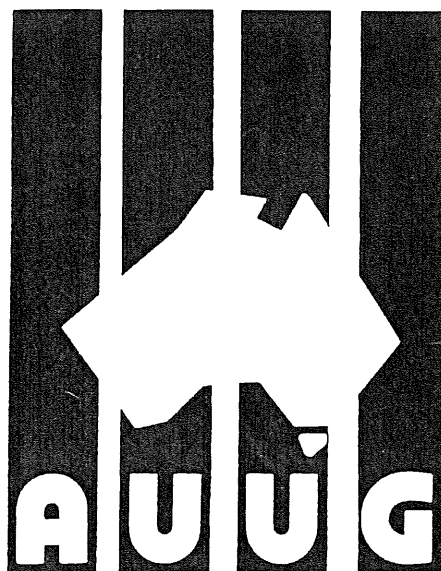


ISSN 1035-7521



AUUGN

AUUG Inc. Newsletter

Volume 14, Number 4

August 1993

The AUUG Incorporated Newsletter

Volume 14 Number 4

August 1993

CONTENTS

AUUG General Information		3
Editorial		5
AUUG Institutional Members		6
AUUG Management Committee Election Results		8
AUUG President's Page		9
AUUG Secretary's Report		11
Important Notification for Institutional Members		14
AUUG'93 Update		15
AUUG Local Chapters		
AUUG Queensland Chapter Update	<i>Mark White</i>	17
South Australian Chapter	<i>Michael Wagner</i>	17
AUUG Inc. - Victorian Chapter		18
Update on AUUG Inc. - Victorian Chapter Activities	<i>Stephen Prince</i>	19
WAUG and Perth News	<i>Janet Jackson</i>	21
WAUG - June Meeting Review	<i>Adrian Booth</i>	21
WAUG - July Meeting Review	<i>Janet Jackson</i>	22
Impressions of SAGE-AU 93	<i>Janet Jackson</i>	23
ACSnet Survey		27
AUUG Book Reviews		30
Wizard's Bookshelf		35
Open System Publications		35
Prentice Hall Book Order Form		36
WoodsLane Information		37
The Electronic Interviews	<i>Adrian Booth</i>	38
!AUUGN - from AUUGN Volume 1, Number 4		
A brief note on UNIX System Performance	<i>Ian Johnstone et al.</i>	43
The First Port is the Deepest	<i>Tony McGrath</i>	45
386BSD in the Shrink-Wrap World	<i>Peter Billam</i>	50
Open, Distributed OnLine Transaction Processing	<i>Anthony Hooten</i>	55
X/Open and Interoperability	<i>Peter Janecek</i>	68

From login: - Volume 18, Number 3

An Update on UNIX-Related Standards Activities	80
Unix Magic <i>Adrian Booth</i>	89
Unix Tricks- forcing NFS cache flushes <i>Janet Jackson</i>	91
Unix Traps- Endlessly recursive directories <i>Janet Jackson</i>	92
Summary of Management Committee Minutes - 2nd July 1993	93
AUUG Membership Categories	96
AUUG Forms	97
Advertisement	
C&T Computing	10
Softway	13
Prentice Hall Book Order Form	36
WoodsLane Pty Ltd	37

Copyright © 1993 AUUG Incorporated. All rights reserved.

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

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

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

AUUG General Information

Memberships and Subscriptions

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

Membership and General Correspondence

All correspondence for the AUUG should be addressed to:-

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

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

AUUG Business Manager

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

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

AUUG Executive

President **Phil McCrea**
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 Plains 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@iconix.oz.au
Iconix Pty Ltd
851 Dandenong Rd
East Malvern VIC 3145

Chris Schoettle
cts@usl.com
USL
PO Box 961
Crows Nest NSW 2065

AUUG General Information

Next AUUG Meeting

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

Advertising

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

Mailing Lists

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

Back Issues

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

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

Conference Proceedings

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

Acknowledgement

This newsletter was produced with the kind assistance of and on equipment provided by the Australian Nuclear Science and Technology Organisation. 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 14 Number 4, an issue with a historical flavour. Adrian Booth supplied an electronic interview with Greg Rose, and I've included in the !AUUGN section a report by Ian Johnstone *et al.* which is mentioned in this interview. The paper from the Sydney summer conference, *The First Port is the Deepest*, which was presented by Tony McGrath continues the historical theme.

Other papers, include Peter Billam's paper presented at SAGE-AU, Anthony Hooten's paper presented at UniForum NZ 93 and X/Open and Interoperability by Peter Janecek. There is also local chapter reports, AUUG Election results (as reflected in the General Information) and some impressions of the SAGE-AU 93 Conference from Janet Jackson. For institutional members there is an important notice relating to contact details.

There has been a lot of activity in the book review area. We have seven reviews in this issue and more coming in future ones. There is also a major announcement from the new distributors of the O'Reilly and Associate books in Australia, WoodsLane.

Finally, remember AUUG93 will be here soon. You should have received registration details in the mail. If not please contact the secretariat. It should be a good conference.

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 14 No 5 Friday 24th September
Volume 14 No 6 Friday 26th November

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

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

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

AUUG Institutional Members as at 05/08/1993

A. Goninan & Co. Limited	Computer De Tokyo Corporation
A.N.U.	Computer Law Corporation
AAII	Computer Sciences of Australia Pty. Ltd.
Aberfoyle Resource Limited	Computer Software Packages
Adept Software	Computer Systems (Australia) Pty. Ltd.
Alcatel Australia	Copper Refineries Pty. Ltd.
Allaw Technologies	Corinthian Engineering Pty. Ltd.
Amalgamated Television Services	Corporate Workgroup Resources
Amdahl Pacific Services	CSIRO, Division of Wool Technology
Andersen Consulting	Curtin University of Technology
ANI Manufacturing Group	Customised Software Solutions Centre
Animal Logic Research Pty. Ltd.	Cyberdyne Systems Corporation Pty. Ltd.
ANSTO	Cyberscience Corporation Pty. Ltd.
Anti-Cancer Council of Victoria	Cybersource Pty. Ltd.
Attorney Generals' Dept.	Datacraft Technologies
Attorney-General's Dept.	Deakin University
AUSOM Inc.	Deakin University
Auspex Systems Australia	Defence Housing Authority
Australian Airlines Limited	Defence Service Homes
Australian Archives	Dept. of Agricultural & Rural Affairs
Australian Bureau of Statistics	Dept. of Business & Employment
Australian Computing & Communications Institute	Dept. of Defence, Kingston
Australian Defence Industries Ltd.	Dept. of Education, QLD
Australian Electoral Commission	Dept. of Family Services &
Australian Information Processing Centre Pty. Ltd.	Aboriginal & Islander Affairs
Australian National Parks & Wildlife Service	Dept. of Industrial Relations,
Australian Software Innovations	Employment, Training & Further Education
Australian Submarine Corporation	Dept. of Planning & Housing
Australian Taxation Office	Dept. of the Premier & Cabinet
Australian Technology Resources (ACT) Pty. Ltd.	Dept. of the Treasury
Australian Technology Resources (WA) Pty. Ltd.	Dept. of Transport
Australian Wool Corporation	DEVETIR
AWA Defence Industries	Digital Equipment Corp. (Australia) Pty. Ltd.
B & D Australia	DSTO
Bain & Company	Easams (Australia) Ltd.
BHA Computer Pty. Limited	Electronic Financial Services Limited
BHP Information Technology	Equinet Pty. Ltd.
BHP Minerals Ltd.	Ericsson Australia Pty. Ltd.
BHP Petroleum	ESRI Australia Pty. Ltd.
BHP Research - Melbourne Laboratories	FGH Decision Support Systems Pty. Ltd.
BHP Research - Newcastle Laboratories	Financial Network Services
Bond University	Fire Fighting Enterprises
Burdett, Buckeridge & Young Ltd.	First State Computing
Bureau of Meteorology	Flinders University
Bytecraft Pty. Ltd.	Fremantle Port Authority
C.I.S.R.A.	Fujitsu Australia Ltd.
Cadcom Solutions Pty. Ltd.	G. James Australia Pty. Ltd.
Cape Grim B.A.P.S	GEC Marconi Systems Ltd.
Capricorn Coal Management Pty. Ltd.	Geelong & District Water Board
CelsiusTech Australia	Genasys II Pty. Ltd.
CITEC	General Automation Pty. Ltd.
Classified Computers Pty. Ltd.	GIO Australia
Co-Cam Computer Group	Golden Circle Australia
Coal & Allied Operations	Great Barrier Reef Marine Park Authority
Cognos Pty. Ltd.	Gribbles Pathology
Colonial Mutual	Gunnedah Abattoir
Com Net Solutions	Haltek Pty. Ltd.
Com Tech Communications	Hamersley Iron
Commercial Dynamics	Hermes Precisa Australia Pty. Ltd.
Communica Software Consultants	Honeywell Ltd.
Composite Buyers Ltd.	Hong Kong Jockey Club Systems (Australia) Pty. Ltd.
Computechnics Pty. Ltd.	I.B.A.

