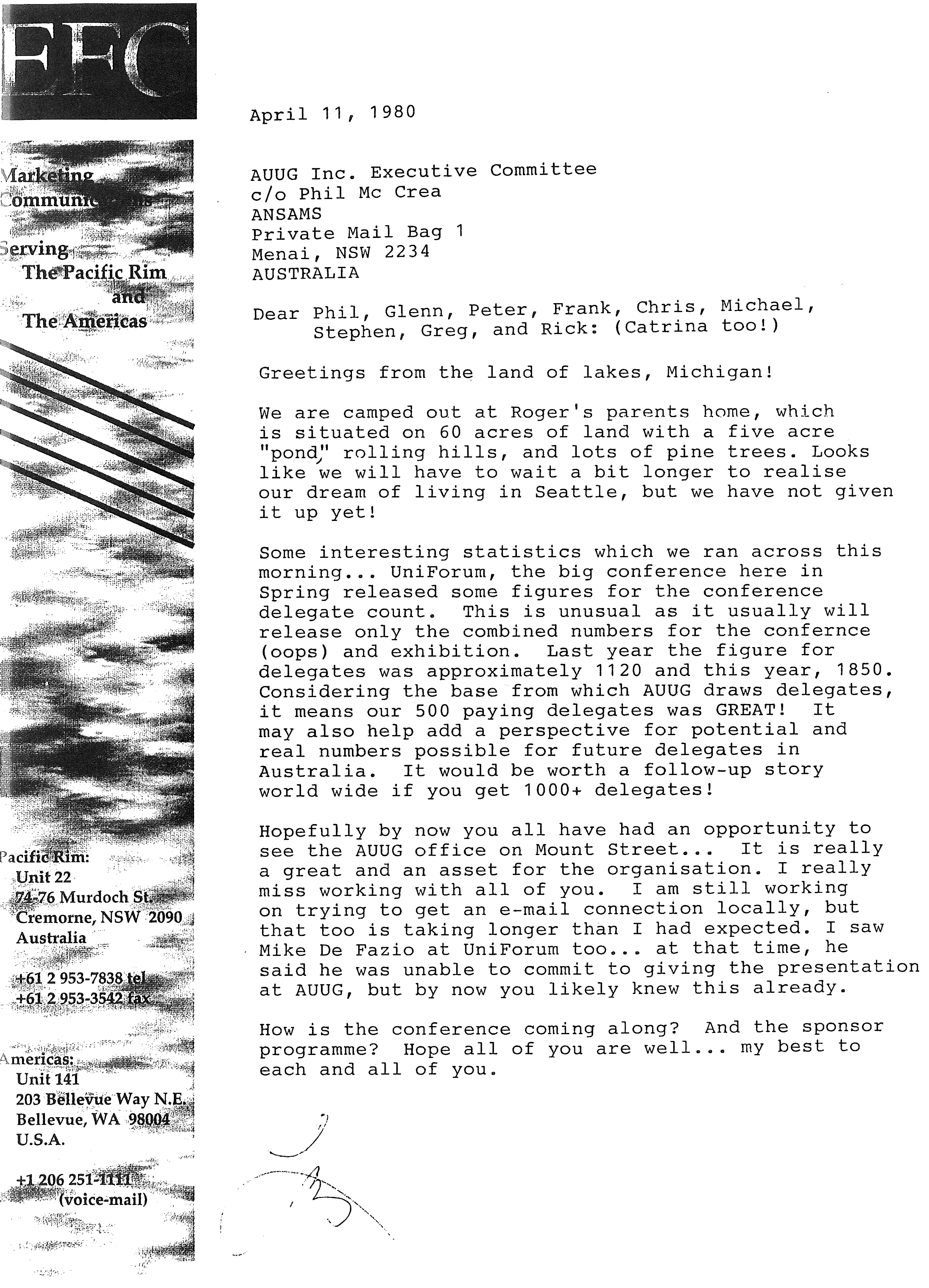
A. Goninan & Co. Limited
A.J. Mills & Sons Pty Ltd
A.N.U.
AAII
Aberfoyle Resource Limited
ACAY Network Computing Pty.Ltd.
ACT Government Computing Service
Actrol Parts
Adept Software
Advanced Software Engineering
Alcatel Australia
Amalgamated Television Services
Amdahl Australia Pty Ltd
Amdahl Pacific Services
Andersen Consulting
ANI Manufacturing Group
Animal Logic Research Pty. Ltd.
ANSTO
Anti-Cancer Council of Victoria
ANZ McCaughan
Atlas Computer Systems
Attorney-General's Dept.
AUSOM Inc.
Australian Airlines Limited
Australian Archives
Australian Bureau of Statistics
Australian Computing & Communications Institute
Australian Defence Industries Ltd.
Australian Electoral Commission
Australian Film Television and Radio School
Australian Information Processing Centre Pty. Ltd.
Australian Museum
Australian National Audit Office
Australian Software Innovations
Australian Submarine Corporation
Australian Taxation Office
Australian Technology Resources (ACT) Pty. Ltd.
Australian Technology Resources (WA) Pty. Ltd.
Australian Tourist Commission
Australian Wool Research & Promotion Organisation
AWA Defence Industries
B & D Australia
Bain & Company
Bain & Company
Bay Technologies Pty Ltd
BHA Computer Pty. Limited
BHP Information Technology
BHP Minerals Exploration Department
BHP Petroleum
BHP Research - Melbourne Laboratories
BHP Research - Newcastle Laboratories
Bond University
Burdett, Buckeridge & Young Ltd."
Bureau of Meteorology
Bytecraft Pty. Ltd.
C.I.S.R.A.
Cadcom Solutions Pty. Ltd.
Cape Grim B.A.P.S
Capricorn Coal Management Pty. Ltd.
CelsiusTech Australia
Chief Secretary's Dept.
CITEC
Classified Computers Pty. Ltd.
Clegg Driscoll Consultants Pty. Ltd.
Co-Cam Computer Group
Coal & Allied Operations
Cognos Pty. Ltd.
Colonial Mutual
Colonial Mutual
Com Net Solutions
Com Tech Communications
Commercial Dynamics
Communica Software Consultants
Composite Buyers Ltd.
Computechnics Pty. Ltd.
Computer De Tokyo Corporation
Computer Law Corporation
Computer Sciences of Australia Pty. Ltd.
Computer Software Packages
Computer Systems (Australia) Pty. Ltd.
Comsys International Pty. Ltd.
Comsys International Pty. Ltd.
Copper Refineries Pty. Ltd.
Corinthian Engineering Pty. Ltd.
Corporate Systems Planning
Corporate Workgroup Resources
CSIRO Division of Information Technology
CSIRO Division of Manufacturing Technology
CSIRO Division of Wool Technology
Curtin University of Technology
Customised Software Solutions Centre
Cyberdyne Systems Corporation Pty. Ltd.
Cyberscience Corporation Pty. Ltd.
Cybersource Pty. Ltd.
Data General Australia
Datacraft Technologies
Dawn Technologies
Deakin University
Defence Housing Authority
Defence Service Homes
Dep. of Environment & Natural Resources
Dept. of Agricultural & Rural Affairs
Dept. of Business & Employment
Dept. of Defence
Dept. of Education, QLD
Dept. of Family Services & Aboriginal & Islander Affairs
Dept. of Industrial Relations, Employment, Training & Further Education
Dept. of Justice
Dept. of Planning & Development
Dept. of State Services
Dept. of the Premier & Cabinet
Dept. of the Treasury
Dept. of Transport
Department of State Services
DEVETIR
Digital Equipment Corp. (Australia) Pty. Ltd.

AUUG Institutional Members as at 14/04/1994

DSTO, Lab 73
EASAMS (Australia) Limited
Edith Cowan University
Electricity Trust of South Australia
Electronic Financial Services Limited
Engineering Computer Services Pty. Ltd.
Equinet Pty. Ltd.
Equity Systems Pty. Limited
Ericsson Australia Pty. Ltd.
ERIN Unit, Australian National Parks & Wildlife Service
ESRI Australia Pty. Ltd.
Executive Computing
FGH Decision Support Systems Pty. Ltd.
Financial Network Services
Fire Fighting Enterprises
First State Computing
Flinders University
Fremantle Port Authority
Fujitsu Australia Ltd.
G.James Australia Pty. Ltd.
GEC Alsthom Australia
GEC Alsthom Information Technology
GEC Marconi Systems Ltd.
Geelong & District Water Board
Genasys II Pty. Ltd.
General Automation Pty. Ltd.
GIO Australia
Golden Casket Office
Golden Circle Australia
Great Barrier Reef Marine Park Authority
Gribbles Pathology
Gunnedah Abattoir
Haltek Pty. Ltd.
Hamersley Iron
Heath Insuramce
Hermes Precisa Australia Pty. Ltd.
Honeywell Ltd.
Hong Kong Jockey Club Systems (Australia) Pty. Ltd.
I.P.S Radio & Space Services
IBA Healthcare Pty. Ltd.
IBM Australia Ltd.
Iconix Pty. Ltd.
Ideas International Pty. Ltd.
Independent Systems Integrators
Information Technology Consultants
Informed Technology
Insession Pty. Ltd.
Insurance & Superannuation Commission
Integration Design Pty. Ltd.
International Imaging Systems
Internode Systems Pty. Ltd.
ISR Group Ltd.
James Cook University of North Queensland
JTEC Pty. Ltd.
Knowledge Engineering Pty. Ltd.
Laboratory Systems Pty. Ltd.
Labtam Australia Pty. Ltd.
Land Information Centre
Land Titles Office
Leeds & Northrup Australia Pty. Limited
Legent Australia Pty. Ltd.
Logica Pty. Ltd.
Lotus Development
Lyons Computer Pty. Ltd.
Macquarie University
Main Roads Western Australia
Matcom Technologies
Mayne Nickless Courier Systems
Mayne Nickless Information Technology Services
Medical Benefits Funds of Australia Ltd.
Memtec Limited
Mentor Technologies Pty. Ltd.
Mercedes-Benz (Australia)
Metal Trades Industry Association
Mincom Pty. Ltd.
Minenco Pty. Ltd.
Mitsubishi Motors Australia Ltd.
Mitsui Computer Limited
Moldflow Pty. Ltd.
Motorola Computer Systems
Motorolla Communications Australia
MPA International Pty. Ltd.
Multibase Pty. Ltd.
Multiline BBS
Multiuser Solutions Pty. Ltd.
National Library of Australia
National Resource Information Centre
NCR Australia
NEC Australia Pty. Ltd.
Northern Territory University
Novell Pty. Ltd.
NSW Agriculture
NSW Teachers Federation Health Society
Object Design Pty. Ltd.
Object Oriented Pty. Ltd.
Object Technology International Pty. Ltd.
Ochre Development
Office of Fair Trading
Office of National Assessments
Office of State Revenue
Office of the Director of Public Prosecutions
Olivetti Australia Pty. Ltd.
Open Software Associates Ltd.
Open Technology Pty Ltd
Opentec Pty Ltd
OPSM
OSIX Pty. Ltd.
OzWare Developments Pty. Ltd.
Pacific Semiconductor Pty. Ltd.
Pacific Star Communications
Paxus
Petrosys Pty. Ltd.
Philips PTS
Platform Technologies Pty. Ltd.
Port of Melbourne Authority
Powerhouse Museum
PPIT Pty. Ltd.
Process Software Solutions Pty. Ltd.
Prospect Electricity
pTizan Computer Services Pty. Ltd.
Public Works Department, Information Services
Pulse Club Computers Pty. Ltd.

AUUG Institutional Members as at 14/04/1994

Pyramid Technology Corporation Pty. Ltd.
Qantek
QLD Electricity Commission
Quality Bakers Pty. Ltd.
Quality By Design Pty. Ltd.
Redland Shire Council
Rehabilitation Tasmania
Renison Golfields Consolidated Ltd.
Repatriation General Hospital, Hollywood
RGC Minerals Sands, Divisional Office
Rinbina Pty. Ltd.
Royal Melbourne Institute of Technology
SCEGGS Redlands Ltd
Scitec Communication Systems
Sculptor 4GL+SQL
SEQEB Business Systems
SEQEB Control Centre
Shire of Eltham
Siemens Nixdorf Information Systems Pty. Ltd.
Smorgon ARC
Snowy Mountains Authority
SoftGen Pacific Pty. Ltd.
Software Development International Pty. Ltd.
Softway Pty. Ltd.
Sony Technology Centre of Australia
South Australian Co-operative Bulk Handling
St. Catherine's School
St. Gregory's Armenian School
St. John God Hospital
St. Vincent's Private Hospital
Stallion Technologies Pty. Ltd.
Standards Australia
State Bank of NSW
State Revenue Office
State Super (SSIMC)
Steelmark Eagle & Globe
Sterling Software
Storage Technology of Australia
Strategic Information Technologies Pty. Ltd.
Sunburst Regency Foods
Sydney Electricity
Sydney Ports Authority
System Builder Development Pty. Ltd.
Systems and Management Pty Ltd
Systems Development Telecom Australia
TAB of Queensland
TAFE NSW, Information Systems Division
Tandem Computers
Tattersall Sweep Consultation
Tattersall Sweep Consultation
Technical Software Services
TechNIX Consulting Group International
Telecom Australia
Telecom Australia Corporate Customer
Telecom Network Engineering Computer
Support Services
Telecom Payphone Services
The Far North QLD Electricity Board
The Fulcrum Consulting Group
The Preston Group
The Roads & Traffic Authority
The Southport School
The University of Western Australia
Thomas Cook Ltd.
Thomas Cook Ltd.
TNT Australia Information Technology
Toshiba International Corporation Pty. Ltd.
Tower Software Engineering Pty. Ltd.
Tower Technology Pty. Ltd.
Tradelink Plumbing Supplies Centres
Turbosoft Pty. Ltd.
TUSC Computer Systems Pty. Ltd.
UCCQ
UCR
Unidata Australia
Uninet Consulting Pty. Ltd.
Unisys Australia Limited
University of Adelaide
University of Melbourne
University of New England
University of New South Wales
University of Queensland
University of South Australia
University of Sydney
University of Tasmania
University of Technology, Sydney
Unixpac Pty. Ltd.
Vanoco Pty. Ltd.
Vicomp
Victoria University of Technology
VME Systems Pty. Ltd.
Walter & Eliza Hall Institute
Wang Australia Pty. Ltd.
Water Board
Western Mining Corporation
Woodside Offshore Petroleum
Work Health Authority
Workstations Plus
XEDOC Software Development Pty. Ltd.
Zircon Systems Pty. Ltd.
Zurich Australian Insurance



EFC

Marketing Communications

Serving The Pacific Rim and The Americas

Pacific Rim:
Unit 22
74-76 Murdoch St.
Cremorne, NSW 2090
Australia

+61 2 953-7838 tel
+61 2 953-3542 fax

Americas:
Unit 141
203 Bellevue Way N.E.
Bellevue, WA 98004
U.S.A.

+1 206 251-1111
(voice-mail)

April 11, 1980

AUUG Inc. Executive Committee
c/o Phil Mc Crea
ANSAMS
Private Mail Bag 1
Menai, NSW 2234
AUSTRALIA

Dear Phil, Glenn, Peter, Frank, Chris, Michael, Stephen, Greg, and Rick: (Catrina too!)

Greetings from the land of lakes, Michigan!

We are camped out at Roger's parents home, which is situated on 60 acres of land with a five acre "pond," rolling hills, and lots of pine trees. Looks like we will have to wait a bit longer to realise our dream of living in Seattle, but we have not given it up yet!

Some interesting statistics which we ran across this morning... UniForum, the big conference here in Spring released some figures for the conference delegate count. This is unusual as it usually will release only the combined numbers for the confernce (oops) and exhibition. Last year the figure for delegates was approximately 1120 and this year, 1850. Considering the base from which AUUG draws delegates, it means our 500 paying delegates was GREAT! It may also help add a perspective for potential and real numbers possible for future delegates in Australia. It would be worth a follow-up story world wide if you get 1000+ delegates!

Hopefully by now you all have had an opportunity to see the AUUG office on Mount Street... It is really a great and an asset for the organisation. I really miss working with all of you. I am still working on trying to get an e-mail connection locally, but that too is taking longer than I had expected. I saw Mike De Fazio at UniForum too... at that time, he said he was unable to commit to giving the presentation at AUUG, but by now you likely knew this already.

How is the conference coming along? And the sponsor programme? Hope all of you are well... my best to each and all of you.

James Sainsbury
Member Nº: M00319
3/42 Ryan St
Hill End Q4101
(07) 844 0285
3 March 1994

The AUUGN Editor
PO Box 366
Kensington NSW 2033

Dear editor,

I am writing to query whether the *AUUG* article which appeared in the Tueday *AUSTRALIAN* (page 37, March 1) was a tongue-in-cheek afair intended for April 1 but with editorial oversight has prematurely misfired by a month.

If this is not the case then I feel that the many inaccurate and unfounded assertions made in this article should not go unchallenged if only to correct historical inaccuracies. For example, if one were to credit Ken Thompson and Dennis Ritchie with having *stole*[n] *all the best features of DOS...* one would also be obliged to credit these two either with the invention of a time machine or with extra-ordinary facility with the crystal ball etc. The UNIX time-sharing system was first described in 1974[1] while MS/PCDOS appeared in August 1981 (version 1.0) and in March 1983 (version 2.0)[2] — version 2.0 being the first with hierarchical directories and file handle based IO (cf CP/M style FCB based IO of version 1.x).

[1] *The UNIX Time-Sharing System*
K Thompson and D M Ritchie
Communications of the ACM. 17(7):365–375 (July 1974)

[2] *The DOS Programmer's Reference 2/e*
T Dettmann and J Kyle
Que Programming Series 1989
ISBN 0 88022 458 4
See Table 1.1 page 12

Yours sincerely

J. Sainsbury

J Sainsbury.

Response from Article Coordinator for the Australian

If Stuart McCormack, author of "Orange DOS Beats Durian Unix", is guilty of anything, it is the crime of misdirected humour. A revisiting of the article in question reveals that Mr McCormack was using the time honoured humourist's style of beginning in a seemingly straightlaced vain, slowly piling ridiculous assertion upon evermore ridiculous assertion. The plan, of course, is to stretch the reader's suspension of disbelief to the point where it collapses, hopefully generating a smile or two on the way.

In practice, this article met with mixed reactions. Some found it funny, others thought that it should never have printed. What happened? The article wasn't tagged as humour; combine this with the fact that many people skim newspaper columns rather than linearly read them and you have the recipe for misunderstanding.

Try reading isolated bits of a Dave Barry column and you'll see what I mean.

In any case, you'll find a copy of the offending article below. Read it again. Think about each assertion. I think you'll find that the whole effect is a strongly pro-Unix, anti-DOS argument.

This is the final reason why, acting as AUUG's editor, I passed this article on the the Australian. It states our case, albeit in an idiosyncratic fashion. My editorial policy is to act as an advisor and a final filter; I certainly do not intend to instruct our authors in what to think or how to express themselves.

Given some of the adverse reactions it could be argued that, as final filter, I erred in accepting this article. After much reflection, I must disagree with such an analysis. Even completely misunderstood, the article served to promote debate, and I have no doubt that Unix will win any such fight on clear grounds of merit.

I'd also like to take this chance to encourage our members to disagree with what's printed. I want to hear your views, and I think that we've got a lot to tell the world about open systems. I eagerly await your articles.

Michael Paddon

CATCHLINE: Orange DOS Beats Durian Unix / Stuart McCormack

Arguments comparing operating systems are potentially endless; it's like comparing two varieties of fruit. You can argue about the relative merits until you're big blue in the face.

Still, maybe it's worthwhile to continue the analogy.

If DOS were a fruit it'd be something like an apple -- whoops, better scratch that -- an orange; pretty common and everybody knows what they're like.

Unix, on the other hand, would be more like a durian: strange, exotic funny smelling and loved dearly only by those few weirdos that like them at all.

It's easy to compare two fruits just on the basis of their more important and inarguable features, like price, availability, and vitamin content. The same can be done with operating systems, but this process is unfair; subjective issues such as ease-of-use and public perceptions are important.

I've heard Unix described both as an operating system on drugs and as the COBOL of the 90's. Everybody's heard of Unix, but anybody with half a brain knows that only academics and the nuttier techos use Unix. How many real businesses run their operations on Unix platforms just because they're cheap and powerful?

Now DOS, on the other hand, is omnipresent, and a couple of hundred million users can't be wrong. I mean, until you can run Wolfenstein on a Unix box, what use is it?

Unix has sold a few tens, maybe hundreds of thousands of copies. How many copies of DOS have been sold? If you divide Bill Gate's net worth by a factor of about six then you'll start to get the idea.

Probably the biggest advantage of DOS is the availability of add-on features to make DOS based machines (ie. PC's) even more worthwhile. You can buy, by the dozen, memory managers, multi-tasking environments, screen windowing systems, compilers, security features, high resolution displays, network cards, networking software, mice (mouses?), big disks and tons of other good stuff.

Try to add features like those to a Unix box and you're in for a real shock.

On the plus side, Unix does provide, as standard, a multi-user, multi-tasking environment but, really, who needs it? If you absolutely have to have facilities like that you can always wait for the next version of Windows/NT.

Unix also allows you to write complicated scripts (like *.BAT files) that permit decision branching and looping. It's all a bit of a wank though; if you really need facilities like that under DOS, you can almost always buy a package that does what you want.

It seems to me that the Unix guys stole all the best features of DOS and somehow managed to get it wrong anyway. Here's some examples:

* The "cd" (change directory) command is almost right but they got the backslash backwards.

* Unix uses "ls" when they mean "dir". You can't just look in a Unix directory and see what's executable because they don't have to (and don't bother to) use file name suffixes like .EXE, .COM and .BAT.

* Data piping under Unix works funny. For example, under DOS you can type "type filename | more" whereas under Unix you get the choice of "cat filename | more" or even just "more filename". It's just too confusing and, anyway, what sort of weird command name is "cat"?

* Norton's utilities aren't available for Unix. What good is an operating system where you can't Quick Undelete and file? Unix forces users to concentrate on time wasting activities like backups.

* For a long time Unix had the drop on DOS when comparing it's built-in editor; any editor had to be better than EDLIN. Now DOS comes complete with MS EDIT while the Unix bunnies are still fooling around with vi.

Now, without doubt, vi is an editor designed for loonies. You don't have to move your right hand over to the side of the keyboard to use an arrow key, because vi doesn't really understand the concept of arrow keys. Plus, there is always at least three ways to do anything you want to do; now that's confusing.

If you've never used vi and you want to discover just how weird it is, crank it up and then try to get out of it. Good luck.

So why is DOS better than Unix? Because it is.

For immediate release
Issued: March 23, 1994

AUUG INC
UNIX® AND OPEN SYSTEMS USERS

AUUG appoints new business manager

AUUG has appointed Ms Catrina Dwyer to the position of business manager, replacing Ms Liz Fraumann who has returned to the United States.

Prior to joining AUUG, Ms Dwyer spent five years with UNIX System Laboratories in London where she held a number of positions in the area of licensing—the last as an account manager with responsibility for the sales and marketing of USL licenses and products in France.

Ms Dwyer is a graduate of the University of Southampton and has a BA Hons in Modern Languages and European Studies.

ends

AUUG Inc., the Australian UNIX and Open Systems User Group, exists to provide UNIX and open systems users throughout Australia with relevant and practical information, services and education through co-operation among users.

Distributed on behalf of AUUG by Lachie Hill Consulting Pty Limited (A.C.N. 056 534 117).

Contacts:

Catrina Dwyer	(AUUG)	(02) 959 3656
Lachie Hill	(Lachie Hill Consulting)	(02) 953 5629

NEWS...NEWS....

"THIS IS AUSTRALIA CALLING"

1

8
0
0

6
2
5

6
5
5

To ensure that AUUG members Australia wide benefit from belonging to the organisation, AUUG is changing it's telephone number to a new Freephone number. That's right, no matter which state or territory you're calling from, AUUG members can call with questions, suggestions and comments **FREE OF CHARGE.** Simply dial the following number and you'll be connected to the AUUG Secretariat in Sydney.

1-800 625 655

What if the Secretariat cannot answer your query immediately? No problem, we'll call you back with the answer.

For further information about this telephone number change, please contact:

Either: AUUG Secretariat
whfoda@acms.auug.oz.au

or: AUUG Business Manager
catrina@sw.oz.au

on the above Freephone number.

McKusick does Oz T-shirt

Announcing the limited release of a souvenir "McKusick does Oz" T-shirt

To mark the successful conclusion of Kirk McKusick's whirlwind tour of Oz for the 1994 AUUG summer conferences, AUUG is producing strictly limited quantities of a "McKusick does Oz" T-shirt.

The Ash Gray T-shirt features (in black, blue and red) the Berkeley daemon as it appears on the 4.4BSD T-shirt, skewering Australia (including Tasmania!) on its pitchfork (reproduced below). The back of the T-shirt lists the "tour dates" of Kirk's trip, under an AUUG Summer Conference logo.

Volunteers are being organised from each chapter to collect local orders and payment, which will then be forwarded to Ian Crakanthorp (ian@atom.ansto.gov.au), who is doing the central coordination. The bulk order from each chapter will be sent altogether for the chapter to distribute.

The T-shirts cost $15 each, which includes shipping to each chapter but not shipping from the chapter to orderers. Chapters will be making their own arrangements for local delivery.

Orders must be to Ian by May 31, so local chapters will have cutoff dates sometime before that date.

For information on who your local contact is, contact your local chapter, or email Adrian Booth (abcc@dialix.oz.au).

McKusick
does Oz

Call for Articles for the Australian

The Australian newspaper runs an AUUG column every Tuesday, in its computer section. The aim of these articles is to inform the public and raise the profile of open systems within this country. Having one's views published in a respected national paper also carries kudos and recognition for authors.

AUUG would like to ensure that all members of the open system community have access to this voice, and we are actively seeking a diverse spectrum of people and opinions.

If you are interested in being part of this process, please provide me with the following information:

* your name
* contact details
* a copy of your article

Your article should be between 600 and 800 words in length, and can address any issue that may be of interest within the open systems community. If you can't decide on an appropriate topic, please provide me with some professional details and I'll try and give you some ideas tailored to your expertise. Some typical subjects are listed at the end of this article.

If you have access to email, this is the preferred form of submission. Please format your article as a plain text file, with lines no longer than 79 characters, and with a blank line separating paragraphs. If you don't have email, please provide a hardcopy in a similar format (there's not much point doing any fancy typesetting).

All submissions are accepted on the understanding that they may or may not be used and that the material may be edited. AUUG will only submit your work to the Australian newspaper, although unless you advise us otherwise we will reserve the right to add your articles to a public FTP archive at some time in the future. The copyright on the material remains yours, your act of submitting material only gives us licence for the abovementioned purposes.

In practice, I submit your work to the Australian unedited and leave the decision of what to print up to them (I'm not in the business of being a thought police!). Usually a period of 2 to 4 weeks will then pass before you'll see your article in print; I maintain a pipeline of material to buffer me against the inevitable fluctuations in supply.

Please email or phone me if you have any further queries.

Michael Paddon
mwp@mtiame.mtia.oz.au
(03) 353 2382

Some topical areas to get you started :-

Standards: POSIX, X/Open, System V.
The sudden demise of COSE;
just another consortium?
The history of Unix.
The future of Unix.
If NT is so popular,
how come no one is using it?
Competition for the desktop:
Unix, Windows/NT and OS/2.
Managing security.
Computer viruses.
System administration.
Network administration.
Networking technologies.
Distributed computing.
What happened to OSI?

Managing multiple network protocols.
Living with the Internet.
Unix on PC's.
Linux - the people's Unix?
Would you run Unix at home?
The graphics revolution.
Virtual reality.
CASE tools.
Is Unix really that hard to use?
Now that Unix has grown up,
where have the hackers gone?
The costs of open systems.
Analyse a market trend.
Run a straw poll on a topical subject.
New technologies.

Announcement

AUUG & ZIRCON SYSTEMS REACH AGREEMENT

Sydney, NSW -- 23 February 1994

In the continually expanding benefits offered to AUUG members, Zircon Systems, distributors of the Air Series™ 2.0, and AUUG Inc. are pleased to announce agreement was reached with access to their full range of Internetworking applications for Windows at a special introductory price of $99 (RRP $499).

A wholly owned Australian company, established in 1986, Zircon Systems specialise in computer and communications consulting. The organisation is the leading distributor of this new technology from SPRY™ out of Seattle, Washington in the United States. The Air Series is designed to ease the integration of the diverse computing environments within organisations. Its easy-to-use interface, coupled with a uniquely flexible architecture, shields users from the underlying complexity of the network. The interoperable architecture, breadth of the applications, and high performance make the Air Series a sensible choice to internetwork Windows desktops.

Gerardo D'Angelo, marketing director, Zircon Systems said, " We are pleased to bring these products to Australia. Rarely in the past has a series of products had such flexibility to run on such a vast number of host machines. The Air Series will work on: DEC: VMS and Ultrix; HP UX, LM/X; IBM: AIX, VM, MVS; Unisys; Sun Solaris; BSD and System V Release 4."

AUUG President, Phil McCrea said, "We are pleased to be working with Zircon to offer our members access to products like the Air Series. The network support encompasses the majority of the major manufacturers from Novell NetWare™ to Banyan Vines and Microsoft products. This is exciting for all AUUG members."

Full details and specifications may be acquired directly from Zircon Systems. AUUG members should remember to state their affiliation and member identification number on all correspondence.

For further information:

Catrina Dwyer - AUUG
(02) 361-5994 tel
(02) 332-4066 fax
email: catrina@sw.oz.au

Gerardo D'Angelo - Zircon Systems
(02) 317-4055 tel
(02) 669-3241 fax
email: gerardod@zircon.oz.au

Ann~~ouncement~~

AUUG & THE EXPRESS BOOK STORE REACH AGREEMENT

Sydney, NSW -- 23 March 1994

In the continually expanding benefits offered to AUUG members, The Express Book Store and AUUG Inc. are pleased to announce an agreement offering discounts on popular computer titles.

Established in 1991, The Express Book Store was the first direct mail order company to specialise in computer titles. Covering all areas from business applications to advanced programming, an up to date listing of titles ensures that AUUG members will be offered the best selection available.

Representing the major publishers of computer books and most of the specialist presses, The Express Book Store is pleased to offer AUUG members a discount of 10% on all titles. Orders totaling $100 or more will attract a 15% discount.

Geoff Wayling, Managing Director of The Express Book Store said, " I am happy to be able to offer these discounts to AUUG members. Our competitive pricing and free freight service means saving time and money".

AUUG Business Manager, Catrina Dwyer said, "One of AUUG Inc.'s objectives is to provide practical and relevant services to its members. In securing these discounts with The Express Book Store, AUUG is providing yet another practical benefit to its members".

Full details and specifications may be acquired from the enclosed brochure or directly from The Express Book Store. AUUG members should remember to state their affiliation and member identification number on all correspondence.

For further information:

Catrina Dwyer - AUUG
(02) 361-5994 tel
(02) 332-4066 fax
email: catrina@sw.oz.au

Geoff Wayling - The Express Book Store
(02) 918-0108 tel
(02) 973-2349 fax

Updated AUUG Regional Contacts
1994 -1995

Location	Contact	Tel/Fax
Adelaide	Michael Wagner mhw@syserv.com.au	tel: (08) 212-2800 fax: (08) 231-0321
Brisbane	Greg Birnie greg@lna.oz.au	tel: (07) 340-2111 fax: (07) 340-2100
Canberra	John Barlow john.barlow@anu.edu.au	tel: (06) 249-2930 fax: (06) 249-0747
Darwin	Phil Maker pjm@cs.ntu.edu.au	tel: (089) 466-666 fax: (089) 270-612
Hobart	Steven Bittinger steven.bittinger@its.utas.edu.au	tel: (002) 207-406 fax: (002) 207-488
Melbourne	David Taylor davet@vaxc.cc.monash.edu.au	tel: (03) 857 5660
Perth	Glenn Huxtable glenn@cs.uwa.edu.au	tel: (09) 380-2878 fax: (09) 380-1089
Sydney	Julian Dryden julian@dwt.csiro.au	tel: (02) 809-9345 fax: (02) 809-9476

AUUG Inc. - Victorian Chapter

(formally SESSPOOLE)

AUUG-Vic is the official Victorian chapter of AUUG Inc. It was the first Chapter of the AUUG to be formed, then known as SESSPOOLE, and its members have been involved in the staging of the Victorian AUUG Summer technical meetings every year since 1990. AUUG-Vic 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 programme 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, 14 April '94	Social
Tue, 24 May '94	Technical
Wed, 6 July '94	Social
Thu, 18 August '94	Technical

Hope we'll see you there!