AUUG Institutional Members as at 05/08/1993

I.P.S Radio & Space Services
IBM Australia Ltd.
Iconix Pty. Ltd.
Ideas International Pty. Ltd.
Information Technology Consultants
Informed Technology
Insession Pty. Ltd.
Insurance & Superannuation Commission
Integration Design Pty. Ltd.
International Imaging Systems
Internode Systems Pty. Ltd.
Ipec Management Services
James Cook University of North Queensland
JTEC Pty. Ltd.
Knowledge Engineering Pty. Ltd.
Labtam Australia Pty. Ltd.
Land Titles Office
Leeds & Northrup Australia Pty. Limited
Logica Pty. Ltd.
Lyons Computer Pty. Ltd.
Macquarie University
Mayne Nickless Courier Systems
Medical Benefits Funds of Australia Ltd.
Mentor Technologies Pty. Ltd.
Metal Trades Industry Association
Mincom Pty. Ltd.
Minenco Pty. Ltd.
Mitsubishi Motors Australia Ltd.
MM Data Networks
Moldflow Pty. Ltd.
Motorola Computer Systems
Multibase Pty. Ltd.
Multiline BBS
Multiuser Solutions Pty. Ltd.
National Library of Australia
NCR Australia
NEC Australia Pty. Ltd.
Northern Territory Library Service
Northern Territory University
NSW Agriculture
Objectif Pty. Ltd.
Ochre Development
Office of Fair Trading
Office of National Assessments
Office of the Director of Public Prosecutions
Olivetti Australia Pty. Ltd.
Open Software Associates Ltd.
OPSM
OSIX Pty. Ltd.
OzWare Developments Pty. Ltd.
Pacific Star Communications
Paxus
Petrosys Pty. Ltd.
Philips PTS
Platform Technologies Pty. Ltd.
Port of Melbourne Authority
Powerhouse Museum
Process Software Solutions Pty. Ltd.
Prospect Electricity
pTizan Computer Services Pty. Ltd.
Public Works Department, Information Services
Pulse Club Computers Pty. Ltd.
Pyramid Technology Corporation Pty. Ltd.
Qantek
QLD Electricity Commission
Quality By Design Pty. Ltd.
Redland Shire Council
Release4
Renison Golfields Consolidated Ltd.
Repatriation General Hospital, Hollywood
Rinbina Pty. Ltd.
Royal Melbourne Institute of Technology
Scitec Communication Systems
Sculptor 4GL+SQL
SEQEB Business Systems
Shire of Eltham
Siemens Nixdorf Information Systems Pty. Ltd.
Snowy Mountains Authority
Softway Pty. Ltd.
Sony Technology Centre of Australia
South Australian Lands Dept.
St. Gregory's Armenian School
St. Vincent's Private Hospital
Stallion Technologies Pty. Ltd.
Standards Australia
State Bank of NSW
State Super (SSIMC)
Steelmark Eagle & Globe
Sterling Software
Sunburst Regency Foods
Swinburne Institute of Technology
Sydney Electricity
Sydney Ports Authority
System Builder Development Pty. Ltd.
Systems Development Telecom Australia
TAB of Queensland
Tandem Computers
Technical Software Services
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.
Toshiba International Corporation Pty. Ltd.
Tower Software Engineering Pty. Ltd.
Tower Technology Pty. Ltd.
Tradelink Plumbing Supplies Centres
TurboSoft Pty. Ltd.
TUSC Computer Systems
UCCQ
Unidata Australia
Unisys Australia Ltd.
UNIVEL
University of Adelaide
University of Melbourne

AUUG Institutional Members as at 05/08/1993

University of New England, Dept. of Maths,
Stats & Computer Science
University of Queensland
University of South Australia
University of Sydney
University of Tasmania
University of Technology, Sydney
UNIX System Laboratories
Unixpac Pty. Ltd.
Vicomp

Victoria University of Technology
VME Systems Pty. Ltd.
Walter & Eliza Hall Institute
Wang Australia Pty. Ltd.
Water Board
Western Mining Corporation
Work Health Authority
Workstations Plus
Zircon Systems Pty. Ltd.
Zurich Australian Insurance

Returning Officer's Report

Results of the 1993 AUUG Management Committee Election

A total of 217 ballot papers were received.

In addition, 5 ballot papers were set aside because the name on the envelope could not be matched against any name on the membership list provided to me by the Secretariat. One further ballot paper was set aside because it was not in the envelope with the declaration. One further ballot paper was set aside because it was received after the due date.

President: McCREA 130
PADDON 58
Informal 29

Elected: McCREA

Secretary: PADDON 52
WISHART 137
Informal 28

Elected: WISHART

Vice President: HUXTABLE 135
PADDON 54
Informal 28

Elected: HUXTABLE

Treasurer: CRAWFORD 154
PADDON 35
Informal 28

Elected: CRAWFORD

Ordinary Committee:

Elected: BIRNIE
MALTBY
PADDON
(2 positions vacant)

Returning Officer: Vacant

**Assistant Returning
Officer:** Vacant

Note 1:

The high informal vote is because a large number of voters failed to read and follow the instructions, regarding use of the number '1', but chose to mark their ballot paper with a tick or a cross. The procedures will probably be changed in future so that the ballot paper itself includes the direction to mark the number '1' etc.

Michael Tuke, Returning Officer

A Leopard really can change its spots

Well, it probably had to happen at some stage I guess. Twenty three years in the IT industry (assuming one's IT career starts on graduating from Uni or wherever) and I've managed to avoid it all that time. Not that it was there in the beginning - in fact its been around for only 10 years or so.

What on earth is he on about you say? Well, I should bear all I guess, and tell you. The fact is, your AUUG President has finally become a DOS user!!! Great howls of angst, flog me with a wet lettuce (apologies to Paul Keating),etc. At the ripe old age of 43, I have finally joined the rest of the computer industry, and started using DOS! I can now understand some of the things my sister's kids talk about to their friends!

So what prompted all this? Well it is mainly associated with a change of job. I felt that Softway needed some fresh leadership, so after taking appropriate steps to ensure that the management of the company was in good hands, I went out into the wide world to find something else to do (Actually the plan was to have a long holiday, which was thwarted - but that's a different story). In my new job at ANSAMS (that'll get you all scurrying for the Top 1000 list...), I had the option of using a UNIX environment for office type things, or using the DOS PC on my desk. So I thought "What the heck - let's find out what the rest of the industry have to put up with!" I should point out that my programming days are long past, and my use of a computer is purely for word processing.

Of course I'm not really using DOS - no-one really does these days. It's Windows that everyone uses. Even the owners of DOS gave it up as a joke years ago.