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

AUUG-Vic Committee *<auugvic-exec@clcs.com.au>*			
President:	Enno Davies TechNIX Consulting Phone: (03) 819 3339 Email: enno@technix.oz.au	**Secretary:**	David Taylor Monash University Phone: (03) 857 5660 Email: davet@vaxc.cc.monash.edu.au
Treasurer:	Neil Murray Webster Computer Corporation Phone: (03) 560 1100 Email: neil@wcc.oz.au	**Programme Chair:**	Michael Paddon Iconix Phone: (03) 571 4244 Email: mwp@munnari.oz.au
Committee:	Arnold Pears La Trobe University Phone: (03) 479 1144 Email: pears@latcs1.lat.oz.au	**Committee:**	Peter Lazarus Legent Australia Phone: (03) 286 5200 Email: plazarus@auspacific.legent.com

Update on AUUG Inc. - Victorian Chapter Activities

by Stephen Prince
Ex-President, AUUG-Vic.
<sp@clcs.com.au>

Yes, the title is correct. At the AGM on March 16, I stepped down as president and handed over the position to *Enno Davids*. For a complete list of the new committee line-up, see the AUUG-Vic announcement elsewhere in this issue. It has been fun, but I feel I have to move on to make room for "fresh blood". So this could well be my last column. For those concerned about continuity, I will be taking a passive role on the new committee.

Regular Meetings

For those left woundering, the technical meeting on March 2 didn't eventuate; it got lost in all the summer conference organisation. However, the good news is, *Michael Paddon* is continuing on with his role as programme chair, so we can expect another year of great technical meetings.

Kirk McKusick Workshop

Before anyone else asks: I did have that quiet drink. Kirk and myself descended on some small pub just around the corner from Robert Elz's place.

As for the event, we had 44 registration. This was comprised of 86% of people who have been using UNIX† for more than three years, 75% had been administering it for greater than three years and 33% had been hacking systems for more than three years. Interesting though, 20% had never delved into the world of system hacking. Well, they are all ‘‘mentally contaminated’’ now. As for the presentation, the majority (about 78%) felt the speed of which was about right and it wasn't too cursory or too detailed. There was almost an even split between those who felt it should have been three or four days. All in all, 86% felt it was useful and very valuable and nobody thought it was a waste of time. :-)

It appears that sections which delegates found of most benefit: anecdotes and relationship to POSIX, security, networking, the additional papers at the end of the course notes, kernel organisation, the algorithms and design strategies or assumptions, 4.4BSD additions, profiling and system tuning, file systems, virtual memory and paging.

The overall impressions of all delegates can best be summed up by quoting one of the delegates, *Warren Toomey*: ‘‘Excellent. I await the 4.4 book, and the movie :-)’’.

Summer94 (Vic) Conference

A bit of sad news to report here; our ambition of running a separate tutorial day did not eventuate. The reason being was a lack of interest by potential presenters. As for the conference, it did go ahead and was deemed to be informative, thought provoking and interesting by the delegates. Some said it was the best summer AUUG so far, thanks to *Arnold Pears*. As I promised at the conference, a copy of the slides from Kirk McKusick's talk have been reproduced in this issue of AUUGN.

Chapter Rules

I'm happy to report that the latest version of the AUUG-Vic Rules and Policy document, has been approved by the National AUUG committee. This document provides a framework for describing and formalising the activities of the Victorian chapter. A PostScript copy of this document is available for anon ftp on *yarrina.connect.com.au* in the usual area.

Internet Access

One issue that was raised and hotly debated at the AGM, was that of Internet access being provided to AUUG members. Whilst AUUG-Vic would like to provide a benefit in this area, they need the answer to some questions, that is, what exactly is it that users want and how many of them want it. Three thoughts seemed to emerge from the debate:

(i) AUUG-Vic should provide a full service, dial-up accounts to an internet machine.

(ii) AUUG-Vic should provide a machine for members to connect their machines to.

(iii) A mail forwarding setup, in which members have a permanent address (on say *vic.auug.org.au*) that can be aliases if desired.

Members felt that option (i) if cheap enough, would be beneficial when they are between jobs or in a job with out any form of internet access. Option (iii) is also just an extension of (i). The committee has been talking to a commercial provider about running this service for our members. Before proceeding with any of these options, the committee would really like to know the exact break down of how the members fit into the above categories.

If you have any thoughts, ideas, comments on any matters, please feel free to contact the committee, preferably via email: *auugvic-exec@clcs.com.au.*

† UNIX is a registered trademark of X/Open in the United States and other countries.

From the Western Front

We've had an interesting couple of months here in Perth. The Summer Conference was a success again. Although attendance was a bit disappointing, and the majority of the talks were given by the usual suspects rounded up by Adrian, the content and presentation were particularly excellent this year.

It was great to have an international guest speaker, Kirk McKusick. There is now talk of having Gene Spafford tour the country in conjunction with the winter conference, which sounds like being an even bigger logistical challenge. Member benefits like that would have been almost impossible for WAUG alone without the organisational and financial backing of the national body.

WAUG's February meeting was held as part of the conference, with Kirk speaking on free software -- and software freedom. What really stuck in my mind was his description of some of the ridiculous software patents that have been taken out in the USA — would you believe someone has patented the linked list? That such a thing could even be considered, let alone approved, defies common sense.

Adrian, elsewhere in this issue, has saved our March talk from the panning it would have received if I had been left to review it. I will admit that fast, good-looking machines with colourful names (such as "Power Challenge"!) can inspire in me a certain amount of techno-lust (and budget-envy), but marketing graphs with logarithmic Y-axes don't impress me at all. Neither do speakers who think that a sales presentation can be made to appeal to a more technical audience by throwing in words like "cache" now and again. But the final straw for me was when the speaker managed to work in the dreaded phrase... *information superhighway*. Aagh!

However, Silicon Graphics have supported us with sponsorship of the last two Summer Conference proceedings, so maybe this talk was their reward. I believe they also provided some sponsorship for the March meeting. There was far too much food, but this was probably because the Curtin Uni vultures weren't around.

Speaking of SGI, Dennis O'Shea is now flogging their systems in Perth, after many years selling Suns. The interesting thing about Dennis is that he is one of the few computer salespeople I actually like. Over the years he has managed to infiltrate Suns into a large number of the idiosyncratic nooks and crannies of my labyrinthine employer. Maybe he will be able to do the same for SGI, even though some of us *have* seen *Jurassic Park*.

Janet Jackson (WA Chapter Sub-editor) <jackson@cwr.uwa.edu.au>, (09) 380 2408

From WAUG, the WA Chapter of AUUG

WAUG Meeting Review

March
Silicon Graphics: From Desktops to Supercomputers

Phil Edmiston, Silicon Graphics

The first half of Phil's talk described Silicon Graphics' (SGI) "success story" and gave a brief company and product overview. I found it somewhat interesting but just a little too marketing oriented, which I guess is to be expected when it's the vendor giving the talk :-)

Phil moved on to SGI's forte — 3D visualisation (have *you* seen *Jurassic Park*?). SGI's attitude is that the human brain is very 3D-oriented, so using computers to present information in 3D has several benefits: it integrates large amounts of information, allows us to quickly grasp and understand complex relationships, to synthesise diverse information, and to interact with the model, leading to rapid project completion and more confident decisions.

For example, every one of Ford's design engineers throughout the world has a SGI Onyx graphics supercomputer on his or her desk. This allows collaborative design between engineers around the world. SGI supply as standard the software that allows multimedia communication between these engineers.

The main required features of a 3D visualisation system are 24-bit colour, Z buffering, fast data paths, a 3D graphics library, hardware accelerators, and an appropriate system architecture.

Phil briefly discussed SGI's moves into the "Information Superhighway" — SGI Power Challenge servers are being used to deliver interactive video into the home in a pilot project in Florida. The home terminals also incorporate SGI graphics technology. (The founder of Silicon Graphics left recently to build a new software company dealing with these sorts of issues).

Phil also described how SGI is moving into the supercomputer market — delivering systems with a substantial fraction of the power of a Cray Y-MP at a much lower price. SGI expect the performance of their multiprocessor systems to rival the leading supercomputers from traditional supercomputer vendors within the next two years.

Finally, SGI's new "Indy" workstation was described. I was surprised at the yawns from the audience as Phil described this (64-bit architecture) workstation. A useable configuration — including a colour monitor, a digital camera, 16Mb RAM and 350Mb or so disk — is available now for around $14,000 list. Perhaps the audience was demonstrating its reluctance to listen to anything that sounded even vaguely marketing, or perhaps they haven't seen just how fast they are :-)

Phil came in for some tough questioning on the architecture of the multiprocessor servers, but referred the questioners to the white paper being handed out at the meeting.

Phil closed with a discussion of the penetration of SGI and MIPS (wholly owned by SGI) into a broad range of markets, from embedded controllers to graphics supercomputers.

I personally found the description of the architecture of the supercomputers the most interesting part, but several people I spoke to afterwards firmly disagreed :-) I also thought that the talk gave a reasonable flavour of how we can expect to see workstation and supercomputing technologies blurring in the marketplace over the next few years.

Adrian Booth, Adrian Booth Computing Consultants <abcc@dialix.oz.au>, (09) 354 4936

From WAUG, the WA Chapter of AUUG

1994 Perth AUUG Summer Conference Overview

The 1994 Perth summer conference — what a tremendous success! The programme chair was inundated with far too many high-quality technical papers to possibly fit into one day. The tutorial chair had to dramatically whittle down submissions from people eager to give in-depth tutorials in areas like UNIX Security, UNIX performance tuning, `perl` and `Tcl/Tk`. And the number of registrations was so overwhelming, we had to change the conference venue (twice!).

Poof! — I wake up, head on the keyboard, and count again how many papers have been received in response to the call for papers. Zero. That can't be right — count again. Zero. Ask my wife to check my addition. Still zero.

Hmmmm. Take the direct approach. Stand up at a WAUG meeting and say, "If you haven't spoken before at a summer conference, I expect to hear you give a talk unless you can provide in writing three good reasons why you can't!". Everyone leaves by a rear exit while my back is turned.

OK, write out a "hit list". First target people who I know could give a good talk on a particular topic, but haven't spoken before. Send out something that hopefully doesn't look too much like a form letter. Get 10% reply rate, of which 100% say "No".

Time to dust off the veterans. Send out same form letter^H^H^H^H^H^H^H^H^H^H^H individual plea to each of them. Eventually get four local speakers! Success!

Hang on — four local speakers aren't enough (unless we have a two-hour lunch, and I know we wouldn't see anything of the guys from Curtin University all afternoon if we did that). Maybe I could give a talk to fill in a slot? But I'm the programme chair! Would I go blind if I approved my own talk? By this stage, I didn't care. A 10-word abstract was duly submitted and approved in record time.

Meanwhile, organise Kirk's tutorial. Whaddaya mean, a one-day tutorial? I'd rather have two or three days. Discuss with committee. Major makes strong argument for two-day version. Major agrees with me; he must be right. Arrange with Kirk to have a two-day tutorial.

Finally, a draft programme is ready. Write out an overview, a programme, a registration form. Snail mail to about 300 companies. Lick 300 envelopes. Try to figure out why laser printer won't feed envelopes any more. Post letters, with less than four weeks to go until the conference. Spend next week contemplating empty letterbox.

Time passes...

Time passes...

Time passes...

Wow, we have 40 registrations, which is even less than last year! Still, Kirk's tutorial is filling up well, and with mainly quite technical people. Should be good.

Friday: Kirk arrives on 4pm flight. He is early; I'm late. Miss turnoff from airport, take Kirk the scenic way home. Exhaustively pack car with essentials for a weekend in the South West — a bottle of wine and a bottle opener. Three-hour drive. Open bottle of wine upon arrival. Feel more relaxed.

Try to compress a week-long holiday into 48 hours. Forget it, try to compress a week's winetasting into 48 hours. Noble effort. Drink and eat too much. Drive back Sunday evening.

Monday: day 1 of tutorial. Wonder where Major is. Fewer people totally overwhelmed than last year. Lots of positive comments. Whew! Email Major that evening.

Tuesday: day 2 of tutorial. Major shows up! — he had the dates wrong in his diary. Spend much of day listening to Kirk's anecdotes. Cool. Wonder how technical the group really was? Review evaluation forms for answers to "most useful part of tutorial". A tie between "kernel debugging" and "kernel internals". OK — they were pretty technical.

Wednesday: the conference! Wake up before 5:00am. Decide to double the number of slides in my talk. Takes twenty minutes to print the new slide.

Steve Landers arrives wearing a tie! — but I forgot my camera. Conference goes very smoothly. OK, smoothly. All of the talks were well received (and no-one fell asleep during mine). Dreaded YACS (Yet Another Curtin Student) talk actually very impressive. Several ladies present want to offer him a job :-). Craig Farrell voted "Best Local Speaker", wins gift voucher from Dymocks. All other speakers get an O'Reilly T-shirt from Woodslane.

Whew, it's over. Attendees run for the bar before Kirk's second talk begins. Kirk gives his talk in record time and the cocktail event starts. Rest of evening passes in a sort of blur (and I drank less than at the AUUG93 cocktail party!).

Thursday: drive Kirk to airport for his next pitstop. Arrive home. Contemplate empty letterbox. So was it worth it? Am I going to put myself through all that again next year?

You bet!

For the record, the speakers at the Perth conference were: Kirk McKusick, who gave two talks, "What's new in 4.4BSD" and "The "Free" Software Phenomenon"; Steve Landers, "GUI Development in the 90s"; James ("YACS") Mercer, "Monitoring Network Load Accurately"; Craig Farrell, "Genetic Algorithm based Network Partitioning"; Adrian Booth, "Delving into the UNIX Kernel"; Toivo Pedaste, "IP: The Next Generation"; and Major, "Internet Resource Discovery Tools". We ended up with 48 attendees plus these speakers.

The conference was sponsored by Silicon Graphics, who sponsored the production of the conference proceedings, and Sun Microsystems, who sponsored the cocktail event. Dymocks Hay Street Mall provided several prizes, and were offering a 10% discount on their book stand to all attendees.

85% of the attendees who filled out an evaluation form said that they would attend again next year; looking forward to seeing you (and some new faces) there!

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

Impressions of the 1994 Perth AUUG Summer Technical Conference

Having arrived slightly late, I had to sit at the back, because there weren't enough seats at tables to go around. This meant I, and several others, had no access to the water jugs and found it more difficult to take notes. However it did mean I could count the delegates unobtrusively. Sometime before morning tea I counted 47 delegates, including 7 women — a record high percentage of approximately 15%.

There was one audible mobile phone, which attracted at least five calls. This was pretty annoying and must have been distracting to the speakers.

Nevertheless, the talks were excellent. Kirk McKusick started off, with an overview of the neat new features of 4.4BSD, from the log-structured filesystem to `nvi` — a rewrite of `vi` with all your favourite features plus more, including the ability to edit binary data and infinitely long lines, and to consistently allow multiple people to edit a file at once.

Kirk also explained some of the history behind the Berkeley developments, which was just about as interesting.

It seems the Berkeley group are going out with a bang, anyway.

Steve Landers described how his company, Functional Software, have invented a new interpretation of "RTFM". And more seriously, how they have used `[incr Tcl]` and `Tk` to add a GUI interface to their software seamlessly and elegantly, without disturbing the existing character-based interface (which is itself pretty nice). As far as I can see, the GUI doesn't provide any more or less functionality — it just looks more 90s, and therefore makes the software more marketable.

`Tcl` (an embeddable command language) and `Tk` (a `Tcl`-driven widget toolkit) do for the X Window System what shells, awk and Perl do for Unix. `[incr Tcl]` is an object-oriented variant of `Tcl`, more suitable for large projects like Steve's.

`Tcl`, its variants, and `Tk` are all freely available over the net. Rumour has it that there may be a tutorial on them at AUUG'94.

After morning tea James Mercer, a postgraduate student in Computing Science at Curtin University, presented the latest addition to their suite of network monitoring tools: `loadman` joins `etherman`, `interman`, et al. (I don't understand why these things are suffixed "man" and not "mon".)

`loadman` is an all-out attempt at the difficult task of accurately measuring, logging, reporting and graphing the load on an Ethernet. It was unclear to me whether James could actually prove that `loadman`'s measurements were correct.

Craig Farrell, systems manager in James's department, then explained how he has implemented a genetic algorithm for partitioning networks. If you've decided to reduce your network's load by splitting it into segments, it can be hard to figure out a distribution of hosts that will spread the load evenly. Starting with a random population of possible distributions, Craig's software attempts to converge on an optimal distribution by simple "mutations". The "goodness" of a distribution is measured by a function of the expected (or previously measured) traffic between each pair of hosts.

With `loadman` to measure the traffic, this software could make network reorganisations less of a black art. An even more interesting idea was that the functionality could be built in to a network hub, which would repartition the network regularly to cope with changing traffic profiles.

Lunch was excellent, as usual. I guess this is why we keep going back to the Orchard Hotel even though the room really isn't the right shape for the presentations.

After lunch, Adrian Booth spoke on *Delving into the Unix kernel.* I expected this to be about *how* to get data out of the kernel, a subject on which Adrian has previously spoken at a WAUG meeting, but it turned out to be about *why* a Unix systems administrator needs some understanding of the inner workings of the kernel. Such knowledge will help you diagnose problems correctly and avoid having to call your vendor's support line, who will probably advise you to put in more memory or spend buckets of money and time sending the damn thing half way round the world (like to Sydney) to be fixed, when all it needs is a kernel patch, available by FTP.

It's a shame Adrian couldn't have given this talk *before* Kirk's 4.4BSD internals tutorial, which had been held over the two previous days.

Toivo Pedaste, from the University of Western Australia, managed to detach himself from cyberspace long enough to describe the research being done, largely under the auspices of the Internet Society, to find a way to overcome the limitations of the existing IP (Internet Protocol) address space.

I thought Toivo spent rather too long on background material, such as the organisation of the Internet Society and its various task forces and working parties, and didn't leave himself enough time to cover the IP addressing work. However the background material was pretty interesting in itself.

Following afternoon tea Major took us on a trip into cyberspace with an excellent explanation of how to use Gopher, WAIS and World Wide Web to track down information on the Internet. I thought this was a good conference-closing talk.

After a break, the February WAUG meeting was held, with an international guest speaker! — Kirk McKusick, sharing his knowledge and opinions on free software. This talk was about the various conceptions and collections of free software, the reasons people write it, and the development of the free software movement. Kirk also discussed the relative merits of free and commercial software.

Kirk didn't say this, but I am increasingly beginning to believe that for software, the correct heuristic is the inverse of the usual "you get what you pay for": the more expensive software is, the worse it is likely to be.

I was disappointed at the number of people who did not stay around for Kirk's talk. I suspect the break between the conference proper and the WAUG meeting was too long.

The finale was a "cocktail event", sponsored by WAUG and Sun Microsystems. (Why are beer, wine, juice and nibbles called "cocktails"? Maybe you're supposed to mix them together.) The break between the talk and the arrival of the drinks was also too long, and a number of people seemed to have drifted away. Still, we chatted for a while and eventually a few of us ended up at a cafe, where the main topic of conversation seemed to be the newsreader nn (more terrific free software!), probably because Peter Wemm, the maintainer of nn, was present.

I thought the conference was particularly well organised this year. (Adrian is getting the hang of it. Actually, I'm a bit worried that he is becoming obsessed with it!) The talks ran on time — in fact one of the tea breaks was even longer than planned, giving us more time to look at Dymocks' large book display. The professional-looking proceedings, again sponsored by Silicon Graphics, contained papers from almost all of the speakers. Nametags were provided, I think for the first time.

The content and presentation of the talks was really good too; I gave the conference a 100% "worth bothering" rating, which is pretty good from someone as easily bored and impatient with bad speakers as me!

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

AUUG NSW Chapter

A significant chunk of 1994 has slipped by already and many AUUGN deadlines have come and gone since the NSW chapter was formed. Hopefully this report will set a precedent and regular NSW chapter reports will appear on these pages.

The NSW Chapter has been meeting regularly each month at the Novotel Hotel in Darling Harbour, nominally on the second Tuesday of the month at 7pm. So far we have had talks on WABI from Sun Microsystems, Versant Object Oriented Databases from ITC, CA-Unicentre from Computer Associates, and a very controversial talk about Windows NT from IT.ConeXions.

The Summer Conference is only just over. From my biased perspective as an organiser it was a great success. There will be an independent review printed in AUUGN soon. The conference was held in North Sydney at the Chamber of Manufactures. The Chamber made us very welcome and provided great venue. Phil McCrea raved about the lunch at least four times, so the catering had the President's stamp of approval.

There was a wide range of papers presented at the conferences, some of which will be printed in AUUGN in the coming months. Kirk McKusick's tutorial and paper on BSD4.4 was the major focus of the conference. As the final session Kirk McKusick, Berny Goodheart and Jamie Honan formed a panel to discuss free software. The panel discussion ran well overtime as Kirk and Berny colourfully detailed the intrigue of the recent Unix courtroom dramas.

The papers presented were:

Keynote Address: *What's new in 4.4 BSD*, **Kirk McKusick**
Internet Firewalls, **Tony McGrath**, Uniq Professional Services
I couldnt survive without my internet, **Charles Cave**, Unidata Australia
NIS: Friend or Foe, **Peter Gray**, University of Wollongong
Making Mission Critical Client/Server a Reality, **Judy Potter**, Legent Australia
UNIX Testing tools, **Peter Chubb**, Softway
Porting with Gcc, **Frank Crawford**, ANSAMS
Implementation of a Document Management System, **Phil Barton**, Prospect Electricity

After the Conference a very quick Chapter Annual General Meeting was held. The NSW Chapter now has an official elected committee comprising:

David Purdue
Peter Chubb
Brenda Parsons (Secretary/Treasurer)
Julian Dryden (Chairman)

Over the coming year we plan to continue the monthly meetings. We are always on the lookout for people to give presentations, so if you have something to talk about please drop us a line. We are also looking at other activities that may interest members. One such event is a combined AUUG Chapter videoconference multicast.

Announcements about NSW activities are posted out and emailed to all NSW AUUG members. If for any reason you are not receiving Chapter announcements please contact one of the committee members.

For further information please feel free to contact:

Julian Dryden on (02) 809 9345 (bh), julian@dwt.csiro.au, or
Brenda Parsons on (018) 647 259, (02) 808 2797 (fax), bdp@sydney.dialix.oz.au

Julian.

AUUG

UNIX AND OPEN SYSTEMS USERS

AARNet

Mail Service

Dear Site Administrator,

As you may be aware, the arrangements for mailing to addresses outside Australia (and also to AARNet sites) changed in May 1991. Since then, the University of Melbourne are no longer managing the administrative details associated with maintaining this service. The AARNet (Australian Academic and Research Network) management has taken over administering the service, and are requiring all ACSnet and similar sites to register with AARNet and pay a fee for continued access to Internet mail services. AARNet have set this fee as $1000 per annum for most sites, with larger sites paying more (you know who you are).

The fee is intended to cover use of AARNet bandwidth for your network traffic. Registration with AARNet, however, provides ONLY the registration of your address in worldwide address tables - your site will be unreachable without this registration. The fee does NOT cover the costs involved in obtaining a connection to AARNet or ACSnet NOR does it include a guarantee that you can be connected or even to help you find a connection point. See Note B for some information about connection services.

AUUG as a service to its members has negotiated with AARNet to achieve a lower price for this basic address registration service. The lower price is based on the reduction in paperwork for the AARNet management authorities. The AUUG/AARNet fee is dependent on the membership status of the owner of the machine(s)/domain involved, and is currently $250 for members and $600 for non-members. As such it is a substantial discount on the AARNet fee, but only applies to sites in the AARNet $1000 category. Larger sites will need to negotiate directly with AARNet.

The address registration is for one AUUG membership year. Membership years start on the 1st January or July, whichever is nearest to receipt of your application. Sites which do not renew their AUUG/AARNet registration annually with their AUUG membership each year will be removed from the Internet tables and will no longer be able to communicate with international and AARNet hosts. Reminders/invoices will be sent along with your membership renewal.

The required initial registration form is attached below. It should be completed and forwarded to AUUG's (postal) mailing address at the bottom of the form or faxed to (02) 332 4066. If you have any queries on the AUUG/AARNet arrangements please direct them to the AUUG office on (02) 361 5994.

Regards,
Chris Maltby
AUUG-AARNET Administrator
AUUG Inc.

AARNet

Mail Service Application

On behalf of the organisation listed below I wish to apply to be a Mail Service Affiliate Member of AARNet, and accordingly request that AUUG Incorporated arrange for the Australian Vice-Chancellors' Committee (AVCC) to maintain on my behalf an electronic mail delivery record in the Australian Academic and Research Network (AARNet) to allow my organisation to send and receive electronic mail carried across AARNet.

I understand that the AVCC may consult the recorded logs of my organisation's usage of AARNet facilities for 1990, and determine that I am ineligible for registration under the terms of the agreement between AVCC and AUUG Inc. I understand that AUUG Inc will invoice my organisation for this service for the calendar year 1991 and for subsequent years unless it receives my organisation's written advice to terminate the Affiliate Membership of AARNet.

I understand that the AVCC and AUUG Inc maintain the right to vary the Mail Service Affiliate Membership charges from year to year, and maintains the right to cease offering this service to my organisation at the start of any year, at their discretion. I understand that in the event of any variation of the Mail Service Affiliate Membership of AARNet, my organisation will be advised in writing by the AVCC or AUUG Inc to the address below.

I understand that in consideration of the AARNet Mail Service Affiliate Membership charge, AARNet will undertake to maintain a mail directory entry which will direct incoming electronic mail to the AARNet gateway system(s) which I have nominated below. Furthermore I accept that there is no other undertaking made by AARNet in terms of reliability of mail delivery or any other form of undertaking by AARNet or the AVCC in consideration of the payment to AARNet for the maintenance of the mail directory entry on AARNet.

I undertake that my organisation's use of the mail delivery services over AARNet will not be used as a common commercial carrier service between my organisation and other organisations receiving similar services from AARNet, nor will it be used as a commercial carrier service between branches of my organisation. Furthermore my organisation undertakes to use AARNet facilities within the terms and conditions stated in the AARNet Acceptable Use Policy. I accept the right of the AVCC or AUUG Inc to immediately terminate this service at their discretion if these undertakings are abused by my organisation (where the AVCC retains the right to determine what constitutes such abuse).

I understand that a fee is payable with this application: of $250 if the host/hosts covered are owned by a member of AUUG Incorporated, or $600 if the host/hosts covered are not owned by an AUUG member. Corporation host owners may only claim the member price if the corporation is an Institutional member of AUUG Inc. My cheque payment of either $250 or $600 as appropriate is enclosed with this application.

AUUG

AARNet

MAIL SERVICE
AFFILIATE MEMBERSHIP APPLICATION FORM

PLEASE PRINT CLEARLY!

Date: ____________________

Name of Organisation/Owner: ____

____

Signed: ______________________________ AUUG Membership No (if known): __________

Name: ______________________________ Position: ______________________________

on behalf of the organisation named above.

Address: ____

____

____

____________________________________ Postcode: ____________________

Administrative Contact: ________________________ Title: ____________________

E-Mail: ______________________________ Phone: ( ) ____________

Fax: ( ) ____________

Technical Contact: ______________________________ Title: ____________________

E-Mail: ______________________________ Phone: ( ) ____________

Fax: ( ) ____________

Mail Delivery Information to be entered in AARNet (see Note A next page)

Domain Names Requested: ____

____

Gateway Addresses: ____

____

____

Expected Link Protocol: UUCP SL/IP MHSnet Other: ______________

* * * *

Send this page only to:

AUUG Incorporated
PO Box 366
Kensington NSW 2033

Phone: +61 2 361 5994
Fax: +61 2 332 4066

MAIL SERVICE
AFFILIATE MEMBERSHIP
APPLICATION FORM cont'd.

Note A. Mail Delivery Information

Two items of information are required: firstly the preferred name of your mail host (or the domain name(s) of a group of hosts) in Internet domain name system format, and secondly the name (or names) or AARNet gateway systems who will accept electronic mail over AARNet (and connected overseas networks) on your behalf and forward it to you. The primary requirement for an AARNet gateway is its ability to recognise your host/domain addresses and perform the necessary mail header rewriting reliably.

Please check with the postmaster at your preferred AARNet gateway host site before citing them as a gateway for AARNet mail delivery. For ACSnet addresses (*.oz.au), the host "munnari.oz.au" (Melbourne University) is a recommended gateway. Other possible sites include "metro.ucc.su.oz.au" (Sydney University), sirius.ucs.adelaide.edu.au (University of Adelaide), uniwa.uwa.oz.au (University of WA) and bunyip.cc.uq.oz.au (University of Qld). Note that all gateway addresses must be fully domain qualified.

Example Mail Directory Information request:

Mail addresses required:	acme.oz.au, *.acme.oz.au
Mail Gateways (primary)	gw.somewhere.edu.au
(secondary)	munnari.oz.au
(secondary)	unnet.uu.net

The addressability of your site and the willingness of your nominated gateways to act in that capacity will be determined before registration proceeds. Processing will be made faster if you contact the postmaster at your nominated gateways in advance to inform them of your intentions. Your nominated technical contact will be notified by email when registration is complete.

Note B. Getting Connected

New sites will need to find an existing AARNet or ACSnet site who will accept their site as a connection, and also select a protocol for transferring data over their mutual link. Although the UUCP package is a standard inclusion with UNIX, it is little used in Australia due to its relatively poor performance. Other possible choices for your link protocol include SLIP (TCP/IP) and MHSnet.

Among a number of organisations who provide connection services, Message Handling Systems Pty Ltd have announced a special offer on both their link software and connect time for AUUG members. For more details on this offer, contact Message Handling Systems on (02) 550 4448 or elaine.mhs.oz.au.

Book Reviews

Welcome to the book reviews for this edition of AUUGN. Unfortunately, it isn't as spectacular as the last one, but still very useful. We have reviews of a few more Nutshell books and the first of a couple of reviews of books from SunSoft Press. We also have two reviews on *The X Resource: Issue 9.* This came about as we had a copy for review and, at the same time, a review was offered for publication. I'm always happy to publish reviews on any books you may have read.

One other interesting activity I participated in was the launch of Berny Goodheart's book *The Magic Garden Explained,* which was reviewed in the last issue of AUUGN. Some mention of this is made the President's Report.

If you are interested in reviewing any books, we receive a fair number, mainly from Prentice Hall and O'Reilly and Associates, keep watching for notices. The current practice is to post a note to the newsgroup *aus.org.auug* when we have new books available. Unfortunately, this disadvantages members without network connections, or on the end of a low speed link. For people in such a position, either mail, via the AUUG PO Box, or fax me on (02) 717 9429, with your contact details and preferences.

Frank Crawford

The X Resource: Issue 9
Proceedings 8th Annual X Technical Conference

Edited by Adrian Nye
O'Reilly & Associates
ISSN 1058-5591

Reviewed by
Michael Werner
Department of Physics
University of Queensland
<werner@physics.uq.oz.au>

The X Resource is a journal published four times per year. Issue 9 of the X Resource contains the proceedings of the 8th Annual X technical conference held in Boston, Massachusetts during January 24-26 1994. As such this journal contains up-to-date information on X issues with Issue 9 containing 19 papers and 4 abstracts. On the back cover it states *The X Resource provides timely, in-depth coverage of the issues and techniques in X programming, administration, and use.* Of interest to many users at the moment are the extensions and new facilities to be provided by X11R6. This volume provides an insight into current research in addressing some of the shortfalls of the current implementation of the X protocol and toolkits. With such a wide range of topics, anyone interested in X development will find at least some of the papers of interest. A list of the themes is as follows:

> *Use Case Driven Design, Formal specification and testing applied to Xt toolkits, Trait abstraction in Motif, Extending Xt to support CORBA embedding, Redisplaying objects in Fresco, Multi-rendering in the Silicon Graphics X Server, A true multiple screen X Server, Xwindows Image Extension, Low Bandwidth X, X Keyboard Extension, Inter-Client Communication in X11R6, The new Session Management Protocol, Distributed Management of the X colour resources, The Network Audio System, X Servers in 3D space, From X protocol multiplexing to X protocol multicasting, new font technology in X11R6, extending X for recording, Kerberos authentication of X connections.*

As you can see, the ideas put forward are comprehensive in their breadth. I have outlined below some of the papers to give the flavour of the material presented:

1. A description of a multi-rendering implementation in the SGI X server, available in IRIX 5.1, to support OpenGL and PEX without compromising interactivity. The authors argue that the synchronisation overheads in multi-threaded X servers are too high. It claims that the SGI multi-rendering is an effective implementation whose mechanisms allow parallelism without the overhead of

locking nearly all X server data structures. The authors outline the relationships between OpenGL, PEX and the X Server. They also discuss some of the system support useful in implementing multi-rendering. This is an interesting paper in its own right but especially so to any SGI box user. In particular, the discussion of how best to increase response of the system (the X interactivity) has a much broader significance in the light of current trends to implement kernel scheduled light-weight processes.

2. Experiments on Low Bandwidth X (LBX) are discussed by Keith Packard of NCD. This is an important area of research as anyone who has run X over a SLIP connection can testify. The system described requires an LBX proxy and the obligatory changes to the X server to handle compression, *etc.* The LBX proxy presents a standard X service to the network. The author discusses some of the problems in the current X protocol and suggests ways to reduce the amount of information required to be transmitted between client and server as well as compression. Caching techniques such as LRU caching of Drawables and GC IDs also look very promising. This paper describes an X Consortium standard so unlike the SGI specific X Server modifications discussed above this paper is bound to have a wide audience.

3. New Font Technology in X11R6 presented by Nathan Meyers of HP. This paper describes enhancements to the X font rendering and server system. A sample authorisation protocol for font servers in X11R6 is presented which can serve as a template for a more complete font licensing system. Also discussed in more detail are the addition of matrix XLFD enhancements allowing simple unidirectional scaling or special effects through affine transformation matrices. These are specified within scalable aliases. The paper discusses how clients can use the transformable fonts to render non-horizontal text. It also introduces the use of charset subsetting in X11R6 to save computations and glyph caching to reduce the X server's font memory requirements. A look at this paper is a definite must for X application developers.

4. Kerberos Authentication of X connections. A new authorisation protocol, KERBEROS-V5-1 has been added to the existing MIT-MAGIC-COOKIE-1, XDM-AUTHENTICATION-1 and SUN-DES-1 protocols. This is for the Kerberos Authentication Service (V5) described in RFC1510. The system administrator will notice some changes to the format of /etc/X0.hosts and users for xhost in X11R6 which will have prefixes LOCAL,INET, DNET,NIS and KRB to specify the address family. Also discussed are the problems in implementing end-to-end encryption due to Xlib maintaining multiple queues. The authors decided to leave it for future work however.

Having wet your appetite with outlines for only 4 out of 19 papers, how could you resist buying this quarterly journal :-) In summary, this issue provides an excellent look at extensions to be seen in X11R6 and addresses a wide range of topics on X.

The X Resource: Issue 9
Proceedings 8th Annual X Technical Conference

by O'Reilly & Associates (Ed)
O'Reilly & Associates
1994, 253 pages, $44.95
ISBN 1-56592-066-X

Reviewed by
J. Wright
Guru Software Services
<Jon.Wright@Citibank.com.au>

"The X Resource" is a specialist journal that covers issues and techniques that are related to programming or administering an X-based environment. It is available in single issue form and also as a regular journal. For more information about the journal, contact the publishers.

This review is about a single volume: "Issue 9: Proceedings of the 8th Annual X Technical Conference". As the name implies, this issue is actually a complete set of conference proceedings rather than a typical journal. For that reason, it is worth investigation as a possible resource for those working with X-windows. Note also that there is still a brief editorial and the end of the journal contains details about previous issue, but the balance of the contents relate to the conference.

The conference was held in January 1994 and there were a series of distinguished speakers. There was a wide range of topics with something to interest everyone. Topics contained in the collection include: design, technology, testing and user stories. Some speakers unfortunately only provided an abstract but the complete papers include (in presentation order):

- Bilow, S.C.; Use Cases, Objects and X
- Nath, S.; Zero-Defect Widgets
- Dardailer, D.; Traitifying Motif
- Price, C.; Extending Xt to Support CORBA-Based Embedding
- Linton, M. *et.al.*; Redisplay in Fresco
- Kilgard, M.J. *et.al.*; X Server Multi-rendering for OpenGL and PEX
- Jones, P.C.; Xvan: A True Multiple Screen X Server
- Fahy, J.B.; Experience with XIE
- Packard, K.; Designing LBX
- Fortune, E.; The X Keyboard (XKB) Extension
- Marks, S.W.; Inter-Client Communication In X11R6 and Beyond
- Wexler, M.; XSMP: The New Session Management Protocol
- Main, F.B.; The POSC/Halliburton Shared Colormap System
- Fulton, J. *et.al.*; The Network Audio System
- Dykstra, P.; X11 in Virtual Environments
- Meyers, N.; New Font Technology for X11R6
- Zimet, M.; Extending X For Recording
- Yu, T.; Kerberos Authentication of X Connections

Overall, I would rate this book as a valuable reference for all those of us who did not actually attend the conference.

Understanding Japanese Information Processing

by Ken Lunde of Adobe Systems Inc.
O'Reilly and Associates
1993
ISBN 1-56592-043-0

Reviewed by
Greg Doherty
Computer Science
Wollongong University
<greg@cs.uow.edu.au>

I offered to review this book as a computer scientist almost totally ignorant of the Japanese language, despite having shared an office with Sadayuki Murashima for a few months, and now finding myself the father of two children whose forays into the language have inspired me to try to obtain a basic understanding for myself.

>From the preface:

> *Expect to find plenty of platform-independent information and discussions on Japanese character sets, how Japanese is encoded and handled on computer systems, and basic guidelines and tips for developing software targeted for the Japanese market. ....*

and

> *It is my intention that this book become the definitive source for information relating to Japanese information processing issues.*

How well did Ken Lunde succeed with his stated aims?

Firstly, he describes with clarity the problems posed by the existence of multiple writing systems, and what will be required to read and write Japanese effectively, then moves on to discuss the JIS standards, which it is common knowledge requires two bytes per character for the thousands of kanji characters in the standards, and then to the input problems these large character sets pose. Then he describes in detail the three basic Japanese encoding methods. JIS, with its two byte escape escape sequences to shift character sets, shift-JIS developed by Microsoft, and EUC (Extended UNIX Code) which as you would guess from the name is used mainly in UNIX systems.

Software developed for the difficult task of keyboard input is then addressed. The most frequently used method is a pronunciation of individual kanji characters via a Roman Input String or Kana Input String which will generate a list of possible candidates allowing the user to select the target character. Unambiguous character input can be achieved using the encoded value of the target character, which must be memorised or looked up on each occasion it is needed. Descriptions of the great variety of keyboard arrangements for use in inputing various alphabets give the novice a feel for what would be required to set up an efficient input mode to satisfy most users.

How to output Japanese characters, where to find fonts, and visual aspects of the various fonts are discussed, mainly in the context of PostScript, not surprisingly since the author is an Adobe employee and PostScript printers abound. Conversion processes between the various encoding modes are given in a chapter of recipes for things you may need to do, and then a summary is provided of the Japanese data processing capabilities of the common operating systems and text editors for PCs and workstations. Finally, there is a short chapter on Japanese news and e-mail, including things to avoid such as 8 bit character encoding for transmission.

The last 150 pages of the book are appendices showing conversion tables for various character sets, where to find software packages and data sets, newsgroups to join. There is an extensive glossary of terms and a multi-lingual bibliography.

I found the writing style very clear. Certainly, it is an excellent entry level guide to the variety of different questions which have to be addressed in order to produce a Japanese data processing capability. I have not had time to sit down and work through the material in detail. In our family, I think that task will fall to my daughter, who is a serious student of both Japanese and Computer Science. On my reading, I would recommend the book to the curious amateur and the would-be professional alike. Ken Lunde has achieved his stated aims in exemplary fashion.

Solaris Application Developer's Guide

by SunSoft
SunSoft Press, Prentice Hall
1993, 102 pages
ISBN 0-13-205097-8

Reviewed by
Adrian Booth
Adrian Booth Computing Consultants
<abcc@dialix.oz.au>

Despite its rather generic title, this book refers to itself internally as a Solaris **2.1** guide. Unfortunately Solaris 2.3 has been available for some time now, and Solaris 2.4 is just around the corner. Quite a lot has changed since Solaris 2.1.

The preface specifies the target audience as someone developing applications for Solaris 2.1 who is familiar with the SunOS system or BSD. It assumes that you are familiar with workstation use, a Unix system editor, and the Unix system directory and file structure. So if you're a DOS, Windows, or any non-Unix developer looking for a starting point before plunging into Solaris 2, this book won't help.

Instead, it is aimed squarely at developers who are experienced at writing Unix applications (particularly SunOS 4.1.x applications), but want an overview of Solaris 2.

Chapter 1 provides a brief (5 page) overview of Solaris 2.1, including SunOS 5.1, how existing applications can be made to run under Solaris 2.1, and application packaging and installation.

Chapter 2 describes the "Solaris 2.1 Application Development Environment" in 11 pages. It first describes the language products available from Sun for Solaris 2.1 (C, C++, FORTRAN, Pascal), and then other development tools (SPARCworks Professional Development Environment, Compatibility and Migration information `lint` libraries, and several debuggers) - *all* of which are "available separately from the Solaris 2.1 product". It also briefly touches on the Extensible Linking Format (ELF), SPARC Assembly Language, dynamic linking and shared libraries, and the source and binary compatibility packages, referring you to another manual in most cases. Slightly more detail is provided about Open Windows, including more manuals to which you can refer and products you can purchase.

Chapter 3 describes ``System Services'' such as file and terminal I/O, processes, IPC, and file and record locking. This was quite a good overview. My only concerns were that it might be too basic for an experienced developer, and that many of the samples of C code wouldn't compile (missing closing braces at the end of `main()`, a program that starts `ude <stdio.h>` (*sic*)).

Chapter 4 describes "The Solaris 2.1 Integrated Environment''. It would have been welcome to see more attention paid to describing whether particular features were System V, System V.4, or Solaris 2.x specific. Realtime scheduling, the virtual file system, and virtual memory are briefly discussed. A smorgasbord of standards is mentioned - ABI, SCD, POSIX, XPG3, and DKI/DDI.

Windows are then discussed - unfortunately some of this information is already obsolete in Solaris 2.3 (with Sun's abandoning of NeWS), and Sun's adoption of Motif will make this information even less useful. Networking (network selection, NIS+, RPC/XDR, sockets and STREAMS, TLI and TIRPC) and Graphics (XGL, SunGKS, SunPHIGS, and PEX) are also briefly described and the appropriate manuals referenced.

Chapter 5, ``Networking Tutorials'' actually only contains one, porting an RPC application to use TIRPC. This was quite well written and would make a reasonable reference.

Chapter 6 provides a short overview of internationalising applications, and Chapter 7 an overview of ToolTalk (an interapplication messaging service).

There seems little in this book that hasn't already been supplied to developers by Sun as part of their migration strategy. It is also marred by being somewhat out-of-date and *very* poorly proof read. If however you haven't been supplied any of this information, and you're prepared to take it with a dose of salt (given its age), it makes a worthwhile read.

Managing UUCP and Usenet
10th Edition

by Tim O'Reilly and Grace Todino
O'Reilly & Associates, Inc.
1992, 342 Pages
ISBN: 0-937175-93-5

Reviewed by
Brenda Parsons
UNIX & Open Systems Consulting
<bdp@sydney.dialix.oz.au>

This is the 10th Edition of a book first published in 1986. It covers most everything that a System Administrator would need to know about setting up their first, second or third connection to the outside world.

A short history of the development of UUCP is given, and throughout the book, special attention is given to the various flavours of UUCP which exist, including BSD, SunOS, HoneyDanBer (BNU) and Xenix, plus appendices for DOS and Macintosh.

A nice explanation of RS-232, DTE, DCE and the handshaking which must occur is supplied, and while it doesn't explain how to make the cables, it gives enough information to do so.

The book is organised in a logical fashion, from the ground up, starting with getting your modem and serial ports configured, configuring the UUCP files, testing the initial UUCP links, analysing the log files and setting up security.

The next sections deal with the various flavours of Netnews Software, including where to get them, how to compile and install them, and how to maintain them.

There is now a section on NNTP (Network News Transfer Protocol) which acts as a replacement for UUCP over TCP/IP networks. The section includes all the usual stuff, like where to get it, how to compile it, and how to administer it.

Almost a third of the book is devoted to appendices. They cover an in-depth description of the UUCP working files (field by field), talking to modems, including the command sets, more on RS-232, DOS and Macintosh setups, and FAQ section, and a description of the UUCP 'G' Protocol.

Also, included in the appendices is a number of useful programs and shell scripts.

All in all, this book would be an excellent addition to the administrators bookshelf, especially for those of us that only have to set up these systems once every two or three years!

Power Programming with RPC

by John Bloomer
O'Reilly and Associates
ISBN 0-937175-77-3

Reviewed by
Ian Crakanthorp
ANSTO
<ian@atom.ansto.gov.au>

My project leader comes in and asks me if I know about RPC. "Remote Procedure Calling" I confidently reply, hoping he will not question me further. Aside from knowing that NFS uses it, and there are two standards floating around, my knowledge was not that great on the subject.

So I managed to obtain a copy of "Power Programming with RPC" to review, another book from the O'Reilly Nutshell handbook series. The Nutshell series of books I have previously read, I have found to be very good, and this book is no exception.

The author assumes you are familiar with the C programming language and with UNIX, though not an expert. He starts with a description of what RPC, the mechanisms it is built on, and a trivial example. The various standards in RPC are discussed, the Open Software Foundation's (OSF) Distributed Computing Environment (DCE), and Sun's Open Network Computing (ONC) RPC which he uses mostly through the book.

Then the book gets down to how to program using RPC, with a high level example application. As he steps through the example, he introduces the various library functions to give the reader an idea about how to program with RPC. Lower level RPC programming is discussed next covering more complex client and server communication, and the use of protocol compilers. A chapter is then dedicated to UNIX networking and Interprocess Communication (IPC). This gives the reader a good understanding of how everything is working below the application layer. The author uses this knowledge in later chapters. Not necessary though for a seasoned UNIX programmer.

The middle chapters cover application development. The first being a networked parallel image processing application. Building on the knowledge gained in the previous chapters, the author discusses some limitations of networking. Then steps through his example complete with source code, pointing out how to use the various RPC tools, how to compile the code, what traps to look out for, and some useful tips for young players. His next example was to distribute an existing application over the network using RPC.

The final chapters contain, how to handle multiple clients and servers, how to get applications to communicate in an asynchronous and concurrent fashions. How RPC can complement a windowing system such as X11. Some advanced programming issues that address security and authentication schemes, and the proposed future of RPC systems.

At the end of the book is a ONC RPC Programming reference. It offers complete detail on; the ONC XDR functions, version 4.0 portmapper binder service; and the RPC programming libraries.

In conclusion I found the book to be well set out and written. Anyone contemplating using RPC would find this book very useful. The book could even give you ideas of using RPC for an application you already have. I would recommend this book for anyone that is interested in RPC, or anyone just plain curious about it.

AUUG BOOK CLUB
&
PRENTICE HALL AUSTRALIA

20% DISCOUNT TO AUUG MEMBERS

Please send me a copy/copies of the following book

□ ***SOLARIS Application Developers Guide***
ISBN: 0132050978, Paper, 1993, RRP $57.95*

*Deduct 20% from listed retail price

Name: ______________________ Organisation: ______________________

Position: ____

Address: ____

______________________________ Telephone: ______________

□ Please send my book/s on 30-day approval (tick box)
Enclosed cheque for $ __________ (Payable to'Prentice Hall Australia')
□ Charge to me **OR** □ Company purchase Order No. ______________
Please charge my: □ Bankcard □ Visa □ MasterCard
Credit Card No: □□□□ □□□□ □□□□ □□□□

Expiry Date: ______________ Signature: ______________

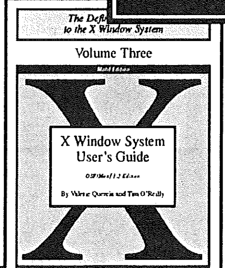
Mail or fax completed order form to Prentice Hall Australia, PO Box 151, Brookvale NSW

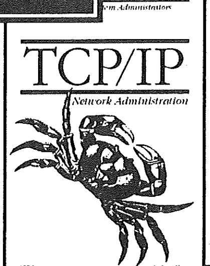
OR ☎ Use our FAST PHONE SERVICE by calling Liz Guthrie.
SYDNEY (02) 939 1333

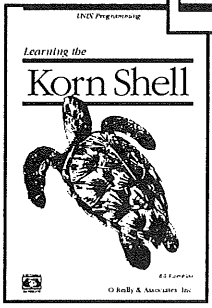
A.C.N. 000 383 406

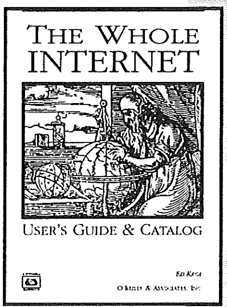
Prentice Hall Pty. Ltd.
7 Grosvenor Place, Brookvale NSW 2100.
Tel: (02) 939 1333 Fax: (02) 905 7934
A Paramount Communications Company

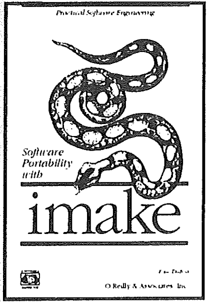
FROM O'REILLY & ASSOCIATES, INC.

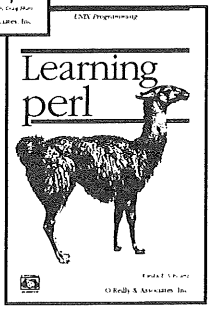

The Definitive Guide to the X Window System
Volume Three
X Window System User's Guide
By Valerie Quercia and Tim O'Reilly
O'Reilly & Associates, Inc.








Practical Software Engineering
Software Portability with
imake
O'Reilly & Associates, Inc.



Books that help people get more out of computers

The Whole Internet User's Guide & Catalogue 2\e

By Ed Krol
2nd Edition, May 1994
400 pages, ISBN 1-56592-063-5 Price: $49.95

The best book about the Internet just got better! This is the second edition of O'Reilly's comprehensive – and bestselling – introduction to the Internet, a resource of almost unimaginable wealth. In addition to email, file transfer, remote login, and network news, special attention is given to some new tools such as World Wide Web and its multimedia browser, Mosaic.

Learning the Korn Shell

By Bill Rosenblatt
1st Edition June 1993
363 pages, ISBN 1-56592-054-6 Price: $55.00

This new Nutshell Handbook is a thorough introduction to the Korn Shell, both as a user interface and as a programming language. It provides a clear explanation of the Korn Shell's features, including *ksh* string operations, co-processes, signals and signal handling, and command-line interpretation. *Learning the Korn Shell* also includes real-life programming examples and a Korn Shell debugger (*kshdb*).

Volume 3M: X Window System User's Guide Motif Edition

By Valerie Quercia & Tim O'Reilly
2nd Edition January 1993
956 pages, ISBN 1-56592-015-5 Price: $69.95

Orients the new user to window system concepts and provides detailed tutorials for many client programs, including the *xterm* terminal emulator and the *twm*, *uwm*, and *mwm* window managers. Later chapters explain how to customize the X environment. Revised for Motif 1.2 and X11 Release 5.

Software Portability with imake

By Paul DuBois
1st Edition July 1993
390 pages, ISBN 1-56592-055-4 Price $55.00

This new Nutshell Handbook—the only book available on *imake*—is ideal for X and UNIX programmers who want their software to be portable. The book is divided into two sections. The first section is a general explanation of *imake*, X configuration files, and how to write and debug an *imake* file. The second section describes how to write configuration files, and presents an architecture that allows development of coexisting sets of configuration files. Several sample sets of configuration files are described and are available free over the net.

Learning perl

By Randal L. Schwartz
1st Edition September1993 (est)
220 pages (est), ISBN 1-56592-042-2 Price: $49.95

Perl is rapidly becoming the "universal scripting language." Combining capabilities of the UNIX shell, the C programming language, *sed*, *awk*, and various other utilities, it has proved its use for tasks ranging from system administration to text processing and distributed computing. Learning perl is a step-by-step, hands-on tutorial designed to get you writing useful perl scripts as quickly as possible. In addition to countless code examples, there are numerous programming exercises, with full answers. For a comprehensive and detailed guide to programming with Perl, read O'Reilly's companion book *Programming perl.*

TCP/IP Network Administration

By Craig Hunt
1st Edition July 1992
502 pages, ISBN 0-937175-82-X Price: $59.95

A complete guide to setting up and running a TCP/IP network for practicing system administrators. Covers how to set up your network, how to configure impottant network applications including sendmail, and discusses troubleshooting and security. Covers BSD and System V TCP/IP implementations.

Available from all good bookstores or call WoodsLane Pty Ltd on (02) 979 5944 for the store nearest you

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
Trosmo	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.

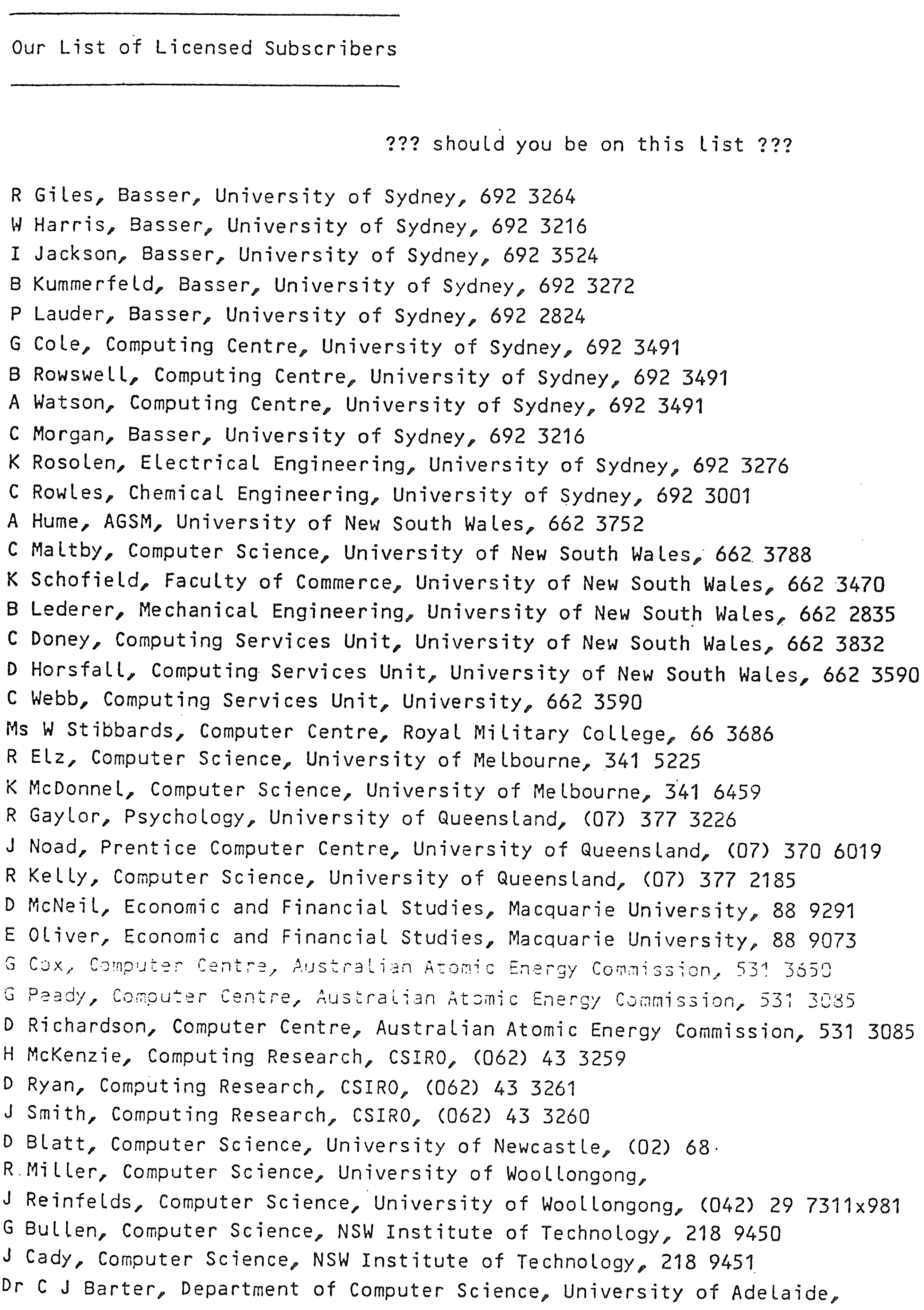
To take advantage of this offer send, in writing, to the AUUG Secretariat, a list of the publications, making sure that you specify the organisation, an indication of the priority and the delivery address as well as the billing address (if different).

AUUG Inc.
Open System Publication Order
PO Box 366
Kensington, NSW, 2033
AUSTRALIA

Fax: (02) 332 4066

!AUUGN

The following list of the original subscribers to AUUGN appeared in AUUGN Vol 1 Number 1. Are you on this list?



Our List of Licensed Subscribers

??? should you be on this list ???

R Giles, Basser, University of Sydney, 692 3264
W Harris, Basser, University of Sydney, 692 3216
I Jackson, Basser, University of Sydney, 692 3524
B Kummerfeld, Basser, University of Sydney, 692 3272
P Lauder, Basser, University of Sydney, 692 2824
G Cole, Computing Centre, University of Sydney, 692 3491
B Rowswell, Computing Centre, University of Sydney, 692 3491
A Watson, Computing Centre, University of Sydney, 692 3491
C Morgan, Basser, University of Sydney, 692 3216
K Rosolen, Electrical Engineering, University of Sydney, 692 3276
C Rowles, Chemical Engineering, University of Sydney, 692 3001
A Hume, AGSM, University of New South Wales, 662 3752
C Maltby, Computer Science, University of New South Wales, 662 3788
K Schofield, Faculty of Commerce, University of New South Wales, 662 3470
B Lederer, Mechanical Engineering, University of New South Wales, 662 2835
C Doney, Computing Services Unit, University of New South Wales, 662 3832
D Horsfall, Computing Services Unit, University of New South Wales, 662 3590
C Webb, Computing Services Unit, University, 662 3590
Ms W Stibbards, Computer Centre, Royal Military College, 66 3686
R Elz, Computer Science, University of Melbourne, 341 5225
K McDonnel, Computer Science, University of Melbourne, 341 6459
R Gaylor, Psychology, University of Queensland, (07) 377 3226
J Noad, Prentice Computer Centre, University of Queensland, (07) 370 6019
R Kelly, Computer Science, University of Queensland, (07) 377 2185
D McNeil, Economic and Financial Studies, Macquarie University, 88 9291
E Oliver, Economic and Financial Studies, Macquarie University, 88 9073
G Cox, Computer Centre, Australian Atomic Energy Commission, 531 3650
G Peady, Computer Centre, Australian Atomic Energy Commission, 531 3085
D Richardson, Computer Centre, Australian Atomic Energy Commission, 531 3085
H McKenzie, Computing Research, CSIRO, (062) 43 3259
D Ryan, Computing Research, CSIRO, (062) 43 3261
J Smith, Computing Research, CSIRO, (062) 43 3260
D Blatt, Computer Science, University of Newcastle, (02) 68
R Miller, Computer Science, University of Woollongong,
J Reinfelds, Computer Science, University of Woollongong, (042) 29 7311x981
G Bullen, Computer Science, NSW Institute of Technology, 218 9450
J Cady, Computer Science, NSW Institute of Technology, 218 9451
Dr C J Barter, Department of Computer Science, University of Adelaide,

The Electronic Interviews

Adrian Booth

Adrian Booth Computing Consultants

This, the second "Electronic Interview", is with **Piers Lauder** of Sydney University, who has been involved with UNIX from its earliest days in Australia. Piers was the programme chair for AUUG93, and is also a director of both Message Handling Systems (MHS) and BSDI (Australia).

Soon after Bob Kummerfeld prototyped what became known as the Sydney University Network (SUN-0) in 1979, Piers became involved and produced SUN-1 shortly afterwards. Bob and Piers worked together on it from then on. Bob proposed in 1983 that the SUN (at version 3 by this stage) and the then CSIRONET be combined to form the Australian Computer Science Network (ACSnet), still running the SUN software (which was store-and-forward). The ACSnet was in operation until it was replaced by AARNet.

More information on these early days of networking in Australia will be out shortly; look for an upcoming interview with **Bob Kummerfeld** coming soon to an AUUGN near you!

How did you first get involved with Unix?

I read the original CACM paper in October 1974. I had then recently migrated to Australia, and was working on a very strange machine in the Basser Department of Computer Science called a CDC 1700 that had no operating system to speak of, and thought UNIX might be the answer. We rented time on a PDP-11/45 in the Department of Electrical Engineering to run UNIX for a few hours/week and gain experience, and one post graduate student actually started using it to write his thesis (`nroff` was a revelation then).

The Department subsequently decided to scrap the CDC 1700, and I persuaded the then temporary head of the Department, Jan Hext, to buy a PDP-11/40 on which to run UNIX. It had 32K memory, and one RK05 - a removable disk holding 2.5 Mb. We also bought an new DEC multiplexor with 8 lines, and I started writing a driver for it on UNIX.

What was the main attraction of UNIX as described in the CACM paper?

The whole paper made me itch to get hold of Unix and start playing.

It was short paper that purported to describe an entire operating system. The system described was fascinatingly simple and elegant, and it ran on hardware a Computer Science department could afford. It offered typesetting, a powerful programming language that was also used to build the kernel, and "tools" that could be joined together by "pipes". Best of all, the idea of devices as names in the file-system name space offered salvation to anyone who had grown frustrated writing device-specific code on earlier operating systems.

What else can you tell me about your first years using Unix?

We (Basser) got involved with Unix for the same reason it became so popular at other Australian universities - it freed us from the tyranny of the computer centre. The computer centres represented centralised, arrogant, big iron computing, with nasty user access mechanisms that tended to shut down suddenly when one's credit expired. (Which of course led directly to our development of the SHARE scheduler now marketed by Softway.)

However, what I most remember was the sheer fun of it. I met many friends through Unix - our collaboration with UNSW led to the first small beginning of ACSnet (to share files, rather than mail - which came later). The ability to read the source to fix problems, to extend Unix's usefulness into teaching support was irresistable. In retrospect, we probably spent too much time speeding up the kernel, but in those days people time was far cheaper than hardware.

When did you first start getting involved with AUUG?
Can you name some other "names" involved with Unix and AUUG at the time?