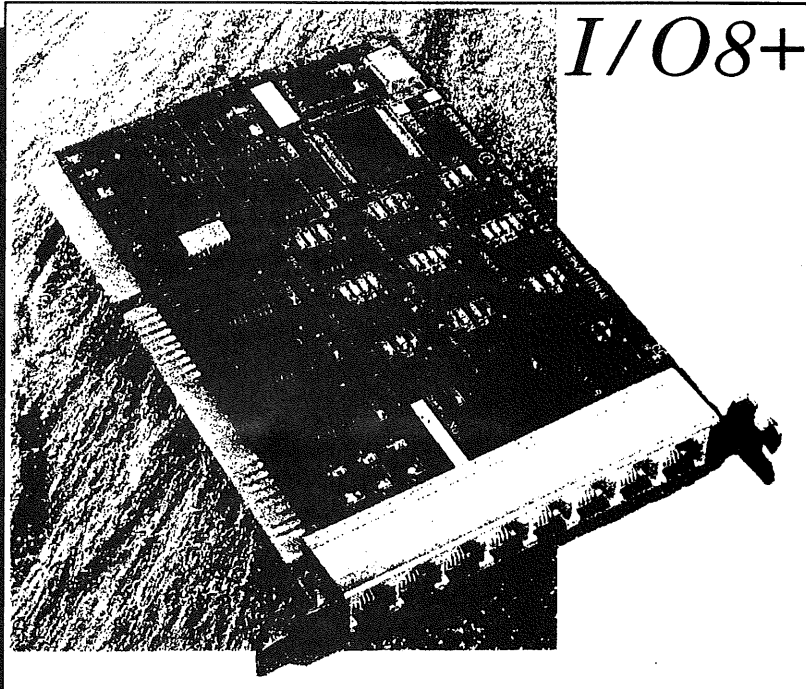
What do I like about it? Well, my impressions are coloured by the fact that I am a UNIX user of the old school - yes, folks, that means *vi* and *troff*! Using Microsoft *Word* in place of *vi* is the biggest single change. Not having insert mode is a real change - you actually get to use all those keys on the keyboard which have useful words on them - like *Home*, *End*, and *Delete*! If I had a dollar for every time I accidentally typed *a* to add a word, or *e* to step to the end of a word, I'd be able to buy a tape backup system by now for my PC!

The truth is, I'm actually starting to like my new computing environment, and was beginning to consider myself a traitor to the cause, till I realised what was really happening - it's not the operating system which will win the day. In the current battle for the desktop (where Microsoft is running hot favourite), the operating system which will run the popular packages will win in the end. For those of us who have an emotional attachment to UNIX, let's hope initiatives like WABI, which will enable Windows applications to run on UNIX, are successful.

There - having bared all, will you still have me as President of AUUG???

Phil McCrea

A Major Price/Performance breakthrough in Intelligent I/O Controllers

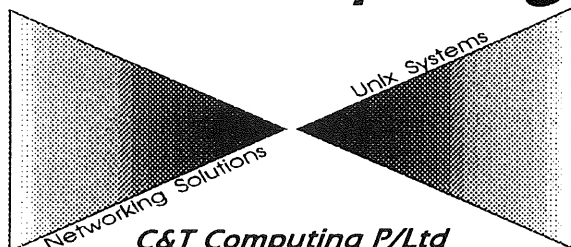


Introducing the new I/O8+. A low cost intelligent I/O controller from Specialix, renowned for quality and performance in connectivity solutions. Ideal for small multi-user systems, the I/O8+ is more than just cost effective, it represents a major breakthrough in price/performance.

The secret lies in its high power RISC processor architecture, developed specifically for data communication applications. This enables the I/O8+ to deliver a sustained throughput of 19.2Kb/sec on all 8 ports simultaneously, with speeds of up to 38.4 Kb/sec supported. The I/O8+ offloads work from the main processor, ensuring that the overhead on the central system is kept to a minimal level. It also incorporates that latest surface-mount technology, providing even greater reliability.

For technical and pricing details on the I/O8+, please contact your local supplier...

C&T Computing



C&T Computing P/Ltd
Unit 4 / 240 Vulture Street
SOUTH BANK QLD 4101
Phone : 07 844 3263
Fax: 07 844 5592

Specialix

I/O 8+

AUUG SECRETARY'S REPORT

1992/93

1. Membership

1992/93 has seen a growth in membership of around 10%. Table 1 shows the membership categories, comparing the equivalent periods (around June 30th) for the last few years. It is interesting to note that over the past 6 months we have had over 140 new members join AUUG. In the period May-June 1993 in particular, we have seen over 70 new memberships. This period corresponded to the surge in activities through the newly formed chapters. I believe that the increase in local events through the increased chapter activities will see a substantial growth in membership over the next 12 months.

	1990/91	1991/92	1992/93
Ordinary	359	445	488
Institutional	207	252	286
Student	10	10	9
Life	1	1	1
Subscriptions	18	22	19
TOTAL	595	730	803

Table 1: Membership Growth

Table 2 shows the membership by location and category as at 2nd July 1993. It shows a steady growth in those regions which have had active chapters (particularly ACT, QLD, NT and VIC). The slight decline in NSW membership will hopefully be addressed by the recent formation of a chapter for NSW. The figures for WA do not show WAUG members who have not yet done a "top-up" on their WAUG membership to get a full AUUG membership, we expect a number of new memberships for WA as WAUG members transfer to AUUG.

It is unfortunate that in May 1993 we had around 100 members from the January 1993 renewal period who were declared un-financial and taken off the membership lists since they had not paid membership fees. Some of these members had changed addresses and we were unable to contact them. It is likely that their renewal notices just did not reach them. Please remember when changing your address to let the Secretariat know your contact new details.

Region	Ordinary	Institutional	Student	Life	Subscription	Total	'91/92 Total
ACT	42	24	2	0	3	71	53
NSW	157	129	2	1	7	296	311
NT	11	4	1	0	1	16	6
Overseas	3	1	1	0	0	5	7
QLD	57	28	1	0	3	89	58
SA	26	13	1	0	0	41	41
TAS	6	2	0	0	0	8	7
VIC	160	73	1	0	5	239	217
WA	26	12	0	0	0	38	30
TOTAL	488	286	9	1	19	803	730

Table 2: 1993 Membership by Location and Category

2. Chapters

This past year has seen substantial growth in AUUG's chapters. A drive by the management committee has given AUUG a chapter in each mainland state/territory (and a Tasmanian chapter is coming!). Existing chapters in the ACT and Victoria have been joined by chapters in the Northern Territory, South Australia, Queensland and most recently, New South Wales. In addition, WAUG (the Western Australian UNIX Group) has changed from being an AUUG affiliated organisation to a fully fledged AUUG chapter.

The chapters are now starting to offer services directly to members in their local areas. The frequency and extent of services varies from chapter to chapter, but all chapters are now offering regular local technical meetings with local, interstate, and overseas speakers. As the chapters grow they will offer even more local events.

We expect a further increase in the number of chapters as other regions form new chapters. It is also likely that the regional chapters will be joined by special interest chapters sometime in the future. The success of chapters depends to a large extent upon the willingness of local members to participate in events. Local chapter events give members the chance to meet with peers and experts in many areas. I would encourage all members to consider giving a presentation at an AUUG event (local chapter meeting, summer conference or national winter conference).

The chapter council is the place where representatives from chapters meet to discuss activities and provide input to the management committee. The inaugural meeting will be held in August 1993 in Sydney. The meeting will help chapters coordinate their activities and also to share ideas on the types of activities that they can offer.

3. Increase in Membership Fees

July 1993 saw the first increase in AUUG membership fees since 1989. The new fees are shown in table 3.

Category	Membership Fee
Ordinary	\$90
Institutional	\$350
Student	\$25
Subscription	\$90

Table 3: Membership Fees From July 1993

The management committee was particularly concerned about the small number of student members. As can be seen in the tables above, we currently have only 9 student members. AUUG as an organisation, tries to further knowledge and interest in UNIX and Open Systems. To encourage more student members and recognising the reduced financial capacity of most full-time students, the management committee decided to drop the student membership fee from \$45 to \$25.

4. Membership Benefits

Our business manager, Liz Fraumann, and the management committee have been working to increase the range of membership benefits available to members. Access to these membership benefits is through the AUUG membership card which members receive when they join or renew.

All members will have received a membership handbook which provides details of membership benefits (training discounts, book discounts, hardware/software discounts ...), organisation rules (like chapter rules and constitution) and AUUG contact details.

5. AUUG Workers

The elections for the 1993/94 management committee saw less than a full slate of candidates. Following the election, two committee positions were still vacant. The management committee, using its constitutional powers had to co-opt two members onto the committee. It is disappointing that there was not sufficient interest from the membership in running for office in AUUG. I hope that next year we will see more interest in the elections.

However on a brighter note, there are an increasing number of members getting involved in running chapter activities. So in addition to the Management Committee, AUUGN Editor, Returning Officer (and Assistant), we now have chapter committees and officers heavily involved in running local events and representing AUUG to local businesses and people. I am sure that all members would join with me in thanking these volunteers for the time and effort that they put into making AUUG a success.

Peter Wishart
AUUG Inc - Secretary 1992/93



Moving to OPEN Systems?

Softway is Australia's largest UNIX systems software house.

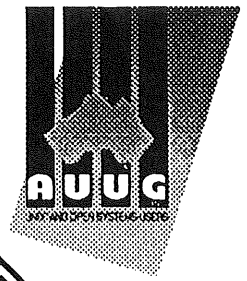
We understand the needs of the Open Systems marketplace, and can provide services in these areas:

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

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

79 Myrtle Street
Chippendale NSW 2008

PO Box 305
Strawberry Hills NSW 2012



PROCEDURE CHANGE

IMPORTANT NOTIFICATION FOR INSTITUTIONAL MEMBERS

PRIMARY CONTACT - ADMINISTRATIVE ONLY

In the recently updated membership application, Institutional Members will notice a change. Listed are three contacts: Primary Contact, 1st Rep., and 2nd Rep. It should be noted the Primary contact will be considered the point of contact into the organisation for updated information, billing, and general business affairs. This person is considered as the administrative contact **ONLY** and will **NOT** receive the benefits of AUUG membership unless they are also listed as either 1st or 2nd Rep.

If you are an institutional member and would like to ensure your contact information is up to date and correct, one of the following actions should be taken:

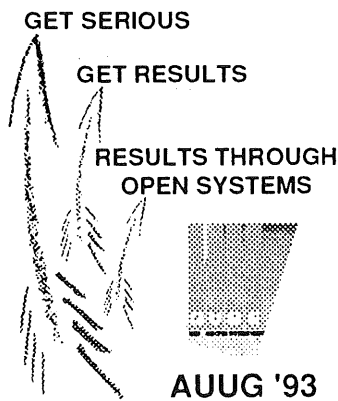
- 1) Complete the Institutional Member Application form included in this publication and mark it UPDATE.
Fax to: (02) 332-4066, or
Post to:
AUUG Inc.
P.O. Box 366
Kensington, NSW 2033.

OR

- 2) Call AUUG Secretariat on: (02) 361-5994 for current information listed and update as appropriate.

It should be noted, as of 1 October 1993, Primary Contacts will be treated as the **ADMINISTRATIVE CONTACT ONLY** and all benefits will be designated to the 1st and 2nd Rep. If you fail to update your information and are listed as the Primary Contact only, you will not receive *AUUGN* and other benefits after this issue.

If you have any questions, you may contact Liz Fraumann on: (02) 953-3542 tel/fax.



UPDATE

By Liz Fraumann

Sydney, August 1993 — Enthusiasm continues to build as AUUG '93 approaches. A first for the organisation, was a Computerworld "belly-band" ad which has already reaped significant interest in the tradeshow VIP passes and calls for registration information. Exhibition organiser, Wael Foda stated, "There has been increased interest," and proudly stated, "we have recently signed on Novell and Microsoft to the exhibition."

Journalists Interest Peaked —

Local and national publications also have shown a keen interest in both the conference and exhibition with feature articles running in *The Australian*, *Asia/Pacific Open Systems Review*, *Computerworld Australia*, *Asia/Pacific UNIX Update*, and regular coverage in *MIS Magazine*, *Informatics*, *PC Week*, *The Australian Financial Review*, and *Communications Engineer*. Public relations consultant, Lachie Hill, feels this is just the beginning and anticipates continued and increased coverage as the open systems event in Australia draws near.

Are You Connected?

A specific point of interest to both delegates and exhibitors, Com Tech will be providing internet connections for the exhibition floor and a delegate "terminal room." Exhibitors interested in "connecting" should contact ACMS on (02) 332-4622 for detailed information. Labtam Australia will once again provide the terminals for the terminal room. Delegates will be provided the opportunity to "stay in touch" with their home office while participating in the conference and exhibition. A change from last year, a sign up schedule will be likely for better CPU utilisation.

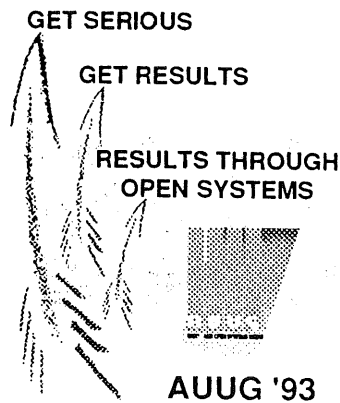
Where in the World?

Delegates for AUUG '93 will find an easier time navigating the Sydney Convention and Exhibition Centre than they did in 1991. AUUG staff, designated by the AUUG '93 staff tee-shirts and an "ASK ME" badge, will provide assistance in negotiating the maze of rooms. This effort is in addition to the monitor information and signage.

Birds-of-Feather...Flock Together —

Introduced last year in Melbourne, the BoF sessions promise to be one of the highlights of the conference. This year, no less than eight sessions have been organised to date. Topics such as Integrating UNIX and Novell LANs, SAGE, Super Nova, SCO Networker Forum, Plan 9, SUG and UNIX vs NT - the user experiences, will provide excellent two way communications between hosts and participants. Delegates will receive an update as to times, locations, etc. with their satchel at registration. Additional and last minute updates will be in the AUUG NEWS DAILY, and posted at the registration area.

more - more - more

***Cron Times Matures —***

In conjunction with Strategic Publishing Group, AUUG Inc. will produce the sequel to the "show daily" which was introduced last year. Unlike movie sequels, the *AUUG NEWS DAILY*, is a significant improvement. An up to minute daily publication, *AND*, will be

distributed at the conference and exhibition. It will highlight the event activities, product launches, and other significant industry news surrounding AUUG '93.

Save - Save - Save —

All AUUG members should have received the Programme and Registration Information brochure by now. If you have not, please call: (02) 332-4622 and make your request. There is also a significant savings if you submit your registration PRIOR TO 28 August. Member price on or before this date is \$350 for the full 3 day conference and exhibition and jumps to \$420 there after. The Tutorials have a similar rate increase after the 28th of August. % > PAY EARLY = SAVE!

Partner Programme —

Organisers are aware this years conference falls during the school holidays and has taken measures to provide activities for partners accompanying delegates. A Sydney Explorer tour, Harbour cruise, and a full day to the Blue Mountains are just a few activities which can provide entertainment while delegates work. On page 18 of the Programme and Registration Information interest in these activities should be indicated by placing the number persons desiring to participate. This is not to be considered making the reservation. Interested parties will receive follow-on communication. For a nominal fee, partners may also join the social programmes of the conference. This is also to be indicated on the registration form on page 18. This indication is a reservation and is to be included in the Total Payment section.

FOR DETAILS —

AUUG '93 Conference and Exhibition information for yourself or a friend...

CALL: (02) 332-4622

FAX: (02) 332-4066

AUUG Queensland Chapter Update

The July meeting of the Queensland AUUG chapter (QAUUG) was held on the 27th, at our usual venue, Brisbane's historic Regatta Hotel. For those who've never had the pleasure of a drink at the Regatta let me tell you that it's a great old pub, overlooking the Toowong reach of the Brisbane River.

Guest speaker for the evening was David Hughes, Senior Network Programmer at Bond University, who gave an interesting and objective comparison of the CMIP and SNMP network management protocols.