I remember the first meeting that John Lions held at UNSW, a group that became the kernel of what was later to become AUUG. I can't quite remember the date, John could fill you in on that, but I remember talking with pride about my multiplexor driver that enabled our 11/40 to support 16 terminals. Lots of people liked to talk about terminal drivers, particularly Robert Elz - it became a joke at early AUUG meetings. Though Robert wasn't at that meeting, I think I remember Andrew Hume (now at Bell), Ian Johnson (went to Bell, then Sequent), Chris Maltby (director of Softway) and Greg Rose (now everywhere).

The meetings had a conspiratorial aura - we were pushing back the frontiers of what could be done with small computers, and were forging unexpected inter-disciplinary connections (many early Unix pioneers came from departments of psychology, for some reason.)

One of the hallmarks of the computer networks pioneered by Unix was the hierarchy busting nature of electronic communications. Programmers in one hierarchy could talk directly to programmers in another hierarchy, something that didn't happen before. In the Internet, anyone can ignore any hierarchy now.

Are there any other events (important and/or funny) that stand out from the early days?

What stands out now is the team spirit that existed between Sydney Uni and UNSW in the late 70s to develop UNIX to support teaching computer science. The work culminated in a version of UNIX we called Level 6+ that ran on PDP-11 computers and supported far more simultaneous users than was possible with the basic version. I think John Lions had classes of 15 students at a time using one PDP-11/40. The work was carried over into Level 7, and then 32V on the first VAX computers - Basser had a VAX with 80 active students at a time - a number that just raised incredulous smiles when reported to people at American Universities (I think they merely pitied our lack of funds :-). The never ending refinements to make the hardware stretch ever further both helped UNIX grow, and also encouraged further use of UNIX (the native operating systems like VMS would never have supported the load) and led to really useful developments like the SHARE scheduler.

Could you describe your subsequent involvements with Unix and the Sydney University Network?

The joint work between UNSW and Basser led to a need to share operating system source. Rather than transport RK05 disks between sites, we decided to share files using modems. In those days "modem" meant 300 baud, but UNIX source files were much smaller to match. Ian Johnson developed an early file-transfer program that copied data unchecked and mostly worked. I added a protocol for error correction. Then Bob Kummerfeld read about early work on UUCP at the Labs, and tried experimenting with back to back versions of "lpd" (the line printer daemon). One attempt caused a huge core dump that hung the VAX for about 20 minutes while it swapped between dumping the core, and swapping in other processes, and I decided to improve things. Which became SUN I.

As I learned more about networking and routing algorithms, SUN improved through versions II and III and began to spread throughout Australia. Somewhere along the line we called the resulting network ACSnet, and invented "domain" addressing, giving Australia the two-letter code "oz".

Then Bob obtained funding for further development, first from CSIRO, and later from Telecom, and we designed and built MHSnet from scratch, benefiting hugely having learnt from our mistakes in SUN.

One of my major worries in those days was the cost of the routing algorithm. Early versions of SUN used an Order N**3 algorithm, which quickly became a major CPU cycle stealer across Australia. MHSnet's algorithm is of course order N**2, but advances in computer science like "skip-lists" have improved the performance even further. These days CPUs are getting faster even faster than network growth, so I've ceased to worry.

In 1982 I visited Bell Labs, working in Berkeley Tague's group on early Ethernet software, and later with Rob Pike on his "Blit" terminal, for which I wrote the "layers" software that was distributed in System V. Bell Labs subsequently donated 12 Blit terminals to Basser and we had windows to play with from the early 80s. The best legacy from those days still in use is "sam" (a bitmapped-display editor), which students at Basser learn to use from year 1.

I returned to Bell in 1986 to work in the research department where I ported the SHARE scheduler into UNICOS (Cray's version of UNIX). There was nothing quite like the feeling of power one had when rebooting UNIX on the world's faster supercomputer!

What do you see as the current strengths and weaknesses of AUUG?

I think AUUG needs to become more of an open systems trade show, without, of course, losing sight of its strong support amongst the UNIX programmers.

There is a much greater market for publishing the intricacies of open systems than in dealing with source code -- which I feel will become an increasingly smaller part of the picture. System administration woes apart, the growth will probably come from applications, and their interaction on the "information superhighway". (Where UNIX is well positioned.) AUUG's weakness comes from the conflict between its traditional supporters, and the new open systems/commercial trend. But we've known that for years, ever since UniForum split from USENIX I suppose. Then again, maybe all it takes to reconcile the different camps is a really good last day speaker :-)

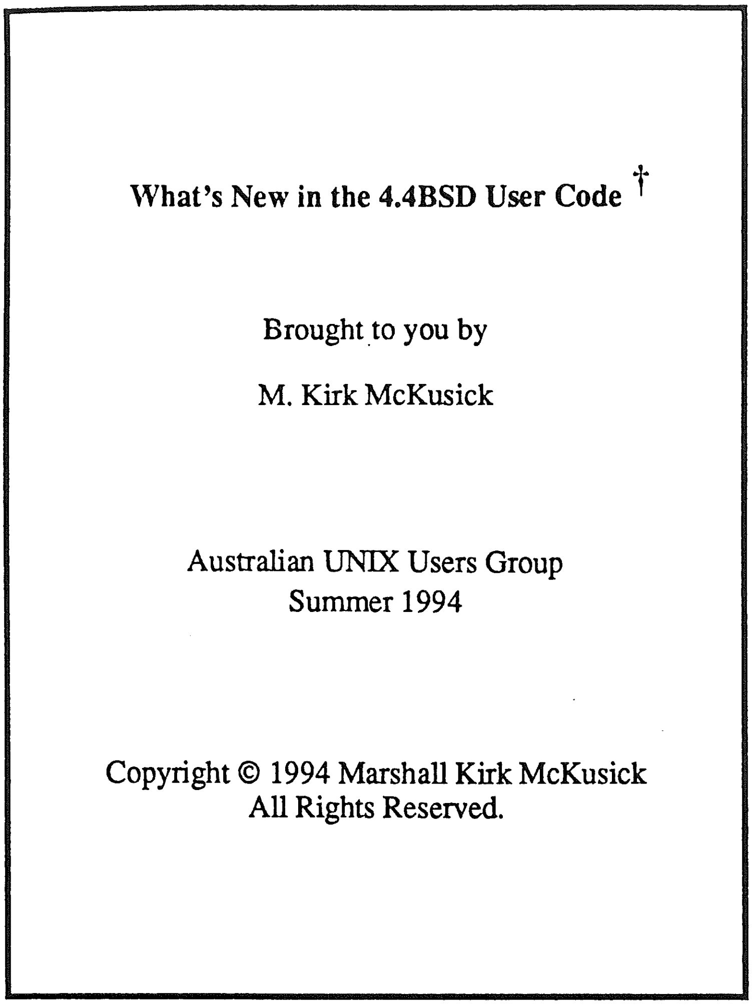
What about its future?

The same as the future of UNIX!

Are you prepared to make any predictions as to the future of Unix and/or the computing industry in general?

I think the growth of the Internet into people's homes will have a heavy influence. Of course, as soon as the telco's see the money to be made, they will muscle in on the provider side, but meanwhile our homes will need very fast and complex *multiprocessing* systems to handle the multiple channels of information.

So, I see the provision of fast multi-media platforms into the house providing a place for UNIX in the home (buried under all the user interface software), while it will also have a place in providing the services that people demand (it will surely be the O.S. of choice when providing video-on-demand systems for consumers).

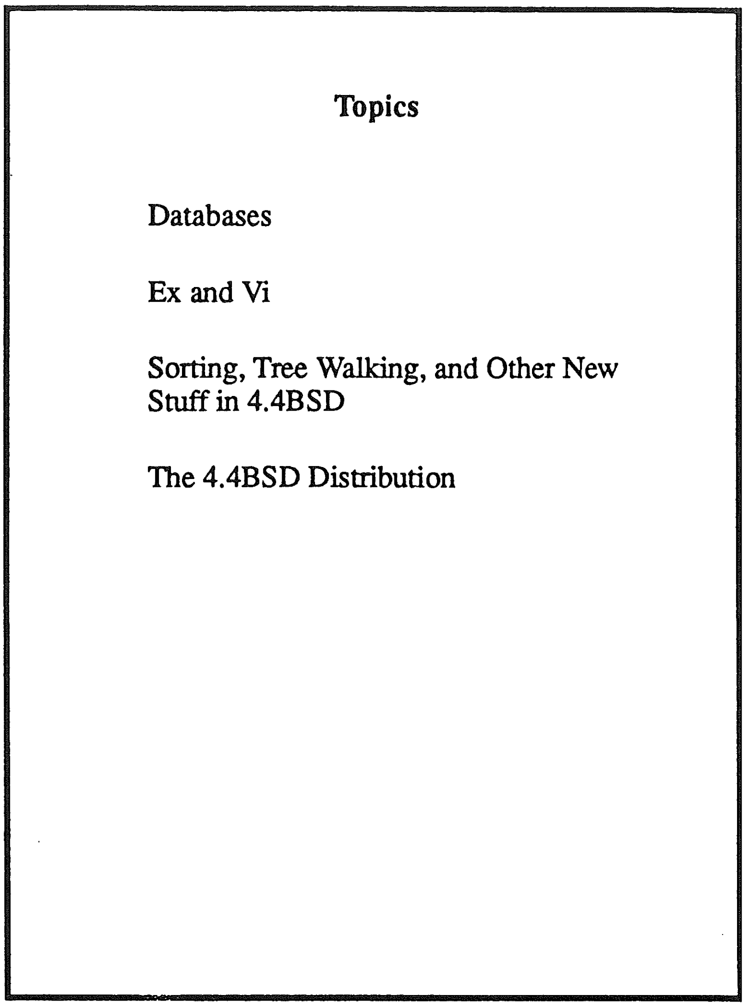


What's New in the 4.4BSD User Code [†]

Brought to you by

M. Kirk McKusick

Australian UNIX Users Group
Summer 1994



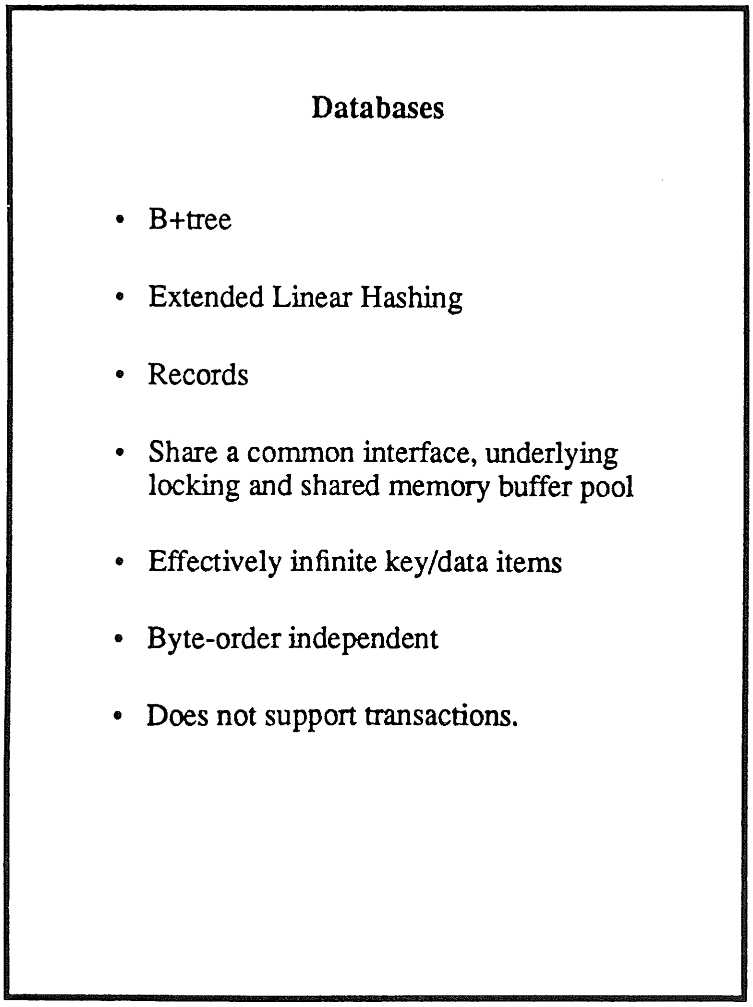
Topics

Databases

Ex and Vi

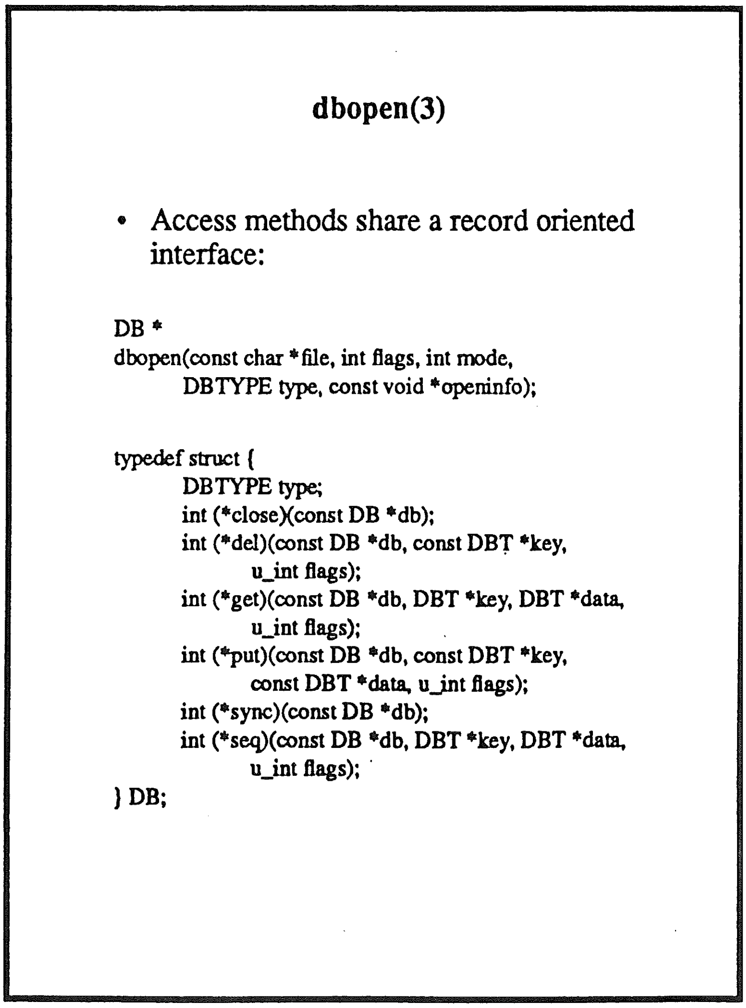
Sorting, Tree Walking, and Other New Stuff in 4.4BSD

The 4.4BSD Distribution



Databases

- B+tree
- Extended Linear Hashing
- Records
- Share a common interface, underlying locking and shared memory buffer pool
- Effectively infinite key/data items
- Byte-order independent
- Does not support transactions.



dbopen(3)

- Access methods share a record oriented interface:

```
DB *
dbopen(const char *file, int flags, int mode,
        DBTYPE type, const void *openinfo);


typedef struct {
        DBTYPE type;
        int (*close)(const DB *db);
        int (*del)(const DB *db, const DBT *key,
                u_int flags);
        int (*get)(const DB *db, DBT *key, DBT *data,
                u_int flags);
        int (*put)(const DB *db, const DBT *key,
                const DBT *data, u_int flags);
        int (*sync)(const DB *db);
        int (*seq)(const DB *db, DBT *key, DBT *data,
                u_int flags);
} DB;
```

[†] Reprinted with Kirk's permission.

Key/Data Pairs

- Access methods share an underlying data structure for keys and data:

```
typedef struct {
        void *data;
        size_t size;
} DBT;
```

B+tree

Comer, Douglas, "The Ubiquitous B-Tree," Computing Surveys, Volume 11, No. 4., 1979.

- Sorted, balanced tree structure, storing associated key/data pairs
- User specified sort function
- User specified prefix compression
- Duplicate keys in no specific order
- Intelligent splitting, intended for ordered insertion

Extended Linear Hashing

Larson, Per-Ake, "Dynamic Hash Tables," Communications of the ACM, Volume 31, No. 4., April 1988.

- Linear hashing, storing associated key/data pairs
- Splits occur in a predefined order
- Overflow pages preallocated between primary pages
- *Ndbm*(3) interface.
    - Used by pretty much everything that does hashing in 4.4BSD

Records

- Fixed or variable length records
- UNIX file support
- Key is a record number
- Used by *nex*/*nvi*(1) and *ed*(1) in 4.4BSD

Transactions

Seltzer, Margo, "LIBTP: Portable, Modular Transactions for UNIX," Proceedings 1992 Winter USENIX Conference, January 1992.

- Level of kernel support:
  - For free in LFS
- UNIX semantics:
  - IBM Almaden, Quicksilver
  - Transaction protected file descriptors passed via IPC.

Ex/Vi

- Fairly stable, beta test quality by 4.4BSD, included as nex/nvi.
- Almost entirely compatible with historical vi.
- Mostly compatible with POSIX 1003.2.
- 25,000 lines of C source, 270K of text

Features

- Infinite length lines, files
- 8-bit clean, ready for internationalization using fixed width characters
- Maintainable
- Two global variables (should be usable as a library)
- Infinite undo.
- Designed for window support.
  - Uses *curses*(3). (Curses has been enhanced to be 8-bit clean and support hardware scrolling.)
  - Tiny screens
  - Split screens.

Futures

- Interpretative language (Tcl/Tk, Lisp)
- Two people are working on X support (Tcl/Tk, Motif)
- Emacs mode
- Tuning -- performance is an issue
- Standalone capability, disk-less network booting support.

Sorting (1)

- New implementations of qsort, heapsort, mergesort, and radixsort. All use the traditional qsort interface except for radixsort. Normally radixsort > qsort > mergesort > heapsort.

- Qsort is faster than the historic qsort, although not that much faster than recent Berkeley implementations. The major win is extending median selection from three to nine.

- Qsort requires no additional memory (ANSI does not permit it to return an error), but is recursive.

  Bentley, J.L., "Engineering a Sort Function", bentley@research.att.com, January 1992.

Sorting (2)

- Mergesort is a stable, modified merge sort with exponential search intended for sorting data with pre-existing order. It requires N * object_size bytes of additional memory.

  Mcilroy, P.M., "Optimistic Sorting and Information Theoretic Complexity", Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, January 1992.

- Radixsort is a variant of most-significant-byte radix sorting, taking linear time relative to the number of bytes in the string. In two versions; the stable version requires N * pointer_size bytes of additional memory.

Sorting (3)

- Heapsort is a traditional Knuth implementation, using almost no additional memory.

- Intention is to eliminate sorting from all introductory programming classes.

FTS

- Tree-walking subroutine
  - Dave Korn, Glenn Fowler, Phong Vo
  - Fopen(3) of a file hierarchy

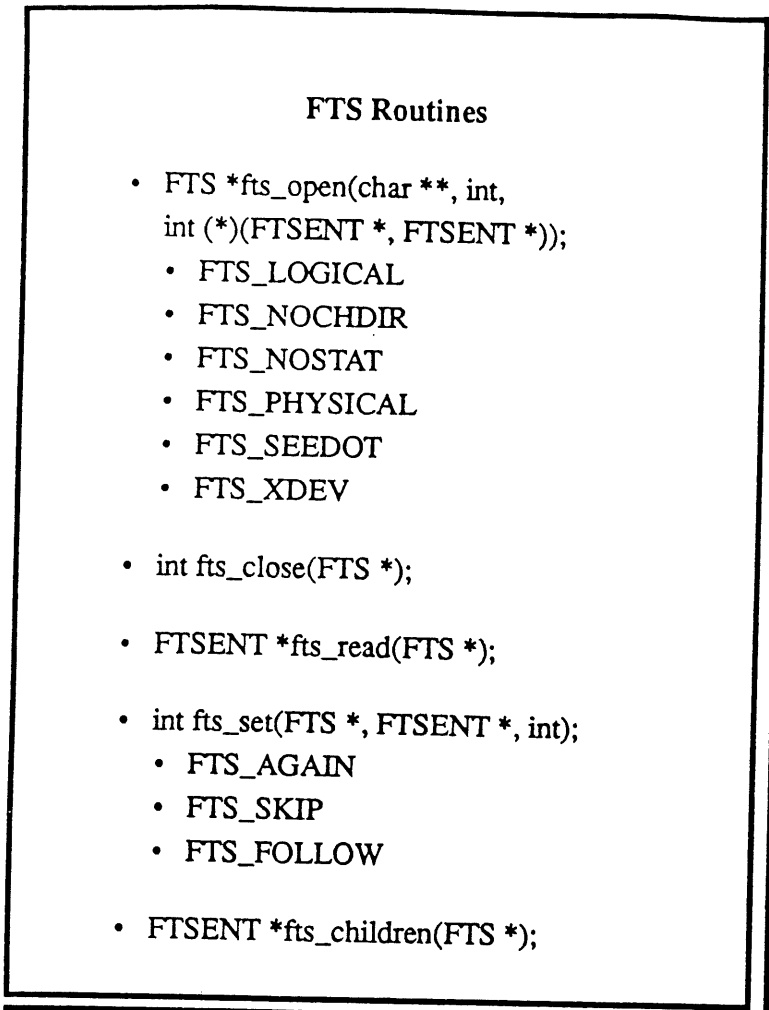
- Replaces ftw(1) and nftw(1)

- 4-1 speedup in system time

- 1.5-1 speedup in user time

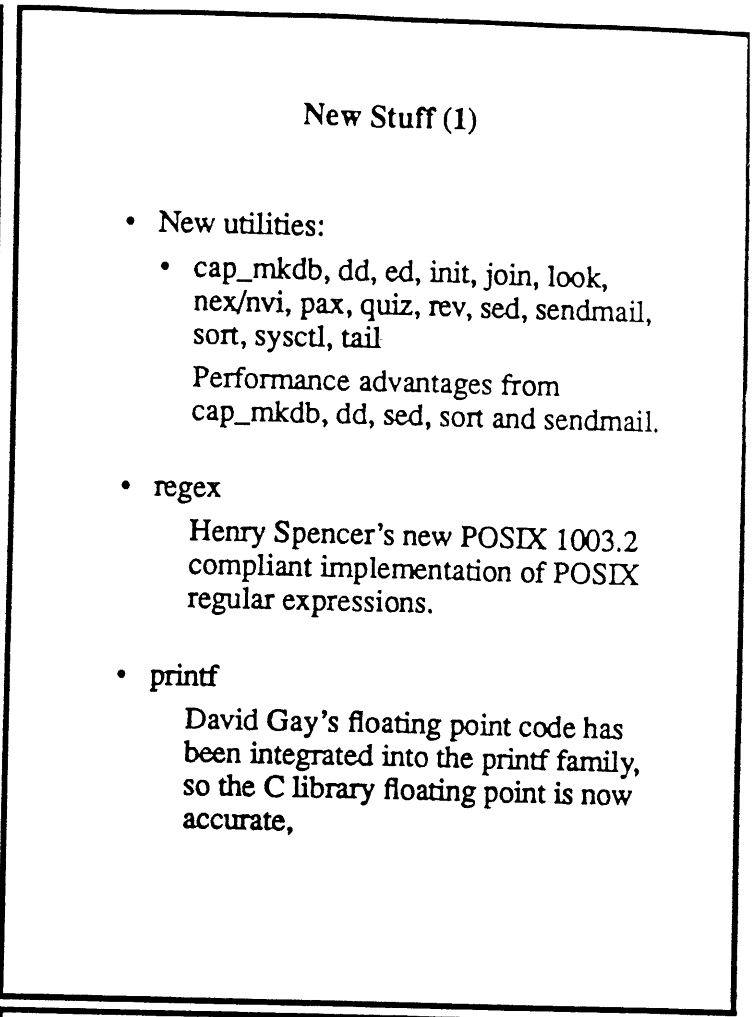
- Infinite depth (POSIX 1003.2)
  - 1000 lines of C

- chgrp(1), chmod(1), chown(1), cp(1), diff(1), du(1), find(1), ls(1), mtree(1), pax(1), rm(1)



FTS Routines

- FTS *fts_open(char **, int, int (*)(FTSENT *, FTSENT *));
  - FTS_LOGICAL
  - FTS_NOCHDIR
  - FTS_NOSTAT
  - FTS_PHYSICAL
  - FTS_SEEDOT
  - FTS_XDEV
- int fts_close(FTS *);
- FTSENT *fts_read(FTS *);
- int fts_set(FTS *, FTSENT *, int);
  - FTS_AGAIN
  - FTS_SKIP
  - FTS_FOLLOW
- FTSENT *fts_children(FTS *);



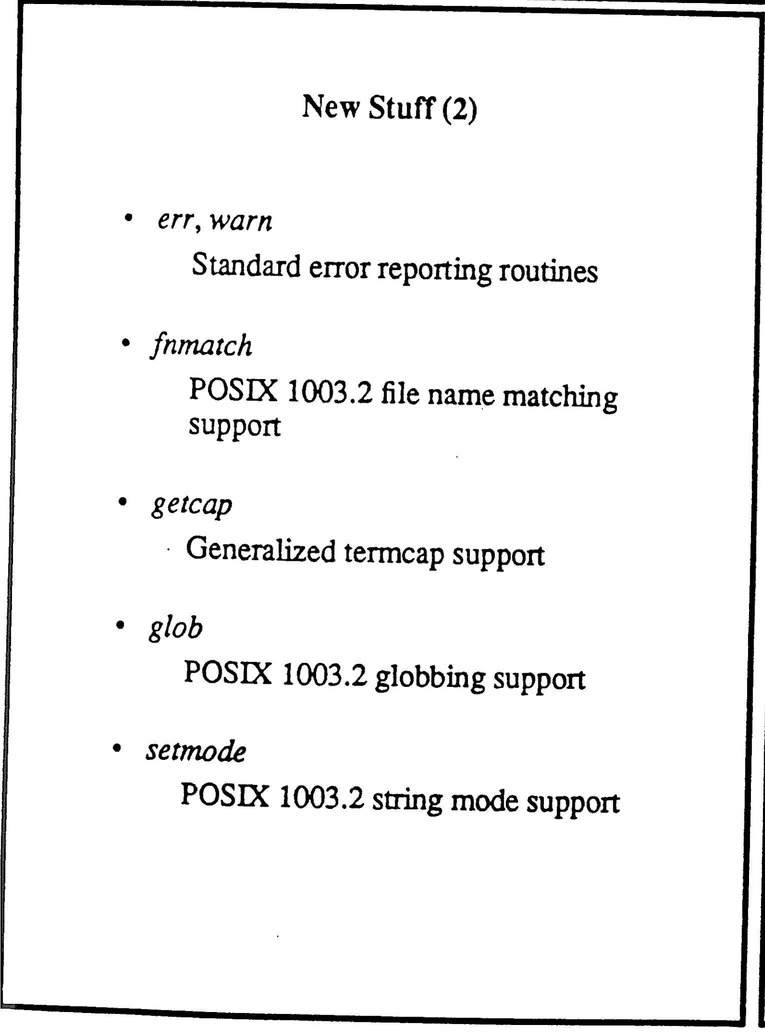
New Stuff (1)

- New utilities:
  - cap_mkdb, dd, ed, init, join, look, nex/nvi, pax, quiz, rev, sed, sendmail, sort, sysctl, tail

    Performance advantages from cap_mkdb, dd, sed, sort and sendmail.
- regex

  Henry Spencer's new POSIX 1003.2 compliant implementation of POSIX regular expressions.
- printf

  David Gay's floating point code has been integrated into the printf family, so the C library floating point is now accurate,



New Stuff (2)

- *err, warn*

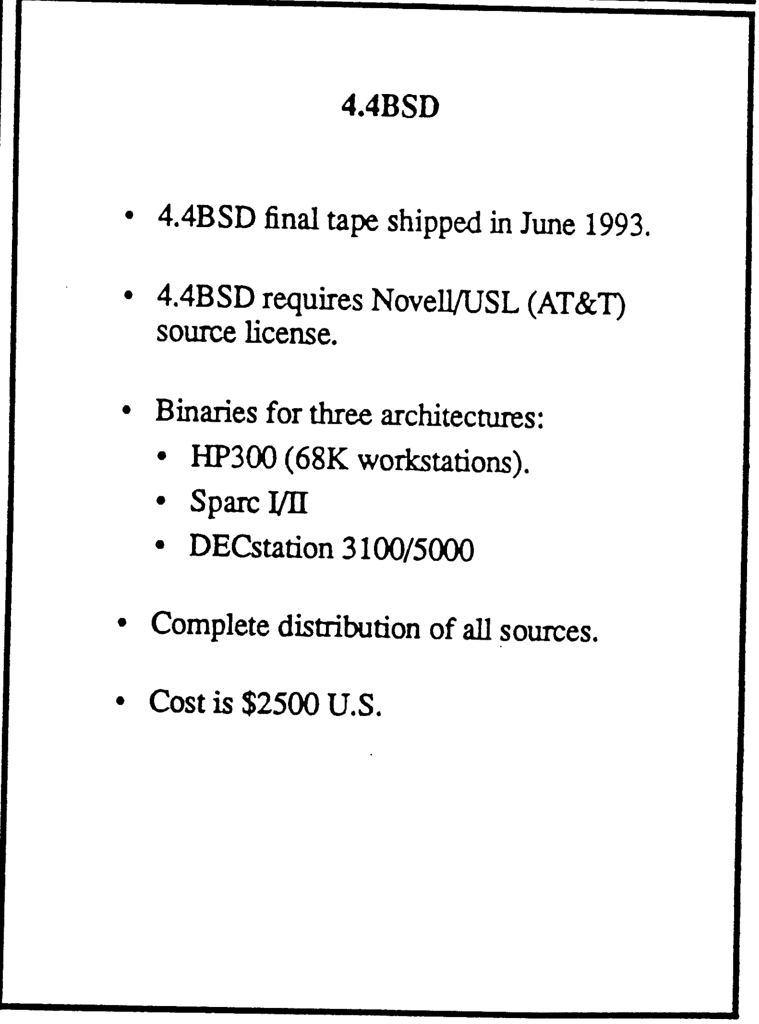
  Standard error reporting routines
- *fnmatch*

  POSIX 1003.2 file name matching support
- *getcap*

  Generalized termcap support
- *glob*

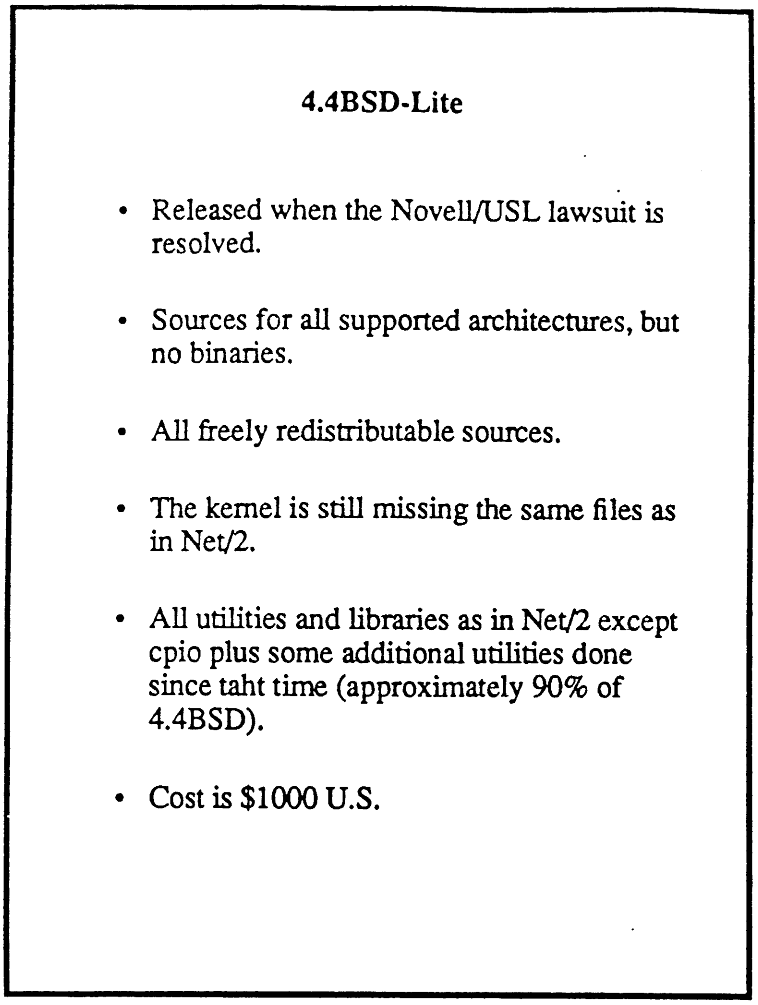
  POSIX 1003.2 globbing support
- *setmode*

  POSIX 1003.2 string mode support



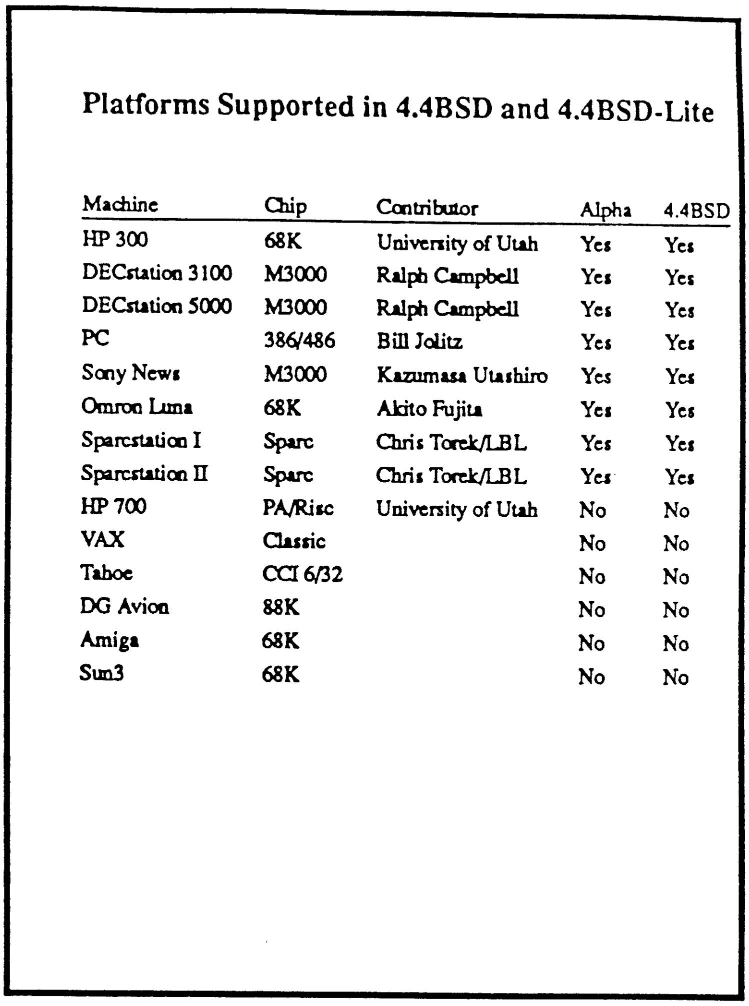
4.4BSD

- 4.4BSD final tape shipped in June 1993.
- 4.4BSD requires Novell/USL (AT&T) source license.
- Binaries for three architectures:
  - HP300 (68K workstations).
  - Sparc I/II
  - DECstation 3100/5000
- Complete distribution of all sources.
- Cost is $2500 U.S.



4.4BSD-Lite

- Released when the Novell/USL lawsuit is resolved.
- Sources for all supported architectures, but no binaries.
- All freely redistributable sources.
- The kernel is still missing the same files as in Net/2.
- All utilities and libraries as in Net/2 except cpio plus some additional utilities done since taht time (approximately 90% of 4.4BSD).
- Cost is $1000 U.S.



Platforms Supported in 4.4BSD and 4.4BSD-Lite

Machine	Chip	Contributor	Alpha	4.4BSD
HP 300	68K	University of Utah	Yes	Yes
DECstation 3100	M3000	Ralph Campbell	Yes	Yes
DECstation 5000	M3000	Ralph Campbell	Yes	Yes
PC	386/486	Bill Jolitz	Yes	Yes
Sony News	M3000	Kazumasa Utashiro	Yes	Yes
Omron Luna	68K	Akito Fujita	Yes	Yes
Sparcstation I	Sparc	Chris Torek/LBL	Yes	Yes
Sparcstation II	Sparc	Chris Torek/LBL	Yes	Yes
HP 700	PA/Risc	University of Utah	No	No
VAX	Classic		No	No
Tahoe	CCI 6/32		No	No
DG Avion	88K		No	No
Amiga	68K		No	No
Sun3	68K		No	No

UniForum NZ'94

The Open Advantage

11th Annual Conference

18-21 May 1994

Quality Hotel, Rotorua

Combine a little work with pleasure! Attend UniForum NZ'94 and see the sights of one of New Zealand's top tourist destinations at the same time.

The exchange rate works in your favour, and AUUG members may attend the conference at the NZ member rate.

Conference Registration	$NZ	$AU Approximate *
before 20 April	$395	$325
after 20 April	$495	$410

(includes attendance at all conference sessions 19-21 May, exhibition, and all meals except breakfast)

Tutorials (18 May)	$150	$125 each
Hotel Accommodation	$100	$85 per night

* Conversion to AU$ will depend on exchange rate fluctuations.

Registration forms will be available mid-March 1994. Contact Julie Jones, UniForum NZ, P.O. Box 27-149, Auckland, New Zealand, Ph 64-25-958-245 or fax 64-9-629-2015.

Testing Tools under UNIX

Peter Chubb
Softway Pty Ltd

AUUG NSW Summer '94

Abstract

Testing is divided into two parts: the hard part and the tedious part. The hard part is determining what to test; the tedious part is doing it.

There are at least three packages I know about for helping with the tedious bit (one developed at Softway, one developed by the FSF, called DejaGnu, and some expensive ones). This talk will describe a tool developed at Softway to automate running tedious tests and compare it with DejaGnu.

1 Introduction

Testing is important. Without it, one doesn't *know* that one's program is actually performing according to specification. However, cost-effective testing is hard to do.

Good testing uncovers problems in the implementation of a software system, and detects when problems recur after modifications. Here, a *problem* is defined as *any behaviour that does not match the published interface and functional specification.* (At Softway, this is usually the Software Requirements Specification (SRS, as per IEEE-830-1984) and UNIX manual pages).

Exhaustive testing is impractical for almost any software system of reasonable size. Instead, we try to test the aspects that are most often used thoroughly, and other aspects at a lesser level.

We typically derive a list of test *assertions* (following IEEE Std 1003.3-1991) from the requirements document and the user manual(s). From this we generate a set of *test cases*, one or more per assertion. Each test case tests an assertion in a specific instance. Test cases are usually chosen to exercise the system under test at likely boundary conditions, to try to make the assertion fail, and to perform the most common operations. (The aim being to ensure that there are no bugs that will be hit in everyday use).

In addition to this functional testing, we do some unit testing. For this, the interface specifications from the detailed design documents are used as inputs to the process; otherwise testing proceeds as before.

1.1 Test Specification Example

The rest of the paper uses as example the UNIX command *cat*. The example requirements are derived from the manual page, *cat(1)*.

Requirement 1 *cat reads each file on its command line in sequence and writes it on the standard output*

Requirement 2 *cat reads from standard input if no filenames are given on its command line*

As an example, consider the requirement 1. It does not specify the contents of the files, or how many there are. When taken with requirement 2, an assertion can be derived for one or more files specified on the command line.

Test Assertion 1 *Where names of existing readable files are specified on cat's command line, the output from cat is the contents of the files named on the command line, in the order specified, without regard to what those contents might be*

This assertion is derived from Requirements 1 and 2.

Test cases then test the assertion in several ways: with a single file which contains only characters chosen from the printable ASCII character set; with many files; with 8-bit data; etc. Only some of the many derivable test cases are listed here.

Test Case 1 *cat succeeds for a single ascii-contents file specified on its command line*

Execute: `cat /etc/passwd > foo`
To pass, the contents of foo must be the same as those of `/etc/passwd`.

Test Case 2 *cat succeeds for as many ascii-contents files as can be specified on the command line*

Test Case 3 *cat succeeds for a single file containing randomly chosen binary data, specified on its command line*

With each test case, there is a test procedure. (Only one of these is shown in the example). These procedures will typically be executed many many times: at least once before any release to our customers, and usually after any major change to the implementation (including bug fixes, for regression testing).

For low budget, short lived, small projects, it may not be worth automating the tests. In automating a test suite, one typically runs all the tests manually several times anyway, in testing the test suite. Hence it may be worth the effort of running the tests manually — especially as the automated test suite itself needs to be validated. In most of our projects, we have a commitment to maintain the software system and so an automated test suite is worthwhile.

2 Requirements for Testing

To be valuable, tests must be well defined, traceable, unambiguous and repeatable.

A *well defined* test tests only one thing. This implies a large number of different test cases to cover the gamut of the tested system's behaviour.

A *traceable* test is one where what is tested can be traced right back to the source requirements documents or interface documents. This implies that tests are uniquely labelled, are well documented, and have been checked for conformance.

An *unambiguous* test is one whose result is obvious: either it has passed, or it has failed. This implies that there should be a well-defined criterion for passing the test. Tests should avoid false positives and false negatives; but should err on the side of reporting a failure when there isn't one rather than the other way about: test failures prompt remedial action; test passes just give a warm fuzzy feeling to the implementors.

A *repeatable* test is one that gives the same result (pass, fail or unresolved) each time it is performed on the same hardware and software. This implies that the version of the tested artifacts should be identified in every test, which implies in its turn a good change management system underlying the entire project.

For ease of test development and test running, it is desirable for more than one person to be able to run tests at a time. It is also desirable to be able to run just some of the tests, rather than the entire set.

Manual testing, from a written test procedure, provides some of the functionality required. However, because people are fallible, and testing is tedious, repeatability suffers. This option is also expensive: it is better to have one's highly skilled, highly-paid staff doing something more productive than running and evaluating tests. This makes automated testing highly desirable.

3 Anatomy of a Test Procedure

Typically, a test case consists of some setup (where needed files are created, and an environment set up), an invocation of the system under test (this may mean sending input to a pre-existing process, or may mean invoking a process with particular arguments), an evaluation of the output of the command under test, and a clean up. A good test procedure will not leave any extra processes, files or other system resources around after it is finished.

Some parts of the setup and cleanup are common to all test cases for a particular system. Other parts are common to all test cases derived from a particular assertion. Also, some test cases lend themselves to subdivision into smaller test cases; setup and cleanup is usually common to these too.

One doesn't usually wish to run tests under the uid and in the environment of the tester. The person doing the tests will usually have different system privileges, and an uncontrolled environment (thus reducing the reproducibility of the tests). Also, because we wish to be able to perform more than one test run at once, we cannot usually use a fixed test uid. Part of the setup, then, is usually to create a test user to run the tests as. All the tests are then run as this user; after the tests are finished, the user is removed from the system. Between each script, the user's directories are removed and recreated; and all processes belonging to that user are killed.

However, some bits of setup and some tests require one to be root (for example, testing various ioctls for a disc driver, or testing a quota system). Therefore, one needs a way to escape from the test user for the duration of a single command — and in a way that does not (as far as possible) compromise security for the rest of the machine. When testing manually, this is typically achieved by the use of a root shell in another window — not a desirable situation for an automated test.

4 Automated Testing

By 'automated testing' is meant the automated running of a test suite, and support for generation of test cases that can be run with no human interaction.

Automated tests can be run lots of times, and will do the same thing each time. However, they are tedious to write and to check, because they tend to consist of much similar code.

It is useful to be able to abstract the common functionality (global setup, such as creating the test userid, result reporting, selecting which tests to run, etc) into separate programs. This is what both DejaGnu and *dotests* do.

5 dotests

We developed a system called `dotests`. `dotests` is a shell script together with supporting programs that:

- Scans a directory of automated test scripts, and executes all the ones that meet a selection criterion. The selection criteria is based on the names of the directories holding the tests. Patterns are specified on the command line, and these passed to a *find* process; any directories found that contain files with names of the form `Test.`*XXX* are treated as test cases.
- Before the test run is started, uses the change management system to work out what version of the software is being tested,
- Before each test is executed, creates a test user with an empty home directory, mail spool file, etc.
- Executes each test as the test user, in an empty directory (not the test user's home directory) with a tightly-defined shell environment.
- Cleans up afterwards (killing all processes owned by the test user, and deleting all files in the test user's home directory and in the test directory, etc)
- Reports a summary of tests passed, etc., to standard output.
- Logs a full transcript of the entire test run to a file.

A `dotests` environment consists of a base directory (held in an environment variable *$TOPTESTS*), containing below it subdirectories. Leaf directories contain files of the form *Test.XXX*, where *XXX* can be any string of legal filename characters. By convention, all baseline tests are held in *$TOPTESTS/TC*; in addition, tests derived from bug reports are in *$TOPTESTS/BUG*. However, this is not enforced by `dotests`.

Tests are named by the sequence of directories below *$TOPTESTS*: for example, the test in file `/usr/src/test/TC/cmd/cat/1/Test.1`, assuming that *$TOPTEST* is `/usr/src/test`, is called `TC.cmd.cat.1`, item 1.

Within a leaf test directory, the following files may be present:

Title contains a single line containing a short form description of the test. This is used in generating test reports.

Mode If this file is present and contains the word 'interactive', the test is an interactive test. Most tests do not require human interaction; some (particularly ones that involve console messages, system shutdown, or that can have deleterious effects on other users of the machine) do. These can be picked out be `dotests` and run separately.

Setup If this file is present, it is executed before any other file in the directory.

Cleanup If this file is present, it is executed after all the tests have been complete. `Cleanup` will always be executed if `Setup` was, even if `Setup` returned a non-zero exit status.

Pre If this file is present, it is executed once before each test `Test.XXX`. If it returns a non-zero exit status, the test is aborted, and the `Post` file is executed.

Post If this file is present, it is executed after each `Test.XXX`, regardless of the test's outcome.

Env If this file is present, it is used to set environment variables and shell functions common to all the tests and to the setup and cleanup, pre and post actions.

Test.XXX Each file named in the form `Test.XXX` contains a single test case. This is the only mandatory file in a test case directory. It is a shell script. If it returns a non-zero exit status, the test is aborted, and `Post`, and `Cleanup` are executed, in that order. If it returns a zero exit status, then the result is the most pessimistic of the results reported using the `report` shell function, or *unresolved* if `report` has not been called.

There are other files that may be present, and that are significant to `dotests`, but they are not so often used.

5.1 Reporting Results

The possible outcomes for a test under `dotests` are:

success The test case succeeded.

fail The test case failed.

unresolved The test case couldn't work out whether the test succeeded or not.

aborted Something went wrong with the test setup or execution.

Results are held in AT&T Mail-box format. This means that ordinary mail readers can be used to process the outputs, and to go quickly to any single test. The messages in the mailbox are:

- Version information dump.
- First test case
- Next test case

  ⋮

- Last test case
- Summary.

The *version information* is the RCS or SCCS id of every file in the system under test, the kernel version and build date, and the version stamp of `dotests` itself.

The *Test Case* reports have as subject a summary of the status of that test case — for example, the headers might contain:

```
From dotests Fri Feb 11 14:20:28 EDT 1994
Subject: TC.cmd.cat.2 FAILED - multiple ASCII files concatenated
To: peterc
From: dotests (Test run Bq)
Date: Fri Feb 11 14:23:03 EDT 1994
Lines: 98
```

The `To:` header contains the real user ID of the person running the tests; the `From:` line contains the test run.

Following the headers is a summary of this test case:

```
Item 1 -- passed
Item 2 -- passed
Item 3 -- failed
Item 4 -- unresolved
Item 5 -- passed
```

Following the summary is a transcript of the entire execution of the test scripts, produced using `sh -x`.

The final mail item in the mailbox is a summary, for example:

```
From dotests Fri Feb 11 14:20:30 EDT 1994
Subject: Summary: 21 passed, 7 failed, 2 unresolved, 0 aborted
To: pod
From: dotests (Test run C4a)
Date: Thu Feb 11 14:29:01 EDT 1994
Lines: 5
```

5.2 Facilities Provided

A library of shell functions and command is provided to the `dotests` command and to the scripts running under `dotests`. The most useful of these are:

report This function takes a single argument: `pass`, `fail` or `unresolved`. Remaining arguments are ignored, and may be used as comments. This function must be used in each test script.

assert It is often easier to form tests on the basis of assertions. The command

```
assert [-fru] "comment" command [arg ...]
```

evaluates `command [arg...]` with stdout directed to `/dev/null`, and (in the usual case) terminates the script abnormally if the command returns a non-zero exit code.

If `-f` or `-u` are specified, the test result is failure or unresolved, and the script continues rather than aborting.

`-r` reverses the sense of the test.

asuser This command is invoked as

```
asuser <username> <command> [arg...]
```

It invokes the command as the specified user (usually root). It does various permission checks to ensure that ordinary users cannot invoke it.

lock is to lock a file, so that more than one user can run tests that use system-wide files at a time. `lock` takes a timeout and the name of a file as arguments; it returns a lock ID to be passed to `unlock`.

If the file to be locked is specified by a non-absolute path, it is assumed to be a script in a special directory and is run. In this way, files can be locked (and created) even if they do not exist. For example, a file in the root directory could be created and locked as follows:

```
lock rootjunkfile
```

where `rootjunkfile` is a script containing something like this:

```
#!/bin/sh
[ -f /rootjunkfile ] && mv /rootjunkfile /rootjunkfile.$LOCKID
asuser root sh -c "mkjunk /rootjunkfile; chmod a=r /rootjunkfile"
```

Merely locking the file causes it to spring into existence. At present, it is up to each individual test to clean up any file created or altered in this way.

5.3 Our Use of the `dotests` Environment

An earlier version of `dotests` was used on the SHARE II project; the current version is an elaboration of that early version for a specific project. In the SHARE II, project, we had only a few tests (65) because only a small part of the project was tested.

The current version of `dotests` was developed under the assumption that it was to be used only for a particular project The version dump, for example, is only for that project, and knows which files to dump. Some of the other functions (not listed above) are also fairly specific. However, it would be reasonably easy to abstract these kinds of preparation into separate, global, setup, extract version, and cleanup scripts.

For the current project, we have over 480 test cases at present. Many of these have several subtests, so the actual number of tests is higher than this.

In general, a test can be coded in only a few lines. This makes tests easy to check. For the 250 baseline tests (excluding bug fixes), the average length, including comments and RCS log, is 31.4 lines. (The standard deviation is high, at 37.5; the shortest test is 11 lines ,the longest test is 142 lines: this test actually puts twelve test cases in a single file).

Because the setup for a test is split from the actual test, extra testing of a test case is easy to add without overmuch code duplication. Splitting off the cleanup means that it still occurs even if the test itself aborts abnormally.

The environment provided by dotests (current directory, home directory, user and group id all distinct from anyone else on the system) is very good, both for limiting the damage caused by testing buggy systems, and for allowing more than one tester at a time to develop and run tests.

We've used dotests for two projects, spanning 4 major releases over 3 years now. It has proved reliable, and moderately easy to use.

5.4 Example Test Script

```
# Title file
multiple ascii files

#Setup
for i in 'count 1 500'
do
# mkjunk creates a file full of rubbish, of random length
  assert "files created OK" mkjunk -a $i
done

#Test.1
>out
for i in 'count 1 500'
do
# we've already tested in the previous test case that a
# single file cat is OK
  assert "cat succeeded" eval "cat $i >> out"
  assert -f "cat succeeded" eval "cat >test.out \
'count 1 $i'"
  assert -f "contents identical" cmp -s test.out out
done
report pass
```

In this example, which would live in `$TOPTEST/TC/cmd/cat/2`, 500 files are created, filled with random ASCII data. (`count` is a shell script that outputs all the integers between its arguments).

In the test itself, `cat` is invoked to concatenate various numbers of these files together, and the result compared with the expected result.

Note that the second assertion has an `-f` argument: this says that instead of aborting the test script, it should report failure.

The same test script could be used for binary data, by removing the '`-a`' option to `mkjunk`.

6 DejaGnu

Cygnus have provided a testing framework called **DejaGnu**. This framework is freely distributable under the Gnu Public License. It uses **expect**[1] rather than the shell as a test command language; this has advantages for testing interactive commands, but is an additional language to learn, that many people are not familiar with.

In addition to PASS, FAIL and UNRESOLVED exit stati, **DejaGnu** provides XPASS, XFAIL, UNTESTED and UNSUPPORTED. The X variants are to indicate expected failures (or passes when a failure was expected). A failure is expected when a known bug is present, for example.

DejaGnu also provides for remote testing, on a standalone single board computer, or on a machine connected via a network.

DejaGnu is less tied to a particular software system than `dotests`; this is at the price of increased complexity. There is some work needed to customise **DejaGnu** for a particular system under test

[1] **expect** is a language designed by Don Libes, built on Ousterhout's **tcl**

(called a *tool* in the **DejaGnu** documentation). The test writer has to provide a module (the 'init file') that defines up to four procedures to set up the test system. These procedures are:

*tool*_**start** that starts the program under test and leaves it running for the test cases. For interactive programs, *tool*_start is called once from the test initialisation; for batch oriented programs, each test script calls *tool*_start.

*tool*_**exit** is to clean up (if necessary) before the test run completes. This must be used to remove any temporary files.

*tool*_**load** that loads something into a tool. For example, `gdb_load` loads a new executable file into the debugger (when `gdb` is the process under test).

*tool*_**version** is invoked to work out what version of the tool is being tested.

Together, these could almost be used to provide the same kind(s) of environment that `dotests` provides; but some way of getting access as root (to create the new accounts) would be needed.

DejaGnu uses almost the same methods to find tests as are used by `dotests`. A *find* process is run to look for tests that match the directories on the command line; tests are named by strings of the form `testname`*.exp*.

Facilities provided by **DejaGnu** are:

- Provision of an environment set up from a set of configuration files for the tests to run in.
- Locating individual test scripts, using a naming convention based on the `--tool` argument.
- Providing special functions (such as the reporting functions described above) that extend Tcl.
- Locating target-dependent functions, to standardise the test environment across a number of test platforms.

As such, the functionality of the **DejaGnu** tools is slightly less than that of *dotests*. The real power of **DejaGnu** comes from the use of *expect*, particularly for testing interactive programs.

7 Conclusion

For our purposes, `dotests` is adequate, and **DejaGnu** as it stands is not. **DejaGnu** does not create new test users, nor does it provide a controlled, unique, environment in which to run tests. This means that only one test run is possible at a time.

DejaGnu is, however, otherwise much more general purpose, and provides the kinds of extensibility that we would like to add to `dotests` for use on other projects. The use of *expect* is particularly interesting from this point of view.

References

[1] Cem Kaner, *Testing Computer Software*, TAB books, 1988.

[2] Don Libes, *expect: Curing Those Uncontrollable Fits of Interactivity*, Summer 1990 USENIX Conference, Anaheim California, June 1990.

[3] Don Libes, *Regression Testing and Conformance Testing Interactive Programs*, Summer 1992 USENIX Conference, San Antonio, Texas, June 1992.

[4] John Ousterhout, *TCL: An Embeddable Command Language*, Winter 1990 USENIX Conference, Washington, D.C., January 1990.

[5] Rob Savoy, *The DejaGnu Testing Framework*, Free Software Foundation, 1993.

[6] IEEE Standard 830–1984, *IEEE Guide to Software Requirements Specifications*

[7] IEEE Standard 1003.3–1991 *Test Methods for Measuring Conformance to POSIX*

A Twisty Little Maze of Machine Descriptions
or
An Overview of GCC Porting.

Frank Crawford

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

ABSTRACT

The Gnu C Compiler (*gcc*) from the Free Software Foundation (*FSF*) has taken over from the Portable C Compiler as the common C Compiler available across many different platforms. In many ways it is also following in the footsteps of UNIX, by being developed and extended by the user community and by being adopted for use on new systems. However, unlike UNIX, there are restrictions on the redistribution of *gcc* for commercial purposes, so its use is limited to either being a secondary compiler or on free (or at least very cheap) systems.

Despite the wide spread use of *gcc*, very few people understand how it achieves its portability. This paper will outline what is required to port *gcc* to a new architecture, using, as an example, a recent port carried out by the author to the IBM System/370 architecture.

Finally, the new paradigm for project development using the Internet, exemplified by *gcc*, will be discussed. Unlike traditional methods where a small team works exclusively on a project, *gcc* has thousands of part-time developers, including experts in the field, with the ability to communicate almost instantaneously. This gives them the ability to report and act on problems and distribute fixes in less time than traditional project groups would take to even be informed of a problem.

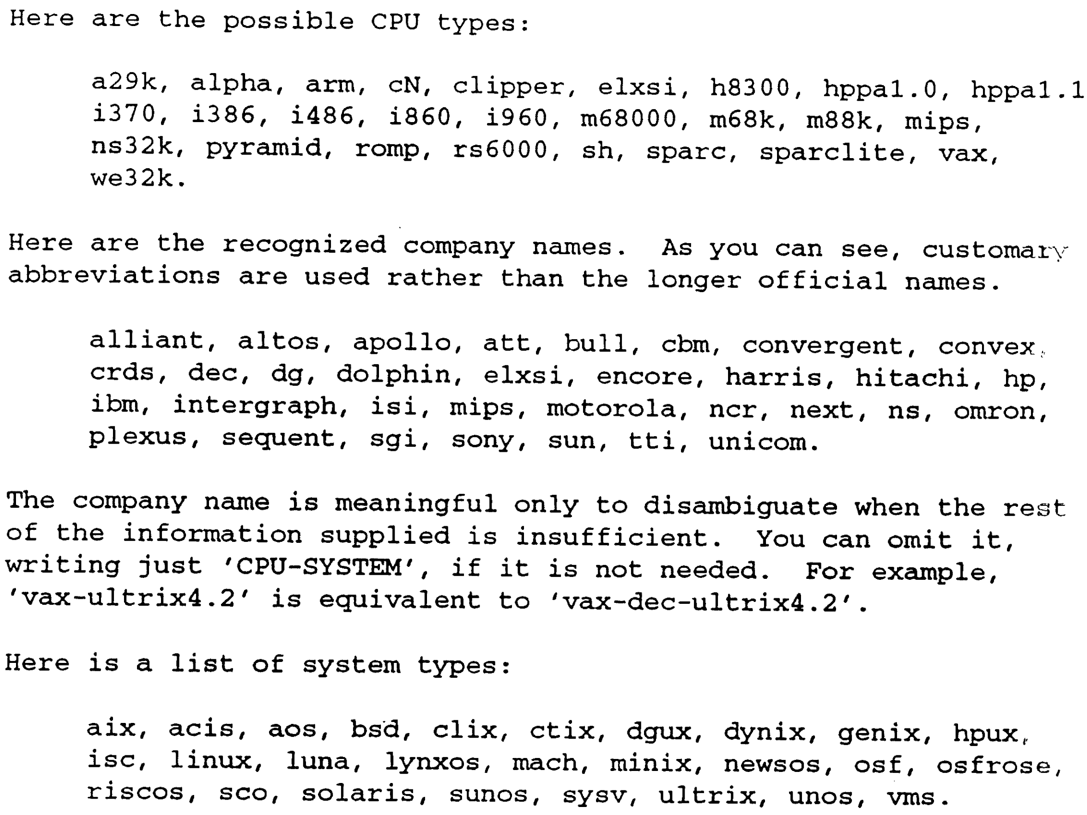
1. Gnu C Compiler Overview

Anyone who has dealt widely with *public domain* or other freely available UNIX software will have come across reference to Gnu products, from the Free Software Foundation (*FSF*), and in particular the Gnu C Compiler (*gcc*). Originally, the Gnu software was an attempt to implement a *free* version of UNIX, which, although not completed, has been the basis of much of the software available on the latest publicly available UNIX systems, such as Linux, BSD/386 and 386BSD.

Aside from the new UNIX systems, Gnu products have been ported to many other commercial platforms, covering many different chip sets, architectures and versions of UNIX. One of the reasons for this wide availability is that the compiler itself has been widely ported. *Gcc* has been ported not only to all the standard versions of UNIX (such as Version 7, BSD 4.2 and BSD 4.3, OSF/1 and the various System V releases) and to all the standard chip sets (such as Sparc, HPPA, RS6000, MIPS and Intel 80386), but also to VMS, MVS and MS-DOS. An extract from the gcc-2.5.8 install documentation is given in Figure 1. The naming convention is by architecture, vendor and system. Obviously not all combinations are supported, but the number that are, is enormous.

There is another important feature of *gcc* that contributes to it being widely ported. Aside from being an ANSI-C compiler, the *gcc* distribution also includes both C++ and Objective-C *front-ends*. In fact, ANSI-C is implemented as a *front-end*, with most of the compiler implemented as far as possible as a *language-independent back-end*. This has made *gcc* the choice of many groups developing compilers for other languages. Currently, there are groups developing front-ends for Ada, Chill, FORTRAN 77 and Pascal, and possibly others.

Finally, *gcc* has been developed with the ability to act as a *cross-compiler*, *i.e.* generate code for a different system than it is currently running on. To generate runnable programs, you also need such utilities as a cross assembler, cross linker, cross-compiled library, and other related development tools,



Here are the possible CPU types:

a29k, alpha, arm, cN, clipper, elxsi, h8300, hppa1.0, hppa1.1, i370, i386, i486, i860, i960, m68000, m68k, m88k, mips, ns32k, pyramid, romp, rs6000, sh, sparc, sparclite, vax, we32k.

Here are the recognized company names. As you can see, customary abbreviations are used rather than the longer official names.

alliant, altos, apollo, att, bull, cbm, convergent, convex, crds, dec, dg, dolphin, elxsi, encore, harris, hitachi, hp, ibm, intergraph, isi, mips, motorola, ncr, next, ns, omron, plexus, sequent, sgi, sony, sun, tti, unicom.

The company name is meaningful only to disambiguate when the rest of the information supplied is insufficient. You can omit it, writing just 'CPU-SYSTEM', if it is not needed. For example, 'vax-ultrix4.2' is equivalent to 'vax-dec-ultrix4.2'.

Here is a list of system types:

aix, acis, aos, bsd, clix, ctix, dgux, dynix, genix, hpux, isc, linux, luna, lynxos, mach, minix, newsos, osf, osfrose, riscos, sco, solaris, sunos, sysv, ultrix, unos, vms.

Figure 1. Extract from gcc-2.5.8 Installation documentation.

but many of these are also available within the Gnu distribution.

Today, *gcc* is being extended in three different ways: ported to new systems, having new language front-ends developed and enhanced to produce better and more optimised code. The thing that makes *gcc* is that all these extensions are being done by the user community, not by a group of professional developers. This does not mean that the development is any less professional, as many of these people are involved in other language projects, or are more intimately involved with this development than with other projects. Further, because of its wide availability, *gcc* is frequently used to experiment with new features and test new concepts, which are then quickly fed back into the standard distribution.

2. *Porting gcc to Fujitsu UXP/M*

In 1993, one of Australian Numerical Simulation and Modelling Services (*ANSAMS*) clients required a C++ compiler for their project, however, at that stage, none was available either from the vendor or from any third party. ANSAMS run a Fujitsu VP2200 Supercomputer, which is based on an IBM System/370 scalar unit and a proprietary vector unit. The operating system is UXP/M, Fujitsu's version of UNIX System V Release 4†. As there was no need to vectorise the code, it was decided to port *gcc*‡.