After David's presentation, the second "show and tell" segment, featuring Stallion Technology's Greg Ungerer, got underway. These mini-presentations were first introduced by the QAUUG committee at the June meeting, as a means of getting to know various QAUUG members, many of whom are sometimes recognizable only by their e-mail address! Greg described many of the interesting projects he's been involved with at Stallion over the last three and a half years.

The results of a survey held to identify areas of local interest were also distributed. This information, and suggestions from members, will be used to plan guest speakers for future meetings.

QAUUG Meetings are held on the last Tuesday of each month, at the Regatta Hotel, Coronation Drive, Toowong. All Open Systems users, managers, and developers are welcome to attend. For further information on QAUUG and it's activities please contact Greg Birnie on (07) 340 2111, or e-mail <greg@lna.oz.au>.

Mark White

<M.White@ncstt.jcu.edu.au>

AUUG South Australian Chapter formation

The inaugural meeting of the South Australian chapter was held on Tuesday 20th July. Ten members elected unopposed a committee of four to arrange local events and organise the 1994 Summer Technical Conference. The committee members are:

Michael Wagner	- Secretary / Chairman	- Systems Services
Alastair Dick	- Treasurer	- Telecom
Bob Korbel	- Committee member	- Amdel
Rodger Chan	- Committee member	- Amdel

The first meeting will be an informal get-together over lunch on Wednesday August 12th at 12:45 pm* - venue to be arranged.

Michael Wagner

AUUG SA Chapter - Secretary / Chairman

<mhw@syserv.com.au>

* This is included for information only, as it will be well out of date when members receive this issue.

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, 2 September '93	Social
Tue, 12 October '93	Technical
Wed, 24 November '93	Technical
Thu, 16 December '93	Social
Tue, 25 January '94	Social
Wed, 2 March '94	Technical
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.auug**, or on the mailing list **sesspoole@clcs.com.au**.

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

Update on AUUG Inc. - Victorian Chapter Activities

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

The last six months since the AUUG Vic. annual general meeting and election have seen the committee working hard to make good it's promise by aligning itself with the AUUG Chapter Rules and Policy, and the focus of it's activities.

For those not familiar with the history of AUUG Vic., it was originally known as SESSPOOLE and focused on the south eastern suburbs of Melbourne, held six informal social meetings every year and its members organised the Victorian AUUG-Summer conference for the last four years. Since the AGM the focus has changed to all of Victoria and the meetings are now alternating as social and technical.

This period has seen four official committee meetings to resolve some of the issues. All the meetings have been accurately recorded by the diligent efforts of our secretary, *Neil Murray*. To give you all some insight as to what has been happening I'll try to summarize the main areas below.

Financial Matters

The first task undertaken by the new treasurer, *John Carey*, was the establishment of a bank account and the transfer of funds from the national treasurer. This all had to be done prior to the first technical meeting in April. The candidate banks were narrowed down to two; Commonwealth and National Australia Bank (NAB). After discussing the pro's and con's, the committee finally settled on the NAB.

John has since managed to formulate a budget based on experience gained from the couple of technical meetings. For the first time since the summer conference series started four years ago, we are looking to cover the costs from our own funds. This should make the process of organizing Summer94 (Vic) a lot easier. In the past we have relied on a float from the national body. However, according to initial estimates, cash flow around that time of year could be tight.

Hopefully, with all the accurate recording of expenses this year, the task of deciding on a budget for next year will be an easy job.

Rules and Policy

All committee's need an official set of rules in which to work under, (or have I just been around law firms for too long:-) hence a document detailing the rules and policy was formulated. This is meant as an extension to the AUUG constitution and it's chapter rules and policy document. The main aim of such a document is to define the tasks of the chapter's office bearers.

The current version (1.14) of the rules and policy has been sent to members on the *sesspoole* mailing list for comment. Once the members agree on it's contents, it will be submitted to the AUUG executive for approval and eventually reprinted in AUUGN.

Administration

The committee decided it's flag-ship advertisement in AUUGN should be revamped to reflect the change in direction that SESSPOOLE has taken this year. In short, these included a mention of the technical meetings, and any focus to the south eastern suburbs shifted to "AUUG Inc. - Victoria Chapter".

Curiosity was raised about how many AUUG-Vic members could be reached via email, for the sake of announcements, etc. After obtaining a copy of the Victorian section of the AUUG database, I pinged the email addresses which were not already listed on the chapter's mailing list, and found some interesting results. Of the 266 Victorian members, 99 had an email address that wasn't on the list. 34 of these addresses are no longer valid, that is, they bounced with a host or user unknown. My apologies to the *connect.com.au* postmasters for all the bounced messages being dumped on their mail boxes. Of the valid addresses, 41 were added to the mailing list, 8 decided against it and the remainder have not answered my ping. This takes the total number of members on the list to about 78.

If any members in Victoria with an email address did not hear from me, and they would like to be included on the mailing list, just drop a line to *sesspoole-request@clcs.com.au* and ask to be added. Don't worry, this mailing list won't flood your mail box; it's currently low volume with only meeting announcements and a few administration messages.

Technical Meetings

An idea floated at the annual general meeting was to trial alternate technical and social meetings. Well, to date, we have held two successful technical meetings. I won't describe too much about the contents of the technical meetings here, since *Michael Paddon* will provide a detailed overview in a separate article.

Our first technical meeting, was held on 29th April at the *Australian Computing and Communications Institute* (ACCI). Thanks to *Greg Rose* for organising the venue. An attendance of about 25 people heard two talks. The first, by *Lim O. Sim* on the subject of a 'C task package' he has been developing, and the second by *James Gardiner* about working with computer animation.

The second technical meeting was held on 21st July at Melbourne University's new Computer Science department (nice building). On behalf of the committee, we would like to thank *David Hornsby* for a magnificent effort in organising the venue at short notice and providing us with online debugging of the audio-visual system during the meeting. This meeting was extremely well received, with an attendance of approximately 100 people. The hot topics presented on the night included; a technical talk and real life demonstration by *Steve Jackson* from Microsoft on 'Windows NT', and a most entertaining talk from *Bob Smart* about the in's and out's of 'Linux'. The meeting concluded with a panel session consisting of Steve and Bob.

Both meetings finished off the night with members consuming pizza by the metre. General consensus is that the two meetings have been well worth the effort, well done Michael.

For those interested in the financial matters, the two technical meetings have incurred costs of \$319.15 and \$270.10, respectively. And has cost the members \$0 to attend!! The committee is currently looking at ways to reduce the costs of the meetings, since these figures are made up of printing, mailing and labour costs their is possibly room for improvement.

The committee is still looking for some short talks or 'work in progress' presentations to fill some of the future slots. As an incentive to potential speakers, the committee has reserved a small amount of \$\$'s for each meeting as a 'speaker honorarium'. The disposition of this money is at the total discretion of the programme chair.

Finally, with two technical meetings left this year, we hope to hold at least one of them (probably

the last one) away from the Melbourne CBD for the benefit of our country members. Work is already underway to organise this meeting in the Geelong area. In order that Melbourne based members can also attend, we have decided to change the 24th November (Wednesday) meeting to 20th November (Saturday) and make it an afternoon, followed by dinner somewhere in Geelong. Stay tuned for further details on this one.

Social Meetings

To date, there have also been two social meetings held. I'm happy to report that these meetings have continued in the same fashion as before the change in direction; with members discussing issues related to open systems over a friendly drink.

Other Business

The committee is always looking for ways to provide and improve benefits to Victorian members. A number of ideas are being debated at the present time, but these are not concrete yet, so I won't go into details.

One idea that has been adopted, in an effort to get country members more involved and give them better access to activities, is a rebate on the summer conferences. The current policy is to rebate the 20% of membership dues which AUUG-Vic receives, to members residing greater than 150km from the Melbourne GPO.

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

WAUG and Perth News

I'm writing this column on my last day of work before a two-week ski-ing holiday. Thanks to the aus.snow newsgroup posters and archive maintainers for keeping us up-to-date on the conditions, which are, well, we hope they will improve by the time we get there! Coming from WA, we have to book our trips in about January, unlike the lucky Melbourne folks who can duck up for a weekend when the snow looks good.

It's also my last day working at the Department of Computer Science at The University of Western Australia. I'm moving two buildings over, to the Centre for Water Research, whose systems I will be managing. The new job will present new challenges, so it's good to know I have so many colleagues in AUUG and my local chapter WAUG.

WAUG has changed its meeting venue to the Blue Note Bar and Restaurant in West Perth. This venue has better lighting than the previous one, the room is a better shape, and the food seems to be superior. However, if you want to buy a drink (as most of us do) you have to go to the bar. Now, I don't mind bars, but I do feel rather uncomfortable paying for my drinks while fighting an urge to throw a blanket around the barmaid. I'm concerned that there may be AUUG members who feel even more uncomfortable and are put off coming to our meetings.

Attendance at the June meeting, the first held at the Blue Note, was unusually low, despite the interesting topic (Glenn Huxtable on Tcl and Tk: see Adrian's review elsewhere in this AUUGN). This was probably because of the sudden venue change and rather late meeting notices.

Attendance at the July meeting was back to normal, despite what I think is a much more boring topic -- relational databases. I guess there are a lot of companies out there who rely on databases to run their businesses.

I recently had the pleasure of attending the inaugural conference of the Systems Administrators Guild of Australia (SAGE-AU). As a systems administrator I thought it was an excellent conference, as you'll see from my review of it. The good news for WA is that next year's SAGE-AU conference is planned for Perth. SAGE-AU is not part of AUUG and its interests are not limited to Unix, but given the number of systems administrators in AUUG, I figure its activities will be of interest.

Well, I've got a lunch to attend, and a speech to give, unfortunately. See you at AUUG'93.

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

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

June Meeting Review

Tcl and Tk — *Glenn Huxtable*

The low turnout for this month's meeting was disappointing, particularly given the interesting subject of the talk. Perhaps the change of venue and the comparatively late meeting notices were responsible.

Tcl and Tk are two related tools that can be used to dramatically improve the ease of X Windows programming. Tcl - which stands for "Tool Command Language", and is pronounced "tickle" - is a simple interpreted language. It also includes a library of C routines which can be bound into any C program.

While useful in its own right, Tcl has also been used to implement Tk, an X11 toolkit (i.e. C library). Tk is invoked using Tcl commands within the application. One exciting feature of Tk is a *send* primitive, which allows one Tk application to send Tcl commands to other Tk applications.

Wish - the Window Interpreter SHell - was also mentioned as being the simplest possible Tcl/Tk program - it merely accepts lines of input from the keyboard and passes them to the Tcl interpreter for execution.

Glenn suggested that Wish should have been called `tclsh`, so it could be referred to as “ticklish”. This led to not a few groans from the audience.

Glenn then showed us several examples of Tcl/Tk on an overhead screen display. The performance of Tcl/Tk does not seem to be a problem - one example (an animated Towers of Hanoi solver) was moving too fast to follow.

Tcl/Tk are two very powerful and impressive tools which are claimed by their author to make an X Windows programmer 10 to 50 times more productive (than one who uses the native X libraries). They should be in the toolkit of any X Windows programmer.

A book on Tcl will be out shortly. In the meantime, a draft of that book is available via the Internet - email Glenn for more information.

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

July Meeting Review

Relational Databases — *Ian McEwan, Ingres Pty Ltd*

I'm not particularly interested in databases, and I was rather tired that evening, but no-one else has reviewed the July talk so I suppose I'll have to try.

Ian started by saying that he had originally intended to talk about the future of relational databases, but had decided to talk about the state of the art instead.

Ian outlined a whole slew of features, most of which were never mentioned in the databases course when I was at uni. I say “outlined” because he didn't give details of how the features were implemented (not surprising I guess), but described what they meant, their benefits and in some cases, how they compared to older ways of doing things.

It was not clear to me which of the features were in all modern commercial relational databases, which ones Ingres had that its competitors didn't, and which ones Ingres was intending to put in a future release. Until the end there wasn't much explicit mention of Ingres itself.

For those who care, Unix was mentioned once.

There were a lot of slides and Ian got through them quite fast, but I still found the pace slow. Each slide seemed to present a bite-sized concept but take the length of a meal to explain. Perhaps this is just a reflection of my state of mind at the time.

This presentation was what I would call “marketing-technical”, meaning it was not really technical but was trying to *sound* technical in order to soft-sell the product. To be *really* technical, a talk must present ideas to our minds, rather than our purses.

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

Impressions of SAGE-AU 93

The inaugural annual conference of the Systems Administrators Guild of Australia came at an interesting time for me: I'd just accepted a new systems management job. I'd already been shown around my new department's systems,* so I had two networks to think about as I listened and learned.

And I did learn. I've never before been to a conference where almost every talk presented ideas relevant to my work. And as for having 60 or 70 fellow systems administrators to discuss ideas with... well, the word that comes to mind is "fabulous".

Wednesday — tutorial day

The conference was held at Melbourne University, a campus not nearly as beautiful as The University of Western Australia, where I work, but much better placed for restaurants and hotels. My day started with a good breakfast (included in the rate, too) at the Townhouse Hotel, then a short walk up the street to the conference venue. However I didn't enjoy the walk because it was raining (only to be expected in Melbourne), and I'd left my umbrella somewhere in Myers' city store the previous day.

"You're wet," said Adrian Booth, flatly. The ladies' room mirror reflected a drowned rat, but fortunately the room was civilised enough to have paper towels as well as hot-air dryers. I towelled my hair and, dignity restored, proceeded to the Solaris tutorial.

Ah, the Solaris tutorial! We waited behind solid wooden desks in the elegant panelled room, knowing its ambience would soon be ruined by the clamour of fifteen or twenty systems administrators grappling with the reality of Solaris 2.

But after about half an hour we began to wonder if there would be a tutorial at all. An anxious organiser assured us that the presenters had indeed been located and were on their way.

They did arrive, after another half hour. Apparently no-one had told them where on campus to come to. This seemed to be the only slip-up in an otherwise smoothly-run conference.

The tutorial was presented by Fraser Gardiner, Rich Baxter and Dennis Letizia, from Sun Microsystems. Rich (or it might have been Dennis, I'm not sure) started by showing us all the screens of Sun's graphical pseudo-sysadmin tool. This was rather pointless, because none of the audience seemed to have any intention of using such an uncustomisable tool, and even if we did, I had no doubt we'd all be bright enough to navigate its simple screens.

Fraser then showed us how to set up Auto-install (otherwise known as Jumpstart), which looks pretty useful. Fraser clearly knows his stuff and gave an excellent practical description — the sort of thing I expect at something called a tutorial.

Unfortunately, the other two then presented product pitches for NIS+ (which looks good, but I already knew that) and Sun Net Manager (nothing new there).

After the lunch break it was my turn to present a completely different tutorial ("Perl for Systems Administrators"), to much the same audience. But first, I had to eject the SAGE-AU committee from the room, where they were holding their first face-to-face committee meeting. (Their one previous meeting was held by email, took several hours, and by all accounts was a most successful experiment.)

Three hours later, suffering from presenter's throat, legs and eyes, I hated to think how Brent Chapman, who had presented the all-day Internet Firewalls tutorial to a packed-out room, must be feeling. (But

*Literally. For some reason, my predecessor found it necessary to walk me along the length of each segment of Ethernet and LocalTalk, pointing out each computer, transceiver, repeater, terminator — and quite a few users, too. This was good exercise, but doesn't mean I can draw a map of the net, and as for recognising the users...:-)

don't let me put you off speaking at conferences! The rewards outweigh the inconveniences.)

Thursday — conference day 1

After a welcome from SAGE-AU president Hal Miller, Elizabeth Zwicky shared her experiences of heterogeneity: trying to integrate different machines and operating systems into a coherent network. She distinguished coarse, medium and fine heterogeneity. Coarse meant completely different operating systems, like Unix and Macintosh, where you want to provide common services, such as email. Elizabeth's advice here was to use the simplest possible protocols. Medium meant different vendors' implementations of the same operating system. Elizabeth spent most of the talk describing the specifics of integrating different implementations of Unix. Fine heterogeneity, the worst kind, meant different implementations [of Unix] from the same vendor (Solaris 2 and SunOS 4, for example).