As no one locally had investigated such a port before, it was assumed it could be done within a couple of months. There was local knowledge of the instruction set and extensive knowledge of the system, however, aside from one installation exercise, no one had much knowledge of *gcc*. So it was decided to

† MSP, Fujitsu's version of MVS, is also available.

‡ In fact, two approaches were taken, porting *gcc* and obtaining AT&T's C++ System, which acts as a preprocessor for the native C compiler. The *gcc* port was completed before the paperwork for AT&T's C++ was finished!

begin ...

2.1 Preparation for the Port

As with any major project, one of the first steps is to study the available documentation, and *gcc* has lots. Unfortunately, this is where the first "problem" is encountered, the format of the documentation is *texinfo*, which is both suitable for transformation into both on-line information and printed output. To accomplish this you need to obtain the *texinfo* package which generates *TeX* output.

If you do not have *TeX* available, there is a *texi2roff* which attempts to print these documents with troff macros, however, this is only an approximation. This package is only designed to print the Gnu on-line documentation, not to process general *TeX* files.

Aside for documentation, you also quickly find references to *bison*, *gas* and Gnu's *ld*. *Gas* is Gnu's assembler, and is only available for a limited number of systems. In most cases it is not needed, although on a couple of systems, in particular MIPS, it is desirable. Gnu's *ld* is a linker, and again is only available for a limited range of systems. It is more important here to use the system supplied linker, if possible, as Gnu's linker can only create statically linked objects.

Bison is Gnu's parse table generator, equivalent to *yacc*. For a standard installation, it is not required, however, if any changes are made, it may be required. Further, if there are any problems with modification times, then *make* will attempt to invoke *bison*.

One other product that should also be considered is *gmake*, Gnu's version of *make*. Although this is not required, *gcc*'s makefiles are optimised to the use of *gmake* and a number of dependencies are not correctly interpreted by the system supplied version of *make*. The only effect of this is that extra work is performed, as various object files are recompiled when there was no need.

There is a *chicken-and-egg* problem with these products, as they all favour being compiled with *gcc*, however, they can all generally be compiled with the system supplied compiler.

2.2 The Port

Once the required tools were installed, in this case the *texinfo* package and *bison* (just in case), the task of porting *gcc* to the VP2200 began. We had deliberately decided not to use *gmake*, due to a desire not to port too many products before we had *gcc*.

Before outlining the details of the port it would be best to outline the structure of *gcc*. When this port was originally proposed, the current version of *gcc* was 2.3.3, while at the time of writing this paper it is up to 2.5.8. There are a number of minor differences between these versions, which had a minor effect on the port, some of which are described below.

Basically, *gcc* is broken into a front-end which is machine-independent, a back-end and a number of machine-dependent files. Although this is a logical breakdown, in practice the various passes are merged. The first step is to generate a partial syntax tree, which is latter converted through RTL's (*register transfer language*) to the appropriate machine code. The syntax tree is both machine and language independent, and this is what allows *gcc* to supply various front-ends for different languages.

The RTL's are machine independent and are used as part of a machine description pattern to define a number of possible *instructions* that a theoretical machine may understand (*e.g. movsi* which moves a word from one location to another). Some of these *instructions* are required, others are optional (*e.g. andhi3* which performs an *and* on a half word). If an optional operation is not available, then *gcc* will attempt to perform it in other ways, for example by extending it to a larger quantity and performing the operation on that. Machine descriptions can also have a **predicate** defining the type of operands they can take (*e.g.* register only) and a **constraint** which is used to control register allocation and reloading.

† Actually, this was installed on another system that also had *TeX*.

For a pattern to match an instruction it must have a matching RTL, and the **predicate** must be matched. The constraint can then select which of a number of alternate instructions will be used. See Figure 2 for an example of two machine description patterns. For a full description see the Gnu documentation, particularly *Porting Gnu CC*[1].

```
(define_insn "tstdi"
  [(set (cc0)
        (match_operand:DI 0 "general_operand" "d"))]
  ""
  "srda %0,0"
  [(set_attr "type "RS")
   (set_attr "ltorg" "0")])

(define_insn "movsi"
  [(set (match_operand:SI 0 "general_operand" "=d,dm")
        (mathc_operand:SI 1 "general_operand" "dimF,*fd"))]
  ""
  "*
{
  /* C code */
}"
  [(set_attr "ltorg" "4")])
```

Figure 2. Machine Description for tstdi and movsi.

These definitions are kept in a *machine description* file, which together with "two" other include files defines the *gcc* port. The two include files (which like any other include file can, and do include other files) are used to define the host machine (in a cross-compilation) and the target machine. In most cases the host and target machine are the same. The definitions for the host machine are mostly related to word format, so *gcc* can determine what sort of transformations are required for constants and other binary data.

The target machine description is much more complicated, it defines both the machine architecture, *e.g.* the number of registers, stack order, argument passing, and function prologue and epilogue code, and the format of the executable format, *i.e.* *a.out*, *COFF*, *ELF* and others. This also includes definitions for outputting generic functions into specific assembler instructions (*e.g.* **ASM_OUTPUT_LABEL**). This file generally includes a number of others, such as **svr4.h**, to define common definitions.

There is one other file that is included, a C file, which supplies any auxiliary functions which are needed to support output. These routines are often used to implement functions that are too difficult in a single-line macro. Finally, there can also be a file defining changes to the standard makefile.

To define a new port, generally, you create a directory for the CPU type†, if one doesn't exist, define a new system type and modify the **configure** program to understand these definitions. **Configure** and **config.sub** are used to match the *cpu-vendor-system* triplet to a system and create the appropriate symbolic links. You then need to create the appropriate files: the machine description, the host definitions, the target definitions, the makefile definitions, and the auxiliary output code. In many cases you can simply copy the host definitions and the makefile definitions from another system, with little change.

† Prior to *gcc-2.4* all the machine descriptions were kept in one directory. One of the changes in *gcc-2.4* was to separate the system specific files into separate subdirectories.

2.3 The Details of the Port

In the case of the ANSAMS port to UXP/M, it was made much simpler. Firstly, as UXP/M is a very standard SVR4 system, it was possible to take a number of the system related definitions from a number of other SVR4 system definitions, particularly the Intel i386 definitions. Making the port even easier was that a previous attempt had been made for *gcc* version 1 by Jan Stein (<jan@cd.chalmers.se>), and though this had a number of problems, it was a good starting point.

The first task was to upgrade this machine description to be compatible with *gcc* version 2. This mainly involved adding a few new required patterns, and modifying a number of others that had changed in some fashion. These changes primarily involved differences in the number or type of arguments to the pattern. To simplify the task, it was decided that any patterns that were not required and were causing a problem would be dropped, trusting *gcc* to generate an equivalent output.

This machine description was tested by trying to generate *gcc*. The standard procedure for making *gcc* is to generate a *stage 1* compiler, by compiling the source with the system supplied compiler, then using this stage 1 compiler to generate a *stage 2* compiler. The final step is to verify this compiler by compiling a *stage 3* compiler using the stage 2 compiler, and then comparing the output of the two compilers, if there are any differences then there is a problem, otherwise it is assumed to be okay. This test is very crude for developing a new port, and more robust testing needs to occur. However, the effort of just obtaining a working compiler is a feat in itself. Further, when compiling *gcc*, a number of steps use the compiler currently being generated and this provides quick feedback on problems.

The first attempts at compiling lead to interesting results. Unsurprisingly, the first couple of attempts failed to generate a workable compiler. At this stage we couldn't even make a stage 2 compiler. During this phase it was necessary to learn the details of RTL's, as the most common problem was an abort, with a dump of the offending RTL, which couldn't be matched. At this time it was also necessary to learn the various options to dump RTL's to a file at different stages of compilation.

Once the most basic problems were overcome, we struck the next major hurdle. This problem was a limitation of the System/370 addressing scheme and the original port. Addressing under System/370 is via a *base register* and an *offset*, with all constants or literals and branches specified by an offset of less than 4096 bytes. Jumps of larger than this can be made using a different instruction, but this is a not possible for addressing literals. This problem caused the original port to have the restriction that all functions had to be less than 4096 bytes long, as the base register could only be set at the start of each function.

The answer to this was to reset the base register within the function, however, *gcc* has no provision to do this. A number of different methods were attempted, with the final solution to be the use of a feature that had been added to *gcc* version 2. The feature was the use of *attributes*, so that every instruction that is generated also has associated with it a code length and a literal object length. Then during the generation of the function prologue to scan this table and insert appropriate instructions in the RTL to update base register. This instruction consists of branch to the following instruction. The code for a jump updates the base register if the certain options have been set. As well as updating the base register, code is also inserted to force a table of literals to be generated by the assembler. There are also a number of optimisations that are done, including testing if a jump has already occurred, and not generating a new one and making use of multiple base registers if any are free. The machine description for a *jump* instruction is shown in Figure 3, and the macro for generating an internal label is shown in Figure 4. The test for **tablejump_label** is because tables of addresses for *switch* and similar statements, may be compiled into the code section, and additional code for updating the base register should not occur.

Once this had been completed it was possible to generate a stage 1 and stage 2 compiler, and ultimately, to generate a stage 3 compiler that compared successfully with the stage 2 compiler. At this stage we thought we were almost finished. How wrong we were!

Although not strictly part of *gcc* for most uses it is necessary to install *libg++*, a library of standard functions for *g++*. The *libg++* package includes a number of test programs to confirm that it is

```
(define_insn "jump"
  [(set (pc)
        (label_ref (match_operand 0 "" "")))]
  ""
  "*
{
  extern int big_code;

    if (big_code) {
        return \"l 15,=a(%l0)\;br 15\\n\\tltorg\\n\\t.=(.+1)\\\\/2*2\";
    } else
        return \"b %l0\";
}"
  [(set_attr "type" "RX")
   (set_attr "ltorg" "0")])
```

Figure 3. Machine description for a jump on System/370

```
#define ASM_OUTPUT_INTERNAL_LABEL(FILE,PREFIX,NUM) \
  { \
    extern int big_code, tablejump_label, base_reg[]; \
\
    fprintf((FILE), "$%s%d:\n", (PREFIX), (NUM)); \
    if (big_code && tablejump_label != (NUM)) { \
      fprintf((FILE), "\tbalr %d,0\n", base_reg[0]); \
      fprintf((FILE), "\tusing .,%d\n", base_reg[0]); \
    } else \
        tablejump_label = 0; \
  }
```

Figure 4. Macro for generating label on System/370

installed correctly. The next few months we learnt far more about the internals of *gcc* than at any previous stage. We discovered how *g++* creates constructor and destructor on SVR4 systems, how *gcc* optimises tests, and how bit manipulation is handled. All of these caused various failures during the test. It soon became apparent that just generating the compiler was a minor step in building a compiler that produces *correct* code.

The approach that was found to be most successful in correcting problems was to delete any machine description patterns that were found to be incorrect, rather than try to correct them. The reason for this was that by this stage the most common problem was one of side-effects, particularly, registers incorrectly set. One example of this was with the tests on 8 and 16 bit quantities. The original code did left shift followed by a right shift, thus loosing the top 24 or 16 bits. Unfortunately, at times *gcc* is smart enough to use the top bits, particularly during cast operations, *e.g.*

```
long l;
...
if ((short) l) ...
```

This is best handled by allowing *gcc* to do the type casting explicitly and then the test. Obviously, most of these types of problems occur when the code is highly optimised, often they are masked by turning off optimisation and blaming the optimisation process, rather than tracking down the problem.

Another interesting problem occurred during the assignment of a structure of length 4 bytes, *i.e.* two shorts or 4 chars. Without optimisation this causes a copy of each element, however, with optimisation *gcc* produced a single copy of 32 bits, which highlighted a bug in memory to memory copy.

It should not be assumed that all the problems were with the System/370 machine description, a couple were traced back to either bugs in *gcc*, or to implementation restrictions which were not documented. An example of this was in the generation of branch instructions. *Gcc* can reverse the order of a branch in certain circumstances (*e.g.* **bne** can be come **beq**), however, if it is also necessary to generate a temporary register at this point, then the reverse is not performed. This made it necessary to permanently allocate a register for use in branches.

At this stage, *gcc* was released for general use, only six months after it was started. At this time it was felt that the project was nearly complete when suddenly† FSF released a new version of *gcc*, 2.4.5. Although the new release was not a major upgrade, it still had enough features to cause problems. In particular the new features included splitting all the machine descriptions into separate directories, adding a new required *instruction* and, probably the most difficult, adding new procedures to handle floating point numbers. As the format of floating point numbers is different under System/370 to any other system, it was necessary to write new sections of *gcc* to handle the conversion between System/370 and *gcc*'s internal format. These changes have since been merged back into the official *gcc* distribution.

By now it was decided to call a halt to the current *gcc* development for the VP2200. It was generating suitable code for the original client, and seemed to pass all tests required of it.

2.4 The Final Wrap Up

At this stage it was decided to return these changes into the official *gcc* distribution. A message was sent to the FSF that they were available along with some details of other problems found within *gcc*. This brought the response that one had just been completed for MVS, and that the two needed to be merged together before this one could be accepted. On looking through the "official" version, it was interesting how similar it was to the original we had started with. It had started from the same version by Jan Stein and had then addressed the same problem of reallocating base registers for code over 4096 bytes. Their method was to call a function when generating the code, rather than through *attributes*, however, the ultimate effect is very similar.

Other differences, were related to MVS and differences in both the assembler and in the linkage conventions for some functions. On the other hand it was also obvious that this version had not been extensively tested, as most of the tools were not available under MVS. The evidence for this came from the fact that some code in the original source that had been found to cause problems, was still in this machine description.

The other outcome of contacting FSF, was that I was made aware of the *gcc*'s developers mailing list, and the schedule for updating *gcc*. This gave a much clearer picture of how *gcc* is being updated and the work involved. Finally, information about the testing of *gcc* ports also became available.

The standard package for testing *gcc* ports is called *c-torture*, which is basically a large script and example of problems that have previously been encountered within *gcc*. Another package supported by the FSF for automated testing, *dejagnu*, is planned to be used in the future, although, who is going to implement this change-over is not yet clear.

At the time of writing this paper, *gcc* is up to release 2.5.8, with planning underway for release 2.6. The VP2200 port is still based on the 2.4.5 release, awaiting time to complete the merge with the MVS version and to test changes for 2.5.8.

† Well not so suddenly according to FSF's timetable, but it surprised us.

3. The Future of GCC

The development of *gcc* is itself an interesting study. It is a project that is undertaken by hundred, if not thousands, of part-timers around the world. It is a demonstration of the power of the Internet to bring together experts from around the world to work on a large project. The FSF project consists of a very active mailing list, a site which is updated weekly with all the latest patches, and a small group who control the direction of the project. Outside of this small group are a number of others who are well aware of the intimate details of some section of *gcc*, for example the description for a particular system. Finally, there is a wealth of talent using *gcc* and reporting back problems or requesting, and often implementing, new features, such as inlining of code, or better optimisation schemes.

Currently, the testing of *gcc* is done manually by volunteers who pick up a copy of the changes, update their source and then run *c-torture*, finally reporting back the results. The are plans to automate this procedure, by mailing out the changes and having a testing script install the patches, generating the new compiler and then reporting back the results. This will obviously greatly improve the quality of the compiler.

A different side of the FSF project is the reaction time to a problem. As the users of the product generally have access to the developers, and those with relevant knowledge can fix the problem themselves and report back the fixes, it means that the time from identification to correction can be measured in terms of days or weeks, instead of months or even years which you find from most software developers. Also because of the widely diverse nature of the developers, almost always someone with an intimate knowledge of the problem and the solution will be available very quickly.

In the long term, *gcc* will continue to expand to new systems, ports are already underway for the Pentium and 64 bit MIPS architecture. It will also be used as the basis for further language development and testing new ideas in compiler technology.

4. References

[1] Stallman, Richard M., June 1993, *Porting GNU CC: For GCC Version 2.4*, Free Software Foundation, Cambridge.

Review of BSD/386 †

"It works! It works!"

Reviewed by Lou Katz
<lou@metron.com>

Tuesday: FedEx package arrives. Open it. Very professional looking package inside. "BSD/386. March 1993. Version 1.0." Traditional shiny, slick paper. Inside of box is a CD-ROM with this weird gekko printed on it. Only one problem. I don't have a CD-ROM drive. Well, when all else fails, read manual. What do you know? It's very readable!

Wednesday: Order the latest and greatest CD-ROM drive. Then worry — will this system recognize it?

Saturday: Pick up new CD-ROM drive from El Cheapo Computer Company. Connect it to my SCSI controller. Boot machine. Well, it only sort of works. Try another SCSI address — 6! CD-ROM drive sighted by boot. Machine comes up and mounts drive! Decide to backup existing disks before proceeding further.

Sunday: Finish backup onto ancient, slow but trustworthy QIC-24 tape drive. Ancient, venerable machine, survivor of the infamous FaceSaver campaign is a 386-16 with 10 MB of memory. Machine currently has one MFM drive (about 100 Mb) and one SCSI drive (1.2 GB).

Now try to install this puppy. The goal is to have BSD/386 running without losing the ability to boot either DOS or Xenix from existing partitions. Anticipated the usual nightmares of incompatible disk partitioning schemes. After further reading of the manual, it was apparent that I would have to move the Xenix partition. Took all day.

Finally started the BSD/386 installation — it was a breeze! First convince DOS that I only had a 1 GB disk, and to use a partition at the beginning. Then BSD/386 used the rest, out to its real limit! Then load the base system.

Since I had the CD-ROM version and enough space; I just loaded almost everything and went to sleep.

Now to configure the system — lets see — need UUCP. Yup. But wait! My modem is not in the list of modems...ahhh...I HAVE SOURCE! Just like the olden days. Quick hack to put in my own modem's idiosyncrasies. Bidirectional TTY ports work fine. I need PCNFS. No problemo! Just RTFM and turn on the right daemons. Now I am a file server and my wife is happily working away on her DOS/Windows machine. PostScript printer which needed cat2dit to work with Xenix troff now up and running directly out of groff.

How about real adventure? Install SLIP/PPP mods to kernel. Kernel rebuilds right off of the CD-ROM by a neat hack. Bringing up PPP itself takes a little more work, mostly because the how-to's and why-for's aren't exactly clear in any book I could find. It now works, and I have my very own internet connection.

Import Eudora and POP. POP installs right away. Now mail can be read from my Mac (don't ask why!). Get a Mac-to-lpd utility. Mac printing spools through the BSD/386 lp spooler to printer. No longer have to push that dorky little switch on the back of the printer where I can hardly reach it (and can't see the interface number in the little window anyway) to go back and forth between local talk and parallel port.

Need to be able to convert AutoCAD plot files to EPSI form (PostScript with included TIFF preview image). No Problem! Get small utility from the net. Use ghostscript (provided) and pbmplus (provided). Hack-a-bit and there you are, thank you.

It is really a great relief having this system. It is even better than the good old days. First, anything I thought I might want seems to be there. Second, there is a VERY active mailing list which has an excellent signal-to-noise ratio and carries lots of good info. Third, the system is supported! Response to phone calls was good, though E-mail response to the reporting of bugs or problems was uneven. Unlike any of the other systems I have used (SunOS, Solaris, HP-UX, IRIX, Xenix, SCO UNIX, AIX, A/UX) there are no crucial missing pieces — no 'PostScript not included' nor compiler to be found in a separate licensed package.

I am hardly a speed or performance freak (with my antique equipment), but it seems that this system, under somewhat greater load due to the PCNFS functions is about the same speed as the Xenix system I ran on identical hardware. It seems to support enough of the mainstream peripherals so that I have had no problems with borrowed SCSI DAT drives as well as my old QIC cartridge drive. The system comes with X11, but I haven't exercised it yet, since I need a more reasonable VGA card first.

† This is a re-print from ;login, the USENIX Association Newsletter, Volume 19 Number 1

Besides all the utilities you would expect to find in a UNIX nowadays, as well as full, up to date networking support, there are also perl, elm, netfax, mh, TeX, nenscript, ispell, RCS, and access to DOS file systems on hard and floppy disks. There is enough interest on the net in this system that lots of software seems to come with BSD/386 as one of the possible compile options. AND THERE IS SOURCE (remember source?). If the man pages don't tell you what you want to know, you can always read it. And you can change it too.

This is not a perfect product, but in my environment it has been very stable, had all the features and functions I needed and does what I want. I would not hesitate to use it in a production setting, nor to install it on a client's machine. Some of the users have reported BSD/386 configurations running as network access servers with multiple dial-in lines, and as file servers. Unlike other commercial suppliers, the folks at BSDI have not gone crazy and have not priced a "PC" product like it ran on a mainframe. Further good news is that they expect to provide support for some of the binary formats of other systems in the (near) future. This would make it very attractive to configure, for instance, database and word processing applications in real commercial environments, because the clients could buy and use commercially and widely available packages.

Most of the problems I had were with the documentation. Many of the man pages were obviously the original BSD pages, and had not been edited to change path or file name references. Although one is supposed to be able to make changes to source and to compile a package from the CD-ROM, this only worked some of the time — the scripts to point to the revised source didn't always work. This is more of an annoyance than a fatal flaw, but it does waste some time. I eagerly await the 1.1 release, which may have some of the binary support and other neat features. If the BSDI folks put a reasonable effort into documentation and bug fixes this system could be around for a long time!

As Karl Malden might (but doesn't) say, "BSD/386, don't leave home without it!"

- BSD/386 V1.0 is available from Berkeley Soft ware Design, Inc.; 7759 Delmonico Drive, Colorado Springs, CO 80919; Phone: 1-800–4BSD, 1-719-593-9445; Fax: 1-719-598-4238; Prices for CD-ROM Source + Binaries $1045, Binaries only $545, price for Tape slightly higher. Version 1.1 is due to be released soon.

Computers Could be Like Autos †

Humor by Dave Taber
<David.Taber@Eng.Sun.COM>

What driving your car would be like if operating systems ran it:

Windows: You'd get into your car and drive to the store very slowly because five boulders are dragged along behind the car.

Windows NT: You'd get into your car and write a letter that says, "Go to the store." Then you'd get out of the car and mail the letter. The dashboard of the car would glow knowingly.

OS/2: After fueling up with 60 gallons of kerosene, you'd get into the car and drive to the store with a motorcycle escort and marching band in procession. Halfway there, the car would catch fire.

Taligent: You'd walk to the store with Ricardo Montalban, who tells you how wonderful it will be when he can fly you to the store in his jet.

UNIX: You'd get in a diesel locomotive and start looking around for the "go" switch. The control panel has 150 unmarked levers. The speedometer calibrations start at 90 miles an hour, and go up from there.

† This is a re-print from ;login, the USENIX Association Newsletter, Volume 19 Number 1

An Update on UNIX-related Standards Activities †

by Nick Stoughton
USENIX Standards Report Editor
<nick@usenix.org>

A standards committee was formed to develop a new standard for Open Systems. The project was approved and the committee got down to work.

For forty days and forty nights the standards committee ate nothing, but wrote their standard. They became exceedingly hungry. Then the devil appeared to them and tempted them to get food by going to ballot early. "It will prove you truly are a great standards committee," he said.

But the standards committee told him, "No! For it is written that bread will not fill a standards writer's soul: obedience to every word of the procedure is all we need."

Then the devil took the standards body to a great International Organisation, and said "If you rewrite your standards in a computer Language Independent form it will prove that you are truly a great standards organisation. Angels will appear to prevent you from smashing on the rocks below." The standards committee retorted "It is also written that existing practice shall be followed, and there is no existing Language Independent practice to follow."

So the devil took the standards committee to the peak of a very high mountain, and showed them the governments of the world in all their glory. "Every one of these governments will require all their people to adopt your standard, if you will worship me and be prepared to invent a set of new communications protocols."

But the standards committee said "Get thee behind us Satan. The procedures say follow only existing widespread practice. Obey only the IEEE."

Whilst some liberalization of the facts has been used to make them fit the story above, all these things have happened over the past few years within several standards bodies.

- The Language Independence Issue raged within the POSIX world for two years or so, until last summer, when, finally, the IEEE agreed to drop the requirement of ISO that all new and existing standards had to be written in a language independent form. This would have meant, for example, that the existing ISO 9945-1 (POSIX.1), which was written using the C programming language, should be rewritten. As suggested by the parable above, this was viewed by most people within POSIX as akin to taking yourself to the top of a tall building a jumping off. When Jesus was tempted in the wilderness, I am sure He had a far higher degree of certainty of survival if he had thrown himself from the pinnacle of the temple. For POSIX, the choice was between following the mandates of ISO, taking forever to produce a standard that no-one could understand or use, and ignoring ISO, thereby risking international acceptance and status for the resulting, language dependent standards.

The third temptation in the parable is probably the most interesting. Why shouldn't Open Systems Standards be invented? The old saying "You can take a horse to water, but you can't make it drink" springs to mind. Good standards are ones that people will want to use. Bad standards, even if they are mandated (e.g. the set of OSI protocols selected for GOSIP), will never gain widespread acceptance. When making a standard, most bodies look around to see what everyone is doing in the area in the lack of a real standard. Big companies, like Microsoft, say "We are so big and powerful, we'll do our own thing and everyone will follow, because we have a good marketing department!" More formal standards bodies document existing practice, in formal language. Sometimes a little massaging is needed to fit together the pieces in a smooth fashion; occasionally there is a glaring hole discovered by the process that a small invention could be allowed to cover. But there lies a slippery slope. Once you allow one little bit of invention, you allow another, and another, till there's little left of the original base document.

Standards bodies are made up of technical people, knowledgeable in the specific area they are standardizing. So why can't they invent new things in their area? Why can't Microsoft rule the world with Windoesn't (Windows NT)? What was wrong with the OSI protocols? Well, probably the single biggest thing is that Internet Protocol Suite, including TCP/IP and all the related protocols, is a very low cost, higher performing, and embodied in an enormous existing network. True, OSI and TCP were being developed concur-

† This is a re-print from ;login, the USENIX Association Newsletter, Volume 19 Number 1

rently, and at the time OSI was being made, not all the above were true! Nevertheless, there was enough existing practice to show that TCP/IP was going to succeed. What OSI produced looks good on paper, at a high level. But what people were doing at the time was ignored. Apart from Government applications, OSI is rarely in use, whilst the Internet, well, need I say more...?

Report on POSIX.4: Real-time Extensions

Lee Schermerhorn <lts@westford.ccur.com> reports on the October 18-22, 1993 meeting in Bethesda, MD:

The POSIX.4 Working Group Chair was unable to attend the meeting because of "real work" commitments. The Vice Chair was also in absentia because of imminent fatherhood, of which there seems to a lot going around lately. The forced absence of the Chairs left the running of the meeting to the "third string"— the Secretary of the Working Group who happens to be your POSIX.4 snitch reporter for this meeting.

So here's the plan: first we'll review the status and schedule of the documents that have already been reported out of the working group for balloting; then we'll cover the activities of the Working Group during the week.

Balloting Status and Schedule

- POSIX.4 **aka** POSIX.**1b:** It's official. The IEEE Standards Board approved Draft 14 of the POSIX.4 Realtime (one word according to POSIX.4) Extensions Standard at the mid-September meeting. At nearly the same time, the IEEE was also renumbering the standards to confuse the innocent. Because POSIX.4 is cast as modifications and additions to POSIX.1, the IEEE has renamed POSIX.4 to POSIX. 1b. Sort of makes sense, except that POSIX.1b will be published well over a year before POSIX.1a! So it's best not to think of the letter suffix as a revision.

- It appears that POSIX.4/1b will be published as a merged document to replace the current POSIX.1-1990, in the March timeframe. In the meantime, the full Draft 14, as opposed to the small set of changes that were actually balloted in the last recirculation, is available from the IEEE at a "modest fee."

- POSIX.**4a aka Pthreads aka** POSIX.**4c:** Draft 8 of Pthreads is being recirculated for a 10 day ballot period from November 1-12, 1993. "Recirculation" means that only the changes from Draft 7 are open for comment and/or objection. JohnZ, the Pthreads (and POSIX.4) technical editor, expressed the opinion that one additional recirculation will be required to clean up loose ends. This would make it unlikely that Pthreads can be ready for the March 1994 Standard Board meeting. The June 1994 meeting is a more likely target.

- Note that POSIX.8, Transparent File Access (TFA), is also expected to be approved at close to the same time. The System Interfaces Coordinating Committee (SICC) has noted this and has determined that Pthreads will be merged with the then merged POSIX.1/1b standard before TFA. It remains to be seen when, and in how many volumes, the results will be distributed.

- POSIX.**4b aka** POSIX.**1d — more Realtime extensions:** Draft 8 of this document was reported out of the working group for ballot again in July. The first ballot is open for 30 days starting on 1 Nov 1993. Those of you who follow comp.std.unix may recall that a call went out for all the UNIX true believers to join the balloting group to make sure that those wild and crazy POSIX.4 real timers don't do something unclean (in the UNIX sense) to POSIX.

- POSIX.4b/1d contains several additional real time extensions, including:

- The *fadvise()* file advisory chapter that replaces the "real time files" chapter that was removed from the last draft of POSIX.4.

- A "Sporadic Server" chapter for budgeting CPU time to aperiodic events so that they can be handled via Rate Monotonic Scheduling analysis, with guaranteed deadlines.

- Definition of Process Virtual Time Clocks under the POSIX.4 Clocks and Timers interface. These are analogous to the virtual "itimers" of BSD and SVr4, and are included primarily in support of the Sporadic server.

- "Device Control"— really *ioctl()*, but with some "enhancements" to address some standards/portability related issues that kept *ioctl()* out of POSIX.1. Wouldn't it be nice, if before the balloting is over, this ends up as good old *ioctl()*?

- "Interrupt Control"— connection of user programs to interrupt sources. Two modes of operation: one where an application requests notification via signal when a particular interrupt occurs — without having to write a driver — and one mode where a user specified function is run at interrupt level. I suspect this one will have a lot of difficulty in balloting.

- POSIX.**13 — Realtime Application Environment Profiles:** It is over a year since the first round of balloting on the POSIX.13 profiles closed. Ballot resolution has been slow because of three gating issues:

- The POSIX.13 Profiles reference the POSIX.4 and POSIX.4a Draft Standards and would, in any event, have to wait for both of these Standards to be approved.

- The POSIX.13 Draft contained four (4) profiles in a single document. An earlier draft of the ISO document that defines profiles (TR 10,000) apparently forbade multiple profiles in a document.

- Three of the four POSIX.13 Profiles restrict an application to a subset of the interfaces in POSIX.1. PASC Profile Steering Committee (PSC) rules for profiles — non-existent when POSIX.13 first went to ballot — forbids a profile to specify a subset of a base standard.

The first roadblock is in the process of resolving itself. POSIX.4 is a done deed, and Pthreads should be approved by mid-'94. A later draft of TR10,000 now allows multiple profiles in a "Standard Profile", if a number of conditions regarding cohesiveness, etc. are satisfied.

The final issue is one which has consumed vast amounts of PASC meeting time, in Working Groups, PSC meetings, SEC (Sponsor Executive Committee) meetings, and in hallway/bar room conversations. An intensive effort during the week of meetings by an Ad Hoc of the SEC has resulted in a compromise, of which more later.

- **LIS — RIP Or "What ever happened to POSIX 4c?"** POSIX.4c was to be the Language Independent Specification (LIS) of POSIX.4. But when, in July, the SEC rescinded the requirement for Working Groups to produce LIS for all PASC Standards, the POSIX.4 Working Group immediately voted to stop work on their LIS. That decision was confirmed again at this meeting.

Thanks to the efforts of Michael Gonzalez, the Working Group has a nearly complete first draft of the LIS. Michael said that he wanted to complete the remaining couple of sections, and would like to see the results be made available to anyone interested. The WG has been assured that it will be no problem to arrange to have the completed, unreviewed draft available for ftp from both the IEEE's emerging SPA (Standard's Process Automation) system, or from Michael Gonzalez's University system at University of Cantabria, Santander, Spain.

Working Group Actions and Plans

With all of its documents, except for POSIX.13, done or out for ballot, one might well wonder what the POSIX.4 working group is doing meeting in exotic places like Bethesda, MD. Two things: planning for additional drafts to standardize additional interfaces, and POSIX.13 ballot resolution.

First, POSIX.13 ballot resolution: The Profiles ballot resolution effort had degenerated to getting the issue of specifying subsets of POSIX.1 resolved. Because this issue is one of inter-Working-Group coordination, it required a lot of interaction with members of the ad hoc committee established to report back to the SEC. Several members of the Working Group, who are also POSIX.13 technical reviewers, — Andy Wheeler, Joe Gwinn, and others — spent a couple of hours every day, Monday through Thursday, in the ad hoc; reporting back to the Working Group daily on progress or the lack thereof.

The ad hoc made a fairly thorough review of the issues, noting that the primary objection to the subsetting was more religious and political than technical— that is, the "dilution" of the POSIX name if it were associated with anything less that full POSIX.1-1990 as we know and love it. In truth, though, a number of technical issues did surface concerning testing of subsets, the effort of respecifying the semantics of POSIX.1 with formal subsets, the integration of Standards that later modify the full POSIX.1, such as Pthreads, POSIX.8, etc., with a subsetted POSIX.1.

Ultimately, the ad hoc placed a resolution before the SEC to suspend the PSC rules for the "special case" of real time subsets for POSIX.13, and allow POSIX.13 to specify the subsets in the profiles. After an hour and a half of debate in the SEC, the motion passed, with an amendment requiring that the POSIX.13 balloting group be reopened for a minimum period of 30 days. The primary objections to the motion were not in objection to allowing POSIX.13 to subset POSIX.1; so much as to having the subsetting done in POSIX.13. The view was that if subsetting were to be done, do it once and for all in POSIX.1. This would probably hold up not only the POSIX.13 profiles for a couple of more years; but any extension standards that happened to coincide with the subsetting revision. The resulting resolution will provide the embedded real time systems community — users and vendors alike — with a standard profile that describes the runtime environment that the target applications can depend on, and that conforming implementations must, as a minimum, support. The Chair of the SEC pointed out that later, when extensions to POSIX.1 settle down, and the real time (subset) profiles have had some use, might be an appropriate time to formalize the subsets in the POSIX.1 standard itself.

The SEC resolution now clears the way to complete the POSIX.13 first round ballot resolutions. But, a fair amount of work now falls on the Technical Reviewers to add the normative text that effects the subsetting to the next Draft. A not so small group of volunteers signed up to work on and review drafts of the subsetting text. The approach discussed in the Working Group is to prescribe what functions are available to Strictly Conforming Applications for each profile. Where some subset of behavior of a required function is not required, it will be explicitly unspecified. For example, *open()* of a non-existent file in a profile with no requirement for a file system will be unspecified; rather than, say, return a specific error. Initial drafts should be available for the January meeting.

The other new work item was additional interfaces for — call it POSIX.4d. The Working Group has had a running list of features and functions of real time systems that are potential candidates for future Real Time extensions of POSIX.1. But, the Chair has instructed the Working Group that we won't generate another PAR unless concrete proposals, including base documents, are presented, backed by a strong commitment to see them through to standardization. The Working Group reviewed several proposals, a couple of which had fallen out of earlier work such as POSIX.4b because of lack of consensus at the time that '.4b was otherwise ready for ballot. The new proposals include:

- **Typed Memory**: This is essentially an additional type of memory object, like /dev/mem, that represents different views of special physical memories, such as external memory modules visible on multiple busses. Extensions to *mmap()* support additional flags for dynamic/contiguous allocation by the object and functions to obtain an offset within the object from the address returned by *mmap()*, needed with dynamic allocation.
- **absolute nanosleep()**: This is an extension to the POSIX.4 nanosleep() function — a new function, actually — to wait until a specified time using the POSIX.4 high resolution timespec.
- **Barrier Synchronization objects**: Independently of these being proposed for POSIX.4, they were also spec'ed by POSIX.14 — the Multiprocessing Working Group. Because POSIX.14 is a Profiles Working Group, they need to have any new interfaces that they propose put into one of the System Interfaces Working Groups' drafts. The POSIX.14 group had already made tentative arrangements for a number of new synchronization primitives to go into POSIX.1a, so the POSIX.4 Working Group may drop this.
- **Enhancements to POSIX.4**: Yes, the ink is barely dry on the official standard approval, and we're thinking about "enhancing" POSIX.4. That's because some people have implemented, or are in the process of actually implementing it. The one extension presented was to POSIX.4 message queues to make registration for notification of message arrival, via *mq_notify()*, optionally persistent.

Other "housekeeping" items, such as resolution of conflicts or unintended ambiguities between POSIX.4 and Pthreads, may come up in time for a POSIX.4d effort.

The January working group is expected to me more of the same: POSIX.13 ballot resolution and new proposals. There are also coordination issues between the POSIX.4 and POSIX.20 — Ada binding to POSIX.4 — Working Groups, and with the Distributed Realtime group, POSIX.21 to be addressed as they arise.

Report on POSIX.7: System Administration

Matt Wicks <wicks@fnal.gov> and Keith Duval <duval.vnet.ibm.com> report on the October 18-22, 1993 meeting in Bethesda, MD:

POSIX.7 is divided into three separate groups, each producing their own standard:

- POSIX.7.1 - Printing Administration
- POSIX.7.2 - Software Installation and Management
- POSIX.7.3 - User and Group Administration

Of all the work of POSIX.7, the work of the printing group is most advanced, with the initial formal ballot conducted in June-July, 1993. The print standard is based on MIT's Palladium, which is a distributed print management system and also the base technology for OSF's offering in the print management arena.

The working group explicitly decided to reject using *lpr* or *lp* as the basis of the standard, believing that neither really addressed all of the issues of a distributed printing system.

The ballot was generally positive, so there seems to be some willingness within the standards com-

munity to approve System Administration based standards. It remains to be seen if both vendors and users are ready and willing to migrate to a totally new printing system.
Commencing the week of October 18th, the POSIX.7.1 Printing standards group met in Bethesda. A great deal of progress was made toward producing a final document which satisfies the overwhelming majority of interested parties, and while resolving the objections and comments is a daunting task, the committee was formed, and objectives and means for achievement of the goals at hand were delineated.

Moreover, there were a number of excellent suggestions from the balloters which will improve the overall standard and implementations derived from same. Further, it was readily recognized by all who participated in the arduous task of interpretation and response to the objections and comments in Bethesda, and by all who are credible in the field, that this represents a significant enhancement of the art with respect to distributed systems technology. Finally, it was generally agreed that 'times have changed' and, if we allow ourselves the intellectual stimulation, change is good for us. In the print technology area, it was increasingly obvious during the week that the 'old' is just a bit too old to be relevant any longer, other than as grist for whimsey and fond recollection of much simpler systems challenges and times.

The Software Management standard is based on Hewlett Packard's software installation package, with some contributions from the SVR4 software installation package. (The HP system is also the base technology for the software management portion of OSF's Distributed Management Environment.)

There are two primary goals of the standard. One goal is to provide a standardized command line interface for all of the typical software management tasks. These include commands to install and remove software, configure software, and list and verify software. This goal allows administrator portability since the software management process will work the same on different machine types.

The second goal is to define a standard software package layout. This goal allows media portability. Software packaged in the standard layout would be able to be managed by any POSIX conformant implementation.

For a good explanation of additional details of the standard, I recommend obtaining a copy of the proceedings of the most recent USENIX LISA conference. Barrie Archer provided a very good paper that not only explains the standard, but also some of the reasons why certain decisions were made.
I have been involved in the Software Group since its formation over two years ago. This meeting has a significantly different flavor as the group is nearing completion of its initial work, planning to go to ballot after the April, 1994 meeting. Although there were several heated discussions on several technical issues over the course of the week, in general the work was focussed on "fine tuning" the document.

The next two meetings will be dedicated almost exclusively to editorial issues and attempting to resolve any discrepancies between different sections in the document.

A separate snitch report is being written by a member of this working group. However, I did want to use this opportunity to encourage other people to get involved.

The User and Group Administration work is in its early stages and is being done primarily by two individuals (a third person joined them this week) both of whom are vendor representatives. Here is an opportunity to get involved and make a difference in the standards arena. Otherwise, you will have to accept what is produced by a very small group of people.

Participation in POSIX does take time, but it is well worth it. Send me mail if you would like more information on how to get involved.

Report on Automated Testing BOF

Kathleen Liburdy <liburdy@hubcap.clemson.edu> reports on the October 18-22, 1993 meeting in Bethesda, MD:

The fourth Automated Testing BOF met on Wednesday afternoon during the week of POSIX in Bethesda, MD. This group provides a forum for the discussion of alternative and progressive approaches to conformance testing. Announcements and discussion related to the group are posted to the mailing list *<oats@stdsbbs.ieee.org>*.

In the opening remarks, a Project Authorization Request (PAR) was announced for POSIX.5 (Ada) test methods. This project will explore the potential application of formal specifications and automated testing in POSIX test methods. In particular, the assertions will be developed using the Clemson Automated Testing System (CATS) assertion language. These assertions are English-like in nature and can be automatically translated into an executable test suite. The decision to apply formal specifications in test methods development was strongly supported by the POSIX.5 working group. The PAR was approved by the Sponsor Executive Committee (SEC), and the first meeting for this effort is scheduled for January 1994 at the POSIX/Irvine meeting.

The first presentation was an update on the ADL Project by Shane McCarron. The ADL Project is a four year project sponsored by the Japanese Ministry of International Trade and Industry. The project is being managed by X/Open and the primary research is being conducted by Sun Microsystems Laboratories. The mission of this project is to improve the test suite creation process through automation.

Each version of each deliverable is being reviewed by a public review group (*XoPub-ADL@xopen.co.uk*). Several sets of draft documents have been submitted for public review, including version 0.5 of the ADL Language Reference Manual and ADL Translator Design Specification. The ADL Project Quality Plan has also been delivered, and is now complete at version 1.0. The next deliverable of the ADL Project is November 1, which includes ADLT Design Spec 1.0 Alpha, ADL Language Reference Manual 1.0 Alpha, and other related documents. All these documents will be placed on the uunet ftp site ftp.uu.net under the directory /vendor/adl.

A technical briefing by Alberto Savoia of Sun Microsystems on the ADL Project was announced for the following morning, and all AT BOF participants were invited to attend. Alberto agreed to present a technical update on the ADL Project and discuss related issues at the next AT BOF in Irvine, CA.

The next presentation was "Automated Testing of POSIX Subsets" by Jim Leathrum. As part of the continuing development of CATS, experiments have been undertaken to investigate the issues associated with specification and execution of tests for subsets of standard interfaces. As part of this work, the CATS test harness has been enhanced to allow the test developer to create subsets of both the specification and the implementation of the system under test. A version of the CATS facility with the subsetting capabilities and the corresponding user manual are scheduled for release in January.

The ability to subset specifications and implementations in the CATS environment has led to many new issues which could not be addressed before. Jim discussed issues such as subset integrity, testability and granularity which are currently being investigated with the CATS facility. Many of these issues were also raised by the POSIX Subset Ad Hoc group. At the conclusion of the presentation, Lowell Johnson asked about the possibility of applying CATS to the POSIX subset dilemma by implementing and experimenting with some of the proposed POSIX subsets. Jim agreed that this would be an interesting application for CATS and indicated that this idea would be considered in future work.

Roger Martin, chair of the Steering Committee for Conformance Testing (SCCT), expressed an interest in being responsive to issues raised in the AT BOF. He stated that the decision in April 1993 to rescind testing requirements should be viewed as an opportunity to explore new approaches to testing. He also announced an invitational workshop on alternative testing methodologies to be hosted by NIST. The precise date for this workshop has not been determined, but the general time frame is spring 1994. The purpose of the workshop is to bring together major players in the field of conformance testing and collectively identify ways to cooperate in the pursuit of improved testing capabilities.

The fourth issue of the OATS newsletter was distributed. In addition to articles related to the AT BOF presentations, the newsletter includes "CATS in the Classroom," "Software through Pictures: The T Tool," and "DejaGnu Product Description." Submissions for future issues are invited and should be sent to <*liburdy@hubcap.clemson.edu*>. Requests for issues of the newsletter may also be sent to this email address.

Report on Fault Management Study Group

Stephen Hinde <S.Hinde@frec.bull.fr> reports on the October 18-22, 1993 meeting in Bethesda, MD:

Do you know the difference between Fault Tolerance, High Availability, a fault, a failure and an error? If so you could consider joining the "Fault Management Study Group" at the next POSIX meeting at Irvine.

October was the first meeting of this group, following BOF sessions at the two previous POSIX meetings. The status of the group is a "Study Group" preparing a Project Authorization Request (PAR). The healthy participation would indicate that Fault Management is something organizations are interested in sending people to work on which is one of the basic criteria for success in these hard times.

A number of existing documents are being studied as base documents, including the "UNIX International High Availability Working Group report," which was contributed by UI at the previous BOF, and a draft of the "ANSI X3.T8-1993 Fault Isolation Information Characterization for Information Technology."

The group found itself with an awesome "laundry list" from the submitted requirements. The requirements ranged from a framework to improve application availability to a framework to improve platform availability. The required system scope is not limited, and ranges from single CPU systems to Symmetric Multiprocessor systems and clustered systems.

The group debated whether it was to be a new PAR, or a sub-PAR of an existing group. The word "Management" had led some to ask whether the group would end up as a sub-PAR of POSIX.7, the System Management group. However some of the objectives of the group were clearly outside the area of System Management, and a number of alternative titles for the group were being considered, for which the current favorite was "Services for Dependable Systems." The PAR/sub-PAR debate will be easily settled when the PAR submission and scope documents are complete.

The work of fleshing out a Fault Management Process Model largely dominated the meeting, this is a model that would allow the decomposition of the error detection and error treatment steps, and allow the identification of the APIs involved. The model was mapped against several implementations as a sanity check. The behavior and definition of the building blocks of the process were examined, including Error Detection, Symptom Encoding, Error Logging, Diagnosis, Notification, Reconfiguration and Recovery. Possible areas for standardization could include APIs for Error Logging, Error Reporting, APIs for recovery modules, and a "fingerprinting" technique for uniquely identifying faults.

Bradford Glad, of ISIS, gave a presentation of HORUS a distributed toolkit layer designed to build distributed fault tolerant systems. The key ideas include virtual synchrony, a fault tolerant membership service, process groups, and a reliable broadcast protocol. New work includes a high level abstraction called the Uniform Group interface.

This working group spent an intensive week looking at a wide range of topics in the fault tolerant arena. The acid test is going to be selecting a hit list of topics for standardization, ready for the PAR submission at Tahoe.

If you are interested in more information on the group why not contact the group chair Helmut Roth, *<hroth@relay.nswc.navy.mi.>?*

An Update on UNIX-related Standards Activities [†]

by Nicholas M. Stoughton
USENIX Standards Report Editor
<nick@usenix.org>

Report on the IEEE Standards Board

Mary-Lynne Nielsen <m.nielsen@ieee.org> reports on the September 1993 meeting:

This was a very busy meeting, with a lot of major actions for the Portable Applications Standards Committee (PASC, the sponsor of, among other things, the POSIX projects): document approvals, PAR (Project Authorization Request) approvals, new review policies, and a whole new numbering scheme for the POSIX documents.

More on renumbering later, but let's get the easy stuff over with first.

SPAsystem Update

Work on the SPAsystem proceeds apace, and the IEEE Standards Board is preparing to create an Industry Advisory Group (IAG) to assist and advise in this process. Preliminary interviews have been conducted with potential candidates, and the Board is looking at creating this committee in 1994. Just as a reminder, the SPAsystem will be the springboard for all the future electronic development and delivery of standards products. The first part of this, the IEEE bulletin board, is already running and accessible via modem; Internet access is the next big step for this project. In addition, the IEEE is working on developing standards on-line in tandem with moving towards developing standards in SGML, the Standard Generalized Markup Language. This step is in research, with an aim to mount pilot projects in 1994. Previous snitches have gone into some detail concerning the development of this project.

Metric, Metric Everywhere

Over the past year, all the Boards of the IEEE received presentations from members of the IEEE Metric Policy committee concerning adoption of a broad-based Institute policy on the use of the metric system and SI units. This policy was approved by the Standards Board and was voted on by the IEEE Board of Directors (the governing Board of the entire Institute) in November. At that time, this policy was approved. It basically states that the IEEE shall actively support the use of the International System of Units (SI), both by educating its members as to its use and through active implementation of the system in its practice. For standards, this means that SI units are the preferred method for unit symbols. While this doesn't begin to resolve the issue of whether a kilobyte is 1000 bytes or 1024 bytes, it is a step in the right direction.

P1003.4

One of the more eagerly awaited standards, IEEE P1003.4, was approved at this meeting by the Review Standards Committee (RevCom). It will be published in the spring of 1994 as IEEE Std 1003.1b. Don't worry about the numbering, I'll explain that later.

NesCom Actions

NesCom, the New Standards Committee, discussed at length the need for good cross-communication between itself and Accredited Standards Committee (ASC) X3, an independent standards group that also covers the field of information technology, but whose membership is company-based rather than individual-based. There have been problems with overlapping scopes of projects in these two groups in the past, particularly when projects are proposed to the American National Standards Institute (ANSI) Board of Standards Review (BSR), which is the US coordinating body for project approvals. In an effort to educate and minimize overlap, NesCom moved to try and avoid those problems at the ANSI level by having NesCom review ASC X3 new project proposals (which are called SD3s) and include X3 for information on all NesCom mailings, including PAR distribution.

A great deal of activity occurred in relation to PASC PARs. Five new PARs were approved. Two of these were a revision and splitting of a previous PAR (IEEE P1003.6) to show the security revisions to IEEE P1003.1 and IEEE P1003.2. One PAR was for a revision to IEEE P1003.5, and one was a new PAR. Also, the PAR for the language-independent (LI) version of IEEE P1003.1 was revised and renumbered.

In addition, three PARs were withdrawn at this meeting. One, IEEE P1003.19, was withdrawn because the working group was chartered to develop Fortran 90 bindings to POSIX.1, and that field is not mature enough to encourage standardization. Two others, IEEE P1003.16 and IEEE P1003.16a, were withdrawn due to the death of

[†] This is a re-print from ;login, the USENIX Association Newsletter, Volume 19 Number 2

the LIS requirement in PASC and hence the removal of the need to have a separate C language binding to POSIX.1. Details of these PARs are given below.

Numbers? What about numbers?

All right, take a deep breath! Here we go!

Yes, many of the POSIX projects have been renumbered by NesCom. Before I get into the whys and wherefores, let's try to understand some of the history and get the details over with.

The POSIX series of standards, referred to by their base number of IEEE P1003, have been under development for a number of years. Originally, the project was relatively small in size; a few key standards existed, along with some adjunct standards projects in development. However, over the years, the growth of work this area has been exponential. There are now over 30 projects that have been or are being developed that fall under the banner of "POSIX."

The numbering for these standards has been varied, in part to show their relationships to base standards, to the working groups developing those projects, or to other projects or supplements in the series. In other words, we've had all kinds of different numbering for POSIX. Both base standards and amendments have been numbered 1003.x; amendments have been numbered both 1003.x and 1003.n (for an alphabetic designator); and projects have been numbered with a "double-dot" number to show their relationship to one of these "base" single-dot numbers (such as IEEE P1003.7.1).

In addition, the POSIX series of standards have been adopted or proposed for adoption as international standards. The international committee in charge of overseeing the POSIX international development, ISO/IEC JTC1 WG15, recommended a certain structure for this proposed work. That structure consisted of one number for the whole series, 9945. (This number parallels the IEEE equivalent of 1003.) However, ISO/IEC proposed that only three parts exist in 9945. All other projects would have to fit into that three-part plan as supplements to those "base" standards.

Remember, at this time, there were already about 10 PARs in existence in PASC. So when ISO/IEC decided that there would only be three parts, the system was already broken. What did have to change, however, was the content of the documents themselves; the IEEE POSIX projects had to make themselves fit into this structure by mapping themselves, section by section, to the content of the "base" standard they were amending. This process has been a long, arduous, and ongoing one in PASC, as many standards needed to be entirely restructured to show their relationship to what were now the three base standards. However, the original numbers assigned to these projects by the IEEE had never been changed to reflect these evolving relationships. People learned that IEEE P1003.4 was going to be part of IEEE P1003.1 and left it at that. All that has begun to be adjusted by action of the NesCom at this meeting.

So what prompted any change? Basically, the PARs to revise IEEE P1003.6. These new PARs split the old POSIX.6 work into two parts, one that amended IEEE P1003.1 and one that amended IEEE P1003.2. The proposed PARs used the double-dot numbering scheme (P1003.1.6 and P1003.2.6). The chair of NesCom would not accept these proposed numbers because they didn't meet the current numbering system used by IEEE standards. (When this numbering system was approved, POSIX was considered and recognized as an exception to it.) This led to a detailed discussion of the overall POSIX project numbering, the end result of which was a conclusion from NesCom that the existing POSIX numbering scheme needed to be cleaned up and corrected.

Most of the discussions concerning this occurred once the NesCom agenda (with PARs) was received by the committee for their 40-day review period prior to the meeting. Thus, Jim Isaak and I (Jim is also a member of NesCom as well as PASC chair) were able to discuss this at great length with the committee members. Despite giving many reasons for why the old numbers should be kept, NesCom remained adamant about bringing these PARs in line. This then left the problem of creating another layer of broken numbers on top of an already broken numbering system. Finally, it was proposed that all of the current POSIX projects not yet published be renumbered to match the existing standards numbering scheme. Period. In toto.

This task was partially accomplished at the meetings of NesCom; the easiest projects to renumber were handled. There are some projects in which further clarification from the committee is needed prior to renumbering, and these problems will be resolved for the December meeting. For instance, it was unclear whether 1003.7 should keep a 1003 number because of a proposal to change its ISO/IEC number from 9945-3 to an entirely new number apart from the 9945 series So the numbering on that will be revised later. In addition, there are some projects that have not yet been renumbered because their early stage of development does not yet clearly pinpoint to which base standard they will belong.

In other words, most of the POSIX family of standards, and all of the mature projects, have been or are going to be renumbered to fit into the current acceptable numbering scheme of NesCom. In some cases, this means no change at all to the PAR. In others, it means a major change.

Sure, this can and will be confusing. No question about it. But NesCom forced the issue, and it had to be dealt with. I suppose you could think that it should have been dealt with all those years ago when the ISO/IEC and PASC numbering schemes diverged, if you need a consoling thought. And there are a couple of advantages to these numbering changes. One is that all the POSIX numbers will be in one numbering scheme, and this numbering system clearly shows which documents are meant to be supplements to a base standard and which are base documents themselves. Previously, the numbers did not express this relationship.

One comment – why did this happen now, at this stage in development? You have to keep in mind that all committees of the IEEE Standards Board, including the Board itself, are as dynamic as the working groups developing standards. The members change every year; some stay on for years, some stay for only one year. The tone and action of each committee are a direct reflection of the current membership. Because there was a new vice-president of the IEEE Standards Board this year (for the first time in three years), NesCom gained a new chair and a lot of different members. Basically, that's why the issue came to a head; it was perceived as a problem by the current membership and the chair.

It is expected that all PASC working groups will start using the new numbers, probably in conjunction with their old numbers, immediately. (In large part, the use of both numbers will occur to address the expected confusion.) Over time, the old numbers will be phased out and the new numbers used exclusively.

Here is a list of the current changes in IEEE POSIX numbering, along with the numbers that don't change because they don't fold into P1003.1 or P1003.2. There will be more to come in December.

Old number	New number	Description
P1003.0	Same	Guide to POSIX OSE
P1003.1	Same-Published	System Interface
P1003.1a	Same	System Interface (revision)
P1003/LIS	1372	Language Independent Specification
P1003.2	Same-Published	Shell and Utilities
P1003.2a	Same-Published	
P1003.2b	Same	
P1003.3	P2003-Published as IEEE Std 1003.3-1991	Test Methods
P1003.3.1	P2003.1-Published	
P1003.3.2	P2003.2	
P1003.4	P1003.1b-to be published	Realtime Extensions
P1003.4a	1003.1c	Threads Interface
P1003.4b	P1003.1d	
P1003.5	Same-Published	Ada Bindings
P1003.1.6	P1003.1e	Security extensions – System Interface
P1003.2.6	P1003.2c	Security extensions – Shell and Utilities
P1003.8	P1003.1f	Transparent File Access
P1003.9	Same-Published	FORTRAN Bindings
P1003.10	Same	Supercomputing Profile
P1003.11	Withdrawn	Transaction Processing Profile
P1003.12	P1003.1g	Protocol Independent Network Interfaces
P1003.13	Same	Realtime Profile
P1003.14	Same	Multiprocessor Profile
P1003.15a	P1003.2d	Batch Extensions for Supercomputing
P1003.16	Withdrawn	C Language Binding to POSIX.1LIS
P1003.16a	Withdrawn	
P1003.17	P1224.2-Published	X.500 APIs
P1003.18	Same	Platform Environment Profile
P1003.19	Withdrawn	Fortran-90 Bindings
P1003.21	Same; to be revised	Realtime Distributed IPC
P1003.22	Same; to be revised	

New PARs

P1003.7.3 Standard for Information Technology-Portable Operating System Interface (POSIX)-Part 3: System Administration-Amendment: User Administration
NOTE: This approval was contingent upon the Sponsor Chair returning to NesCom in December with an "appropriate numbering system."

Revised PARs

P1003.1.6 Standard for Information Technology-Portable Operating System Interface (POSIX)-Part 1: System Application Program Interface (API)-Amendment n: Protection, Audit, and Control Interfaces [C Language]
NOTE: Renumbered to P1003.1e.

P1003.2.6 Standard for Information Technology-Portable Operating System Interface (POSIX)-Part 2: Shell and Utilities- Amendment n: Protection and Control Utilities
NOTE: Renumbered to P1003.2c.

P1372 Standard for Information Technology-Portable Operating System Interface (POSIX)-Part 1: System Application Program Interface (API)-[Language Independent]
NOTE: This PAR revised the PAR for P1003.1/LIS.

PARs for Revision of Existing Standards

P1003.5 Standard for Information Technology-POSIX Ada Language Interfaces-Part 1: Binding for System Application Program Interface (API)

Withdrawn PARs

P1003.16 Standard for Information Technology-POSIX C Language Interfaces-Part 1: Binding for System Application Program Interface (API)

P1003.16a Standard for Information Technology-POSIX C Language Interfaces-Part 1: Binding for System Application Program Interface (API)-Amendment 1: System API Extensions

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

Some Humor

I recently received the following from Dr. Kerry Raymond at the University of Queensland, and it may amuse those of you who have ever sat on a standards committee:

I'm On a Committee

Oh, give me your pity, I'm on a committee
Which means that from morning to night,
We attend and amend and contend and defend
Without a conclusion in sight.

We confer and concur, we defer and demur
And re-iterate all of our thoughts.
We revise the agenda with frequent addenda
And consider a load of reports.

We compose and propose, we suppose and oppose
And the points of procedure are fun!
But though various notions are brought up as motions
There's terribly little gets done.

We resolve and absolve, but never dissolve
Since it's out of the question for us.
What a shattering pity to end our committee
Where else could we make such a fuss?

Report on POSIX.0: Guide to POSIX OSE

Kevin Lewis <klewis@gucci.enet.dec.com> reports on the January 10-14, 1994 meeting in Irvine, CA:

From The Battlefield

Work to get POSIX.0 approved continues. The ballot recirculation is in. Let me first give you the statistics. There are 86 total people in the balloting group of which 81 are eligible to vote. A total of 75 ballots were returned. The breakdown of those votes are as follows: 45 affirmative, 17 negative, 13 abstentions. This represents a 92% return. The 45 votes represent a 72% affirmative. The ballots consist of 167 comments and 182 objections, which represent about 25% of the total submitted during the first round of balloting.

Before commencing resolution of the recirculation ballots, in Irvine, we discussed a letter that had been received by the IEEE from the ACM. This letter focused on the overall IEEE balloting process, the concern that some of IEEE work overlaps with standards work within X3, and that our guide document still lacks the necessary level of consensus. A portion of our rationale for rejecting specific objections was inadequate.

This letter amounted to a hand grenade lobbed into the middle of our work, or, to quote working group members, "a tactical nuclear attack."
I won't go into fine detail here, but the group did meet with two people, who were wearing ACM hats, to share their concerns along with those of ACM. However, some working group members who were also ACM members were quite disturbed by the tone of the letter, part of which included an *'ad hominem'* attack against the working group itself. They were also distressed by the

approach taken by ACM of sending such a letter to the IEEE without first having a dialogue with the group. Words such as 'protest' and 'malfeasance' made their way into the discussion.

In my humble opinion, the only part of the letter I considered valid (and I'm quite sure I would have the unanimous assent of the group on this) was that portion addressing our rationale. This, by the way, was quite helpful.It redirected our efforts for the better during the week. The group decided to return to the unresolved objections from the first ballot for the purpose of reviewing each one for possible acceptance or correcting/strengthening our rationale for rejection, which I admit was both weak in places and occasionally arrogant.

We completed this task, but did not get to the recirculation ballots. Because of this, and also due to the overall feeling in the group that more productive resolution work could be done at a meeting away from the quarterly PASC (Portable Applications Standards Committee) meetings, we agreed to schedule an additional meeting to take place in March in the Washington, D.C. area, specifically for the section leaders. The only activity at this meeting will be resolution of the recirculation ballots. The exact date has yet to be determined.

I feel quite strongly that we will be able to complete all of the recirculation ballots at this March meeting. What remains now is the review-and-comment action by SC22, the ISO subcommittee responsible for POSIX, which is now in progress. It looks like it will be October before we have a document ready for submission to the IEEE Standards Board.

One more thing: the POSIX.0 working group is scheduled to meet for two days at the April PASC meeting in Lake Tahoe. This will be a skeleton crew to effect coordination with and provide representation to some other key PASC committees, such as the Profile Steering Committee and the Sponsor Executive Committee. In addition, this crew will monitor the resolutions to the international committees that directly or indirectly affect the guide effort.

Report on POSIX.22: Computer Security Framework

Randall Wayne Simons <rsimons@somnet.sandia.gov> reports on the January 10-14, 1994 meeting in Irvine, CA:

The POSIX.22 committee is defining a framework for distributed computer security. The framework will be a common reference model to guide members of other POSIX committees in addressing security needs in the standards they are defining.

At this first POSIX meeting I have attended, my main impression was of heads silently bowed over clacking keyboards as multiple laptops were simultaneously applied to modifying a document. David Rogers, chair of the committee, brought a troff version of the X/Open Snapshot called the "Distributed Security Framework," POSIX.22 wants to keep the X/Open and POSIX documents in sync since both groups are working on the same problem. The most recent version of the document had just been reviewed by X/Open, and there were numerous suggestions for improvement, including many that required some restructuring of the document. POSIX.22 took on this task, and simultaneously reviewed and added their own improvements. Different sections of the document were distributed to each committee member who then did the cutting, pasting, and merging.

The reorganized document begins by introducing top-level information system security concepts, terms and models. There is a description of threats, most of which were moved to an appendix. More detailed models define security architectures and characteristics of interfaces to security services. Finally, the individual services and interfaces are modeled and described in detail. Interfaces support both management and operational functions for each of the services.

The basic services included are: authentication, access control, security audit and cryptographic services. At a higher level, domain interaction services, which combine various basic services in a distributed environment, include user authentication and secure association service.

After more review and revision by both X/Open and POSIX.22, the Framework document will be ready for balloting around July. The balloting group should meet in April, so be watching for it. POSIX.22 had seven people attend this meeting. There was plenty of work to go around. Anyone willing to help develop the POSIX Computer Security Framework will be more than welcome at future meetings. There is much to be done in security for POSIX – see the report from POSIX.6.