After coffee, Gregory Bond described how his company made a successful decision to put X-terminals, rather than workstations, on their users' desks. He gave us a lot of information about the pros and cons of both solutions.

Mike Selig, of Functional Software, described the rather elegant database approach used in Functional's industrial-strength Unix administration software. I've heard a number of talks about this work, and I thought Mike explained their approach particularly well.

We then enjoyed a stand-up buffet lunch in the combined exhibition/registration area. The food was nice, but anarchistic queueing caused contention, with people trying to move in several different directions.

The exhibition was low-key, with stands from a couple of vendors and a bookseller.

Labtam and Melbourne University provided X terminals and an Internet connection. There was some trouble getting them going, despite at least ten well-meaning advisors and about fifty others ardently ignoring the problem as only sysadmins know how:-)

After lunch Geoff Huston, an impressive, entertaining speaker, explained the status of AARNet — traffic doubling every eight months! — and attempted to predict its next few years. He thought it could become too big for the current owners (the Australian Vice Chancellors' Committee) and may end up privately owned.

Neil Brown from the University of New South Wales told us that "Installing Software can be Fun". Despite the title, I didn't enjoy this talk. I found it frustrating. With Neil's Conform system, his team control their software installation by programming in a rather esoteric special-purpose language (or perhaps the code is automatically generated: I couldn't be sure). This idea sounded interesting, but Neil spent a lot of time describing the language, in tedious detail, without giving a clear description of the overall system. It was also unclear how well the system worked in practice.

After a coffee break, SAGE-AU held its first Annual General Meeting, an informal affair. Having done such a good job of the conference, the committee were re-elected.

Conference dinner, Thursday evening

It seems that by and large, systems administrators are a scruffy lot. But they scrubbed up reasonably well for the conference dinner, which was held in Melbourne University's rather sumptuous University House.

The food was good (although fighting through a pastry crust to drink my soup is not really my thing), the conversation stimulating and the wine plentiful. I was amused to discover that I am not "a reasonable

female”, because I don’t like port.

Friday — conference day 2

Like many people, I was late for Mark Andrews’ talk on how to clean up a Domain Name Service hierarchy. (This was partly an after-effect of the dinner, and partly because the talks started half an hour earlier than the previous day’s.) Mark’s was a tutorial-style talk explaining various mistakes people commonly make with their DNS records. He gave lots of information, but had used a rather small font on his slides. Unfortunately I sat near the back, and found it impossible to take notes while peering at the screen (admittedly, through rather bleary eyes).

Brent Chapman then woke us up with a short, animated talk on the do’s and don’t’s of Internet Protocol packet filtering. An important point was that different vendors’ routers will apply your filtering rules in different orders, which can make a big difference to which packets they let through.

Next, Gregory Bond explained how to convert an old diskless workstation into an X-terminal. This talk was an excellent short tutorial. Hmm... maybe that’s what I can do with that old Sun 3 in my new department.

I spent the coffee break getting increasingly nervous, because I was the next speaker. I was surprised to find I was so much more nervous about a fifteen-minute talk than about yesterday’s three-hour tutorial: I’ll blame the preceding speakers for setting such a high standard!

I was disappointed to be giving the only fifteen-minute work-in-progress talk: I hope to see more of these at future conferences. I described a way to distribute certain networked system files, such as printcaps, using a recursive invocation of `make`, and was pleased with the number of ideas and suggestions that came from the audience. Sharing ideas is one of the things SAGE is for.

I felt much better all of a sudden, and enjoyed David Jones’s description of his university course in systems administration. He takes a small class for a semester, giving them a Unix system to look after. His course involves gradually teaching the students the basics of systems administration, from software installation to ethics, and includes an “emergency week” where students are expected to cope with simulated crises. The course doesn’t make them good systems administrators (not in 13 weeks!) but does give them a head start in learning how to be good systems administrators. David mentioned one problem with the course: no-one seems to want to use the student machine for anything much, so it’s hard for the students to get experience with real users. Another problem I can see is that academics interested in systems administration appear to be pretty rare.

Peter Billam then described the environment he provides for his users at the Tasmanian Treasury Department. He has developed a simple, rather elegant login shell that presents users with a menu of options, including “Change preferences”, “Contact systems administrator” and “About”, as well as mail, news and site-specific applications. A pleasant surprise was that this is all done under 386BSD.

At lunch, the previous day’s queueing problem didn’t happen, which showed that systems administrators are good at adapting!

The first afternoon speaker was Elizabeth Zwicky, who talked about backups for an hour and a quarter. I thought this was too long, for both speaker and audience. It was really two talks in one: the first, about how to design a good backup policy, and the second, about the inadequacies of the various Unix backup programs. Having already read her excellent LISA conference paper on the second topic, I found the first topic much more interesting, but other attendees may not agree with this.

Steve Landers then presented a different angle on the same topic, with his description of Functional Software’s backup scheduling architecture. This system provides a coherent, flexible front-end through which a relatively inexperienced operator can drive any of the Unix backup programs. (Steve seems to like `cpio`, whereas Elizabeth prefers `dump`. Sorry Steve, I’m with Elizabeth on this one. Still, you should see what they’ve done to `dump` in OSF/1.)

After coffee Srdjan Nikolic described the Remote Console Access Facility (RCAF) developed by Fujitsu for a client. I was pleasantly surprised that this definitely not a typical “vendor” talk. Srdjan gave a good deal of technical detail about the RCAF, which provides logging and network management as well as a system console located at the other end of a networked or dedicated connection.

The final talk was about Silicon Graphics’ NetVisualyzer. This did seem to be a “vendor” talk, extolling the virtues of the product without telling us very much about it (although a few screens of it were shown). However, it was presented by a customer, Peter Bonnelo from BHP Minerals International. He had been using NetVisualyzer for a short time — not really long enough to give a balanced view of it. Although vaguely interesting, this talk was not strong enough to be a good conference-closer.

After the closing remarks, a bunch of us adjourned to the nearby Clyde Hotel. Here, I ran into an amazingly familiar-seeming man who certainly knew me. However, after three days conferencing, my brain was overloaded with new faces, and I just couldn’t think who he was.

“What are you doing here?” he asked.

“Drinking with some people over by the window.”

“But what brings you to Melbourne?”

“I’m here for the Systems Administrators Guild conference.”

At this point I still hadn’t placed him, so I excused myself and returned to my seat. After about five minutes, some background job in my mind finally exited, and I suddenly remembered him. He was one of the people who had interviewed me for my new job!†

Janet Jackson <jackson@cwr.uwa.edu.au>

†A few days later I saw him again, apologised, and all was well.

ACSnet Survey

1.1 Introduction

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

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

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

Please send replies to:

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

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

Technical enquiries to:

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

Thank you

=====

1.2 Contact Details

Name: _____
Address: _____

Phone: _____
Fax: _____
E-Mail: _____

1.3 Site Details

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

New Connections

If you require a network connection please complete the following section.

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

- A1. Do you currently have networking software? Yes No
- A2. If **no**, do you require assistance in selecting a package? Yes No
- A3. Are you willing to pay for networking software? Yes No
 If **yes**, approximately how much? _____
- A4. Do you require assistance in setting up your network software? Yes No
- A5. Type of software: SUNIII MHSnet UUCP
 TCP/IP SLIP
 Other (Please specify): _____
- A6. Type of connection: Direct Modem/Dialin Modem/Dialout
 X.25/Dialin X.25/Dialout
 Other (Please specify): _____
- A7. If **modem**, connection type: V21 (300 baud) V23 (1200/75) V22 (1200)
 V22bis (2400) V32 (9600) Trailblazer
 Other (Please specify): _____
- A8. Estimated traffic volume (in KB/day): < 1 1-10 10-100
 (not counting netnews) > 100: estimated volume: _____
- A9. Do you require a news feed? Yes No
 Limited (Please specify): _____
- A10. Any time restrictions on connection? Please specify: _____
- A11. If the connection requires STD charges (or equivalent) is this acceptable? Yes No
- A12. Are you willing to pay for a connection (other than Telecom charges)? Yes No
 If **yes**, approximately how much (please also specify units, e.g. \$X/MB or flat fee)? _____
- A13. Once connected, are you willing to provide additional connections? Yes No
- A14. Additional Comments:

Existing Sites

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

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

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

AUUG Book Reviews

Well, the book reviews are flowing again. We have seven in this issue and lots more coming in following issues. Hopefully, elsewhere in this issue of AUUGN we have a major announcement from Woodslane, the new Australian distributors of the O'Reilly Books, one of the best respected and most sought after series in the Unix book market. As if you need to know more about the O'Reilly books, I've republished some of Ian Hoyle's reviews (with his permission, of course), which he previously mailed to the AARNet-contacts mailing list.