Report on POSIX Test Methods

Fred Zlotnick <fred@mindcraft.com> reports on the January 10-14, 1994 meeting in Irvine, CA:

The requirement that POSIX working groups develop test methods in parallel with their standards was suspended at the April 1993 meeting, and then finally withdrawn at the following July meeting. Nevertheless there are two active test methods activities and more in the works. The working groups, which met at the October POSIX meeting in Bethesda and at the January meeting in Irvine, are group 2003 (which is revising POSIX.3, the standard that describes how to write test methods) and group 2003.2 (which is developing test methods for POSIX.2, Shell and Utilities). Technically it wasn't the 2003.2 working group that met, but more about that later. Both of these groups are chaired by Lowell Johnson of Unisys.

Revision to POSIX.3

Working group 2003 has been writing a revision to POSIX.3 for about a year and a half. Although POSIX.3 has been used successfully to write test methods for POSIX.1, and its methodology has formed the basis for quite a few commercial test suites, the use of this methodology has revealed a number of problems. The purpose of this revision is to deal with these problems:

- It is difficult to use POSIX.3 to write test methods for standards that modify other standards. Real-time (POSIX.1b, which used to be POSIX.4) is a good example. Because the real-time standard consists of a collection of optional extensions to POSIX.1, every assertion for real-time must be conditional (type C or type D). But there are other conditions within many real-time assertions, and this makes the statement of each assertion in POSIX.3 format rather cumbersome. Moreover, some of the options of POSIX.1b place additional semantic requirements on POSIX.1 interfaces such as open( ). Writing the assertions to test these requirements raises questions not adequately addressed in POSIX.3-1991: How should they be numbered? How should they be conditioned? How should they be classified (assertion-typed)?
- A number of the users of POSIX.3 have found the standard difficult to understand. A number of related but distinct concepts in POSIX.3 have been confused by users of the standard.
- ISO had difficulty with the terminology of POSIX.3, which is not always consistent with that of other test methods standards at the international level.
- POSIX.3 was originally designed for and is specified as only applicable to POSIX standards. The IEEE's Portable Applications Standards Committee (PASC) currently manages a number of projects, some of which fall under the POSIX umbrella. Yet the test methods methodology of POSIX.3 applies to, and should be specified for, other PASC working groups, such as P1327 and P1328. In general, the scope of POSIX.3 should be broadened.

Working group 2003 began the week in Irvine with Draft 2.0 of the revised standard. This draft had been completed by the group's technical editor, Anthony Cincotta of NIST, just prior to the meeting. By the end of the week the group had agreed on a set of changes that, when completed, will produce Draft 3. This draft should be suitable for mock–ballot.

The basic methodology of assertion-based testing has not changed in this revision, but the form in which assertions are written has changed drastically. The familiar, frequently misunderstood, and often vilified 2x2 matrix of assertion types is gone. The syntax of an assertion now closely resembles a conditional sentence, with possibly many nested conditions. If an assertion applies only when conformance to a particular version of a standard (e.g., POSIX.1–1993) is being tested, a condition indicates this fact. If an assertion depends on support of an option (e.g., job control) a condition indicates this fact. Sometimes an assertion may specify required behavior but may only be testable if the implementation supports optional features (such as certain appropriate privileges). If so, a condition indicates this fact. Assertions are now labeled by assertion IDs rather than assertion numbers; an assertion ID is a string.

The new assertion structure promises to make assertion writing easier and to allow the structure of test methods standards to more closely parallel the base standards against which they are written.

POSIX.2 Test Methods

For the last four meetings, the 2003.2 working group has been laboring over the ballot resolution for the ballot of Draft 8. According to IEEE rules, this means that it is not really the 2003.2 committee that is meeting. Ballot resolution is done not by the committee that wrote the draft, but by a group of technical reviewers. They just happen (in this case, as in most) to be many of the same people.

Ballot resolution for Draft 8 has been a slow job for a number of reasons. One is the size of the task: there are almost 9000 assertions in Draft 8. Another is that despite its size, Draft 8 is incomplete, and a number of ballot objections make note of this fact. Some of the missing pieces (like assertions for yacc) have not been easy to supply. Nevertheless, part of the task of resolving these objections will be to fill in those missing pieces. Another problem is that participation in this working group has not been as consistent as one might like, although the October and January meetings were well-attended.

In addition to the incompleteness of Draft 8, major ballot issues include the fact that the test methods must be locale-dependent but the draft frequently addresses testing only in the POSIX locale. Moreover, Draft 8 is not explicit about this fact. Other problems include the omission of reasons for classifying assertions as extended, and the omission of clear references for reference assertions.

By the end of the October meeting, reviewers had made enough progress to enable the technical editor, Andrew Twigger of UniSoft, to produce interim Draft 8.5. This will not be balloted, but it has been useful as a working document at the January meeting. At that meeting the technical reviewers completed ballot resolution. The technical editor now has to integrate the resulting changes to produce Draft 9, which will go out to ballot.

Test Methods for POSIX.1b

At the Irvine meeting, the PASC Sponsor Executive Committee (SEC) approved a new Project Authorization Request (PAR) for test methods. The PAR creates a POSIX 2003.1b project under the direction of the 2003 working group. Its goal is to write test methods for POSIX.1b.

You may recall that during the "test methods wars" the POSIX.4 working group was grandfathered out of the requirement (since lifted) to develop test methods along with base standards. Thus there are no test methods, even in draft form, for POSIX.1b. Yet there is substantial interest in the development of conformance tests for POSIX Real Time, and such tests need a specification. In Irvine, a number of organizations, including representatives of several DoD agencies (DISA, JITC), were committed to provide resources to develop these test methods. Ken Thomas of JITC has offered to serve as vice-chair of the 2003 working group for the direction of the 2003.1b effort. Bruce Weiner of Mindcraft has offered to act as technical editor for the test methods document.

Report on POSIX.5: ADA Bindings

Delbert L. Swanson <DSWANSON@mhs.sp.paramax.com> reports on the January 10-14, 1994 meeting in Irvine, CA:

The primary charter of the POSIX.5 group is to produce Ada language bindings to POSIX standards. The Ada binding for POSIX.1, POSIX.5 has been published as an IEEE standard, and we are now preparing bindings to the Real-Time Extensions standards being developed by the POSIX.4 group. These bindings have been designated as POSIX.20.

Draft 2 was developed as a "thin" binding to the Real-Time Extensions. That is, it merely made a correlation between the constructs defined in the C version and the constructs in the Ada version. None of the explanations or semantics are repeated. This was done following what was the policy of IEEE and the ISO community – that all language bindings would be thin bindings of a normative language independent specification (LIS) of the standard. Actually, our approach was a compromise even then, since there was not yet a completed LIS version.

Circulated for a first ballot last summer, this draft was updated to account for comments and objections. In the meantime, the policy on thin bindings to LIS versions of standards changed, and so the group has been revising the document.

The next draft will be a thick binding – a complete specification of the interfaces for Ada applications. The advantage is that users will not need to refer to multiple documents (Ada and C) to understand the behavior of the Ada interfaces. The disadvantage is in the maintenance mode: if the baseline document changes, the binding document needs to change correspondingly. Moreover, it takes more work to produce a thick binding than a thin one.

We expect to work on more issues between meetings, and then polish the draft up to be ready for another full ballot after the April, 1994 meeting.

In January, the group re-examined our approach to Ada bindings to the threads extensions. We have concluded that almost all of the functions offered in POSIX.4a are going to be provided for Ada applications in the revision of the language commonly called Ada9X, which is expected to be granted standard status within the year. It seems beside the point for us to duplicate, as OS functions, these capabilities, which will soon be available as language constructs. A couple of remaining pieces will be incorporated into the new draft of POSIX.20.

The status of our coordination ballot on POSIX.4a, the threads extensions is an item of some concern to the group. All but a couple of our objections were resolved in discussion. Unfortunately, the objection that we consider most important has been rejected on the grounds that it would reduce consensus. It is our view, particularly when handling signals, that it is important to be able to mask asynchronous signals for the entire process. This is important in Ada runtime environments, and it will also be important within C programs. The current C interface includes only per-thread signal masks. It is uncertain what the resolution of this issue will be.

Meanwhile, we are preparing a revision to POSIX.5, to correct errors found in the standard by implementers (missing parameter, missing function definition, error condition oversights). The only way to make "substantive" changes, even for errors, is to revise the standard, which means balloting, etc. Revisions should be ready for ballot as soon as the administrative details are taken care of.

Report on POSIX.6: Security Extensions

Lynne Ambuel <ambuel@dockmaster.mcsc.mil> reports on the January 10-14, 1994 meeting in Irvine, CA:

Introduction

As a first time snitch, I would like to divulge my thoughts on standards – from a security geek's point of view. Subjects include the peculiarities of information security and those who live by it, the activities taking place, and the status of the POSIX security working group (previously known as P1300.6). Other issues may creep into the discussion, but everything will relate (no matter how obscurely) to these greater issues.

A Different Animal

Computer Security specialists are used to being called names like"different," "special," and even "strange." Although some might take offense, I must agree with the characterization. Computer security really is a different animal. While most software designers and developers can kick back once their code does what it is supposed to do, we have just started – the important part is what the code also does *not* do. For other applications, added functionality brings cheers from users – more bells and whistles are always better. We add functionality and our users cringe – more restrictions. If we are good, no one will notice we have added more, while our counterparts fly banners with their latest new features. Is it any wonder we get no respect?

In the standards world, we are treated in a similar fashion. We in the POSIX Security Working Group (P13.6) have the unenviable job of policing the work of other POSIX groups, to be sure that gaping security holes are not mandated in the standards. That makes us many friends. We add interfaces that have sweeping effects on well-established sets of interfaces. We change those pillars of POSIX interfaces and utilities to accommodate our added features. Our job never ends. As new standards are developed we continue to study them for the impact on the security of POSIX–conformant systems. We have just started studying what security means when systems are interconnected. The concepts of user identification and authentication and data markings become remarkably complex once they are taken out of a single system and spread throughout a network. We have a lot of work to do in getting standards that meet the needs of the market and protect the information of those using the end product.

The Great Thing About Standards is There Are So Many To Choose From

Not so many years ago computer standardization was a foreign and even ridiculous thought. In the eighties, however, we started moving toward a more friendly world in which everyone wanted to talk in the same language. Organizations that previously held design and implementation information excruciatingly close soon started sharing their gems freely. Security joined right in: standards were written for what security should be, first in individual countries and then in international cooperation. The utopian view was that someday (soon) there would be a single security standard for the globe. Working groups were formed to look at standardization of security interfaces, utilities, and data. But with limited coordination among groups and with the current problem of downsizing, organizations now send fewer representatives. Each group lacks the resources to make progress on their standards. Substantive progress might be made by pooling resources, and there would be one accepted standard instead of a handful of incomplete ones.

Progress of POSIX Security Working Group

Now I can tell you what we have accomplished. A third ballot of the five initial security options for POSIX.1 (access control lists, mandatory access control labels, information labels, audit and fine-grained privileges) is being distributed as you read this. However, it is about four months behind schedule due to loss of half of the ballot-resolution team. In addition, we have identified several interface areas that we need to tackle in order to complete a set of security interfaces for portable applications (identification and authentication, administration and portable formats of security attributes, cryptography, and network security interfaces). We have been unable to make headway in these new areas because we cannot get enough organizations to submit proposals, nor can we reach enough people willing to do the work.

What's a chair to do? Flood the Internet with calls for participation and proposals? Done it. Personal appeals to ex-members? Done it. Complain and wallow in self-pity? Done it. Get mad and stomp around some Marriott? Done it. Ignore the problem and act like fifty new attendees will show up? Done it. Continue the work and make progress, no matter how slow? Doing it, for as long as it takes.

Standards Update, What Next?

by Nicholas M. Stoughton
<nick@usenix.org>

Steady-State Behavior

Three years ago, attendance at POSIX meetings was around 340 people. At Irvine this January, the number had fallen to 165, and it is expected (hoped?) that it will bottom out at around 150.

So what has happened? Where have all the contributors gone? There is never a simple answer to questions like these, and there are at least two major influences at work in POSIX attendance. The first is straight economics. The world is in a recession. If your company is losing millions or even billions of dollars each year, is sending people to POSIX meetings the best way of spending what money it does have? Why not work at a distance, through the ballot process?

Another key factor in attendance is the type of work that is required. Three years ago, new standards development was in full swing. Now we are settling down to a steady state. Approved standards for POSIX.1, POSIX.2, POSIX.3, POSIX.4 (aka POSIX.1b), POSIX.5 and POSIX.9 are all there. Maintenance work on these is now a major focus, and that takes a different type of person on the working group. At Irvine, a considerable amount of time has been spent dealing with interpretation requests, chiefly for the most blatantly UNIX standards, POSIX.1 and POSIX.2.

Do we, the UNIX user community, care? Let's give up going to these meetings... after all, someone else is sure to do the right thing, aren't they? It can't be all that hard to maintain a standard! Anyway, we don't want standards rammed down our throats at every opportunity.

Unfortunately, the serpent whose name is Invention is lying there, coiled and ready to strike the moment that someone stops saying, "But where's the existing practice?" I've lost track of how many times Jeff Haemer and I have trotted out that phrase. In the standards world, practice really does make perfect. If standards are to be rammed down our throats, let them at least be palatable ones.

The rules of interpretation say that the approved standard should be interpreted as loosely as possible. If it is actually wrong, it is up to a later revision to fix the wording, but no one can complain in the interim. This will of course lead to lots of "Weirdnix" implementations: systems that claim POSIX conformance, but are about as far removed from UNIX as you can imagine!

So it is necessary to keep a significant presence within POSIX, attempt to restrain and guide the work occurring. Existing documents need revision to clarify the wording and to help prevent the worst excesses of Weirdnix. New POSIX projects are still being introduced, but at a much–reduced level. Many of these are not necessarily "mainstream UNIX" things either — an ADA interface to time synchronisation, Test Methods for POSIX.1b, the Realtime standard (yes, the number has changed, it used to be known as POSIX.4), and so on. Nevertheless, new direct UNIX projects are not unthinkable, and we must be ready to meet these challenges.

To give you a flavor of what is happening in the interpretations process, Andrew Josey, the Vice Chair of Interpretations has put together a chart–see next page.

In the past months new draft guidelines for PASC interpretations have been circulated and a BOF session met at Irvine to discuss them further.

The guidelines attempt to address the issues of timely response, and the issues of the scope of the interpretations. They are being followed now and we hope to see improvement in the process over the next six months.

What Do You Care About Standards?

I would like to take this opportunity to solicit your opinions. What do you think should appear in this column? I was recently invited to submit a series of articles to a prominent Open System Magazine. After I sent in material, I was told, "These are far too POSIX–centric. What about some other standards?"

There are several other areas that might be useful to report on, both in the *de facto* and *de jure* worlds. But rather than trying to read your minds, I'll solicit your suggestions. What else would you like to hear about?

Working Group	Mailing List	Current Position	Outstanding Requests
P1003.1	intrp1003.1@stdsbbs.ieee.org	61 request pre-Irvine, 31 addressed & in progress	30
P2003.1	intrp1003.1@stdsbbs.ieee.org	17 request, 5 complete	12
P1003.2	intrp1003.2@stdsbbs.ieee.org	30 requests pre-Irvine	0
P1003.5	posix-ada-interps@spectre.mitre.org	9 requests being processed	1
P1224	intrp1224@stdsbbs.ieee.org	6 requests have been answered	0
P1224.1	intrp1224.1@stdsbbs.ieee.org	2 requests have been completed	0
P1224.2	intrp1224.2@stdsbbs.ieee.org	3 requests being processed	0
P1327	intrp1224@stdsbbs.ieee.org	3 requests have been answered	0

Thank You

I would like to take this opportunity to publicly thank Michael Hannah, a regular contributor to this column over the years, for all his work with POSIX.9, and for his wit, and his enthusiasm. Michael has been promoted to a new position within Sandia National Labs, running a 2000-node Intel Paragon system. I know he has some stories that would interest SAGE members, although he will no longer be in a position to continue his work with POSIX. The first article I edited for this column was from Michael, and the ease with which I was able to work with him persuaded me to take on the job permanently. I am sure you will all join me in wishing him every success in the future.

Unix Tricks & Traps

PCs are out to get me. (Yes, I often use a PC, at least on customer sites). If they aren't hanging or crashing, they're attacking my precarious sanity in more subtle ways. For example, several of my customers run TCP/IP on their PCs. The other day, I tried to telnet from my PC to an important Unix server — and got connected instead to an insignificant Vax.

Convinced that no-one could have installed VMS on the Unix server without me noticing, I did some quick checking — and found that someone had started up a TCP/IP stack on the Vax in question. Unfortunately, they'd picked an IP address at random — the same address as the Unix server!

Just how do we identify duplicate IP addresses on the network? (By the way, I just love the message "Duplicate IP address!!! From <MAC address>". On a several hundred host network, a MAC address alone is *not* very useful. Why doesn't the message include the IP address as well?).

I wrote a small script that checks whether a particular IP address has been duplicated on the network. Before we look at it, let's discuss some networking and TCP/IP theory.

When you attempt to connect (say via `telnet`) to another host, your machine first determines that machine's IP address. However, to physically get a TCP/IP packet to that machine (assuming we are talking an Ethernet LAN), it needs to know its "physical" Ethernet address (also known as the "MAC" or Media Access Control address).

To find out, your machine broadcasts an ARP (Address Resolution Protocol) packet. This packet simply says "Who does this IP address belong to?". The machine with that IP address sees the ARP request and responds with its MAC address.

One gotcha is that if more than one response is received to the ARP request, the *last* one is used for some inexplicable reason. So when my PC did an ARP request for the IP address of the Unix server, the Vax was of course much slower at responding, so its MAC address was the one that was used.

We can use this knowledge to detect duplicate IP addresses. Simply reference an IP address (which will cause an ARP request), and keep track of the MAC address(es) that are returned. If the same MAC address is returned each time, it is unlikely that the IP address is duplicated. If more than one MAC address is returned, however, then a duplicate IP address has been allocated.

However, since MAC addresses are cached for a few minutes after an ARP request, we need to explicitly delete the ARP table entry for the IP address before we reference it, forcing a fresh ARP request broadcast.

A script that implements this check follows.

```
#
#   unip
#
#   Checks to see if an IP address has been duplicated by
#   making repeated ARP requests and counting the number of
#   Ethernet addresses that are returned.
#
#   March 11, 1994      Adrian Booth
#

USAGE="'basename $0' <IP addr> [count]"

if [ ! "$1" ]; then
    echo $USAGE >&2
    exit 1
fi

IPADDR=$1
```

```
if [ "$2" ]; then
    COUNT=$2
else
    COUNT=10
fi

while [ "$COUNT" -gt 0 ]; do
    arp -d $IPADDR >/dev/null 2>&1      # flush the ARP table entry
    ping $IPADDR >/dev/null 2>&1        # reference the IP address
    arp $IPADDR                         # output the new table entry
    COUNT='expr $COUNT - 1'
done | nawk '{ count[$NF] += 1 }
END {
    for (i in count) {
        printf "Ethernet address %s occurs %d times\n", i, count[i]
    }
}'
```

Adrian Booth, Adrian Booth Computing Consultants <abcc@dialix.oz.au>, (09) 354 4936

Please send your contributions for this column to Janet Jackson <jackson@cwr.uwa.edu.au>.

AUUG MANAGEMENT COMMITTEE
SUMMARY OF MINUTES OF MEETING 11th February 1994

Present: Phil McCrea, Chris Maltby, Glenn Huxtable, Frank Crawford, Michael Paddon, Rick Stevenson, Stephen Boucher, Peter Wishart, Greg Birnie.

Apologies: None. Guests: Liz Fraumann, Catrina Dwyer, Ian Hoyle.

1. President's Report

Favourable comments had been reported about the Australian articles. We have also been approached by Financial Review to do articles for them. We need to carefully consider the effort required to support this given our experiences with the Australian articles. "Chips and Bytes" in Darwin was also interested. This activity was producing a high profile for AUUG.

The committee gave its formal thanks to Liz Fraumann for her inspiration and ideas over the last two years.

2. Replacement Business Manager

Catrina Dwyer was considered by the committee for the Business Manager position and subsequently offered the position.

3. AUUG94 Report

Ian Hoyle (AUUG94 conference program chair) reported to the meeting. Overseas speakers positions were nearly filled:

Linux Torvalds (Linux)
Tom Kristensen (PERL)
Bob Glass (Sun - user interface) - funded by Sun.
Bill Plauger (C++, tools ...) - funded by Whitesmiths.
Chris Stone (OMG)
Dennis Ritchie (UNIX) - not confirmed.
Bell Cheswick (Security/AT&T/Firewalls) - not confirmed.
Mike Defasio (Novell)

The network is being organised by Hugh Irvine. It will be connected to AARNet via a T1 microwave to Melb Uni. A mini Interop style of event is being investigated.

It was noted that food is a major cost item at the conference (cost breakdown of conference was presented by EAF). It was noted that AUUG should make space available on its stand for other user groups.

4. Secretary's Report

There are quite a number of memberships which have not been renewed since the last expiry in Jan 1994. It runs to around 100 individuals and 50 institutions. However the Secretariat reports this figure is considerably down on previous years. All these members have been sent a second renewal notice just before this meeting.

5. AUUG FTP Archive

People at archie had approved of adding an AUUG disk to archie.au. It was suggested that we pay AARNet to buy the disk on our behalf. Estimated to be <$3K.

6. Proc. to Subscription Members

Some subscription members had requested copies of the AUUG proceedings as well as AUUGN.

Motion: That AUUG increase the subscription costs to $150 as at 1st July 1994 and provide subscription members with a copy of the AUUG proceedings. Moved: FC/GH. CARRIED.

7. Corporate Sponsors

PM mailed out 72 letters asking for corporate sponsors for AUUG. $5K - $10K. EAF/CD to supply article for AUUGN on corporate sponsorship.

It was suggested that we could have a sliding institutional fee. This would get larger groups to pay more, based on organisation size. This may be a´problem for a large organisation with a small interest in AUUG. Could also have a discount for membership of related organisations. Decided to delay further discussion until after corporate sponsorship drive.

8. Other Business

8.1. Calendar of Events

Should have something in AUUGN that includes AUUG, SUG-M, ACS, ... GB to keep in contact with organisations and chapters for this item.

8.2. X/Open

Request for involvement in their survey. Previous X/Open survey sent to AUUG institutional members and only got around 9 responses. It was noted that we were doing our own survey soon, we might see if we could incorporate X/Open into that, otherwise it may overload members. Resolved to get be involved in X/Open survey next year.

8.3. Chapter Payment for SA/NT

Motion: That chapter payments for SA and NT should be raised to 50% in line with other chapters not holding the national conference. Moved: FC/GH. CARRIED.

8.4. Aust. Articles

Currently have 2 articles in queue. Committee needs to provide more input, within next two months. Should also get chapters more involved. Could get Kirk to supply items. One article on summer conference series.

8.5. 008 Number

Use of AUUG 1-800 (free call) number would allow interstate members easier access to secretariat etc. Need to consider AUUG directory entry in each state. Need to get a single contact number. Number could be directed to ACMS now and moved later if needed.

Motion: That AUUG obtain a 1-800 (free call) number and direct it to the ACMS AUUG contact number. Moved: FC/GH CARRIED.

FC to organise with Telecom.

8.6. Constitutional Changes

There were some problems with the current constitution, e.g. membership periods. With another election coming up we should try to include a ballot for constitutional changes in it.

It was decided that CM would draft constitutional changes covering:
membership periods
position of immediate past president (replacing one committee)
restricting officer positions to non chapter office holders
ACTION: CM.

9. Next Meeting

The next meeting will be on Fri 22nd April 1994 at Softway in Sydney.

Peter Wishart
AUUG Inc - Secretary

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.

MEMBERSHIP APPLICATION

INSTITUTIONAL

To apply for AUUG membership, complete this form and return it with payment in Australian Dollars to:

REPLY PAID 66, AUUG MEMBERSHIP SECRETARY,
P.O. BOX 366, KENSINGTON, NSW 2033, AUSTRALIA
Tel: +61 2 361-5994 Fax: +61 2 332-4066

Tick this box if you wish your name withheld from mailing lists made available to vendors. □

NOTE: Please do not 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, and remember to select either surface or air mail.

We, ______________________________
(Company Name)

do hereby apply for:

Renewal/New* Inst. membership of AUUG	□ $350.00
International surface mail	□ $ 40.00
International air mail	□ $120.00
TOTAL REMITTED	AUD$______

(Cheque, money order, or credit card)

I/We agree that this membership will be subject to the rules and by-laws of AUUG as in force from time to time, and that this membership will run from time of joining/renewal until the end of the calendar or financial year.

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.

Signed____________________ Date ________

Title ____________________________

INSTITUTIONAL MEMBER DETAILS:
To be completed by institutional members only.

Following are our specified contacts. The primary contact holds the full member voting rights. The two designated reps will also be given membership rates to AUUG activities including chapter activities. By default a regional chapter will be selected for you. If you would rather nominate a chapter, please specify in space provided (indicate NONE for no chapter). *(Please print clearly or type)*

Primary Contact____________________

Position/Title ____________________

Address ____________________

____________________Postcode________

Bus. Tel: ______________ Bus. Fax:______________

e-mail address ____________________

Local Chapter Pref. ____________________

1st Rep.____________________

Position/Title ____________________

Address ____________________

____

Bus. Tel: ______________ Bus. Fax:______________

e-mail Address ____________________

Local Chapter Pref. ____________________

2nd Rep.____________________

Position/Title ____________________

Address ____________________

____

Bus. Tel: ______________ Bus. Fax:______________

e-mail address ____________________

Local Chapter Pref. ____________________

Please charge $__________ to my

□ Bankcard, □ Visa, □ Mastercard

Account number:____________________

Expiry date:____________________

Name on card:____________________

Signature:____________________

AUUG Secretariat Use

*Chq: bank*______________ *bsb* ______________
a/c ______________ *#* ______________
Date: ______________ *$* ______________
Initial: ______________
Date processed: ______________
Membership # ______________

AUUG Inc. as a user group, exists to provide UNIX® and open systems users with relevant and practical information, services, and education through cooperation among users.

MEMBERSHIP APPLICATION

INDIVIDUAL

To apply for AUUG membership, complete this form and return it with payment in Australian Dollars to:

REPLY PAID 66, AUUG MEMBERSHIP SECRETARY,
P.O. BOX 366, KENSINGTON, NSW 2033, AUSTRALIA
Tel: +61 2 361-5994 Fax: +61 2 332-4066

Tick this box if you wish your name withheld from mailing lists made available to vendors. □

NOTE: Please do not 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, and remember to select either surface or air mail.

INDIVIDUAL OR STUDENT MEMBERSHIP:

I, ______________________________ do hereby apply for:

Renewal/New membership of AUUG	□ $ 90.00
Renewal/New Student membership	□ $ 25.00 (please complete certification portion)
International surface mail	□ $ 20.00
International air mail	□ $ 60.00
UniForum affiliate membership	□ $ 20.00

TOTAL REMITTED: AUD$________ (Cheque, or money order, or credit card)

I agree that this membership will be subject to the rules and by-laws of AUUG as in force from time to time, and that this membership will run from time of joining/renewal until the end of the calendar or financial year.

____
Signature

____
Date

LOCAL CHAPTER DESIGNATE:

You can participate in the activities of a local AUUG Chapter. Part of your fee will be given to the chapter to support those activities. By default a regional chapter will be selected for you. If you would rather nominate a chapter, please specify here ________________ (indicate NONE for no chapter).

TO BETTER SERVE YOU, PLEASE PRINT YOUR CONTACT INFORMATION:

Name/Contact: ______________________________
Position/Title: ______________________________
Company: ______________________________
Address: ______________________________
______________________Postcode__________
Tel: BH____________________ AH ____________________
Fax: BH____________________ AH ____________________
email address:______________________________

Over for Institutional Membership

STUDENT MEMBER CERTIFICATION: *(to be completed by a member of the academic staff)*

I, ______________________ certify that
(administrator)
______________________ is a full time student at
(name)
______________________ and is expected to
(institution)
graduate approximately ________________.
(date)

____
Signature

____________________ ____________________
Title *Date*

Please charge $____________ to my

□ Bankcard, □ Visa, □ Mastercard

Account number:______________________________
Expiry date:______________________________
Name on card:______________________________
Signature:______________________________

AUUG Secretariat Use

*Chq: bank*________________ *bsb* ________________
a/c ________________ *#* ________________
Date: ________________ *$* ________________
Initial: ________________
Date processed: ________________
Membership # ________________

AUUG Inc. as a user group, exists to provide UNIX® and open systems users with relevant and practical information, services, and education through cooperation among users.

Notification of Change

You can help us! If you have changed your mailing address, phone, title, or any other contact information, please keep us updated. Complete the following information and either fax it to the AUUG Membership Secretary on (02) 361-5994) or post it to:

AUUG Membership Secretary
P.O. Box 366
Kensington, NSW 2033
Australia

(Please allow at least 4 weeks for the change of address to take effect..)

□ The following changes are for my personal details, member #:________________

□ The following changes are for our Institutional Member, primary contact.

□ The following changes are for our Institutional Member, representative 1.

□ The following changes are for our Institutional Member, representative 2.

PLEASE PRINT YOUR OLD CONTACT INFORMATION (OR ATTACH A MAILING LABEL):

Name/Contact: ________________
Position/Title: ________________
Company: ________________
Address: ________________
________________ Postcode________
Tel: BH ____________ AH ____________
Fax: BH ____________ AH ____________
email address: ________________

PLEASE PRINT YOUR NEW CONTACT INFORMATION:

Name/Contact: ________________
Position/Title: ________________
Company: ________________
Address: ________________
________________ Postcode________
Tel: BH ____________ AH ____________
Fax: BH ____________ AH ____________
email address: ________________

AUUG Secretariat Use

Date: ________________
Initial: ________________
Date processed: ________________
Membership # ________________

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
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.
- 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#* ________

Calendar of Events

Date	Org	Event	City
1994			
May 2-6		Networld + INTEROP '94	Las Vegas, NV
May 7-13	DECUS	DECUS - Summer '94	New Orleans, LA
May 18-21	UniForum NZ	UniForum NZ Conference	Rotorua, New Zealand
Jun 6-10	Usenix	USENIX	Boston, MA
Jun 6-10		Networld + INTEROP	Berlin, Germany
Jul 11-15	IEEE	IEEE 1003	
Sep 6-9	AUUG	AUUG '94	Melbourne
Sep 12-14		Networld + INTEROP	Atlanta, GA
Sep 19-23	Usenix	LISA VIII	San Diego, CA
Oct 23-27	ACM	OOPSLA	Portland, OR
Oct 26-28	Usenix	Very High Level Languages	Sante Fe, NM
Nov 12-18	DECUS	DECUS - Winter '94	Anaheim, CA

Date	Org	Event	City
1995			
Jan 16-20	Usenix	USENIX	New Orleans, LA
Feb	AUUG	Summer Conference Series	All States in Australia
Feb 21-23	UniForum	UniForum	Dallas, TX
May 13-19	DECUS	DECUS - Summer '95	Atlanta, GA.
Jun 19-23	Usenix	USENIX	San Francisco, CA
Sep 19-22	AUUG	AUUG '95	Sydney
Nov 2-8	DECUS	DECUS - Winter '95	San Francisco, CA
1996			
Jan 22-26	Usenix	USENIX	San Diego, CA
Feb	AUUG	Summer Conference Series	All States in Australia
Mar 12-14	UniForum	UniForum	San Francisco, CA
May 18-24	DECUS	DECUS - Summer '96	Orlando, FL
Nov 16-22	DECUS	DECUS - Winter '96	Anaheim, CA

AUUG Inc. is pleased to provide the following worldwide calendar of events, in cooperation with other open systems bodies. For updated information please contact the sponsoring organisation directly.