The agreement with Woodslane will complement our current agreement with Prentice-Hall. We hope to conclude agreements with other publishers and distributors in the near future.

All this means that we will have lots of books for review. The current practice is to post a note to the newsgroup *aus.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 9404, with your contact details and preferences.

Frank Crawford

High Performance Computing

by Kevin Dowd
O'Reilly and Associates
ISBN 1-56592-032-5

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

Hands up who has a purportedly "high performance" workstation at their site.

Who has been inundated by a never-ending stream of so-called benchmark numbers, MIPS, SPECS, TPC, cache sizes, I/O speed etc etc ??? Who looks with some disdain upon the check-coated sales droids who trot out these figures with all too monotonous regularity??

This new book in the O'Reilly stable of computer titles may go some way towards making those of us who find themselves in the above situation more able to pare away all of the marketspeak and ask the really hard questions of the vendors pushing their equipment at us.

For some time I have wanted a source of information that would help with my understanding of just what it is that make workstations actually go fast. Split into four sections,

- Modern Computer Architectures,
- Porting and Tuning Software,

- Evaluating Performance,
- Parallel Computing,

the author leads us through processor and memory designs, software issues including optimizing compiler techniques and advanced coding tricks, benchmarks and finally a look at the differences between massively parallel (MIMD & SIMD) and shared memory multiprocessors.

Certainly the book is up to date, including mentions of latest processor releases such as the MIPS R4000 & SSR, Intel Pentium and DEC Alpha. I found the section on shared memory multiprocessors very relevant for issues confronted at these laboratories as the vendors reach for new performance levels by introducing multiple processor machines.

The only real omission I found was the lack of an in-depth treatment of workstation "farms" now being marketed by companies such as IBM, DEC and HP. Only passing mention is made of PVM, Express or Linda and their use in treating a network of workstations as a single parallel machine (or "farm").

As has been the case with O'Reilly book the writing style is very clear, indeed occasionally quite humorous. I especially liked the analogy drawn between the MIP performance metric and pie eating contests !!

Recommended reading

Smileys

by David Sanderson
O'Reilly and Associates
ISBN 1-56592-041-4

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

What else can one say about a book devoted to 650 curious combinations of text characters meant to be read on their side!!! As the author says at the start:

A smiley is a sequence of ordinary characters you can find on your computer keyboard. Smileys are used in e-mail and other forms of communication using computers. The most popular smiley is a smiling face that people use to say 'don't take what I wrote too seriously'

Smileys are a serious subject! :-)

Certainly I've seen lots of these things in email and USEnet news messages over the years, but party smileys ????

```
#-) partied all night
%*} very drunk
:*) drinking every night
%-<I> drunk with laughter
%- hungover
%*@:- ( so hungover my head hurts
```

Some of them get a bit eclectic and take a bit of thinking about but all in all I found it an amusing book to browse through.

X.desktop Cookbook

by Michael Burgard with Mike Moore
Prentice Hall International (UK) Ltd
376pp., (Soft Cover)
ISBN 0-13-978537-X

Reviewed by
Mark White
National Centre for Studies in Travel and Tourism
<markw@ncstt.jcu.edu.au>

The proliferation of Unix systems and X Windows has strongly shifted the general perception of Unix as an applications platform, rather than just a scientist's playground. This book, co-written by a Senior Product Consultant

with IXI Limited (the developers of X.desktop) provides "an ideal companion to X.desktop, the most widely available desktop manager for Unix users", and despite this quite unabashed bias throughout the book's entirety, will actually be a valuable resource for X.desktop users.

The opening chapters introduce (and occasionally justify) the use of a desktop manager, in addition to the various window managers (e.g. Motif) available. Further chapters deal with the more advanced aspects of a desktop environment, such as X resources, designing icons, popup menus, writing help files and the system file structures.

There's also a chapter about the X.desktop API, to assist developers in integrating their applications. The final chapter describes a "smorgasbord" example using most of the concepts the book introduces - a nice touch for the more adventurous user who wants to "put it all together" in one example. The appendices also make a worthwhile reference.

Overall, I found the book to be well-constructed and laid out; applications users should be able to clearly progress through the book and improve their working desktop. For existing X.desktop users, or even those just interested in the product, this is a great book.

ISDN in Perspective

by Fred R. Goldstein
Addison-Wesley Publishing Co., Inc., 1992,
246pp., (Soft Cover)
ISBN 0-201-50016-7

Reviewed by
Scott Colwell
Labtam Australia Pty. Ltd.
<scott@labtam.oz.au>

This book has a reputation on the net for being an accurate and readable resource for *users* as opposed to the usual books on ISDN which are aimed at either developers or coursework use. It is this more down to earth aspect that the title of the book alludes to. The author also provides much of his own opinion, clearly marked with a set of spectacles in the left column making it simple to differentiate from the more objective material. However, as time goes on, it becomes harder for the reader to remember what was fact and what was opinion.

The book starts by trying to define what ISDN is, a very difficult task since it is different things to different people. Fred tries to cover his bases here by describing it in terms of appearance, functionality and historical perspective.

This is perhaps where the book misses in the reviewers opinion. He does not clearly make the point that ISDN is a *customer access* technology that simply provides a single interface to services that for the most part were pre-existing, if not all previously available to customers. The author does establish this point over the first three chapters but it is a pity that he does not come straight out and say it.

The book goes on to describe the feature set and architecture of ISDN in chapters 4 and 5, followed by a dissection of the ISDN layers in chapters 6 and 7. There is more detail in chapter 7 than most people will want but it is a good description for those who want to understand the mechanism of how calls are setup. The book finishes with chapters on "Broadband ISDN" and "ISDN in Practice" which cover the future evolution of the telecommunications network and the reality of ISDN in the present network.

The author has an OSI background and makes the occasional comment that some readers may disagree with but he is not an OSI pedant by any means. In fact, he makes plain a number of home truths at the expense of the OSI purists. The author also has a USA centric perspective but does cover how ISDN is implemented around the world. The only problem is his implication that ISDN is 'standard' to the extent of phones and terminal equipment being portable from country to country whereas in fact this is not the case. This is particularly odd since the USA has three incompatible ISDN standards of it's own!

This book should be on the reading list of anybody using ISDN as more than a cheaper version of a leased line. However, as the author suggests, there are other books that provide the sort of detail that developers or specialists will need. Still, this book provides much background for why things are the way they are, something that standards documents are not known for.

The Basics Book of X.25 Packet Switching Second Edition

Motorola University Press/Addison-Wesley
1992, 74 pages
ISBN 0-201-56369-X

Reviewed by
Rick Stevenson
Stallion Technologies
<rick@stallion.oz.au>

This book is one of the "Motorola Codex Basics Book Series", whose titles cover a variety of data communications and networking topics (ISDN, OSI and Frame Relay for example). As the title suggests, this is not a detailed treatment of X.25 packet switching, but an attempt to explain the essential concepts and provide an overview of the applicability and benefits of the technology. Authored by "a team of experts at Motorola Codex", it is a short, easily read volume, aimed at those who require an understanding of X.25 but don't wish to be buried in acronyms and CCITT X recommendations.

The first two chapters of the book describe, in relatively simple terms, the components of a packet switched data network and how these networks operate. The third chapter describes the relationship between the CCITT (or ITU-TSS as it is now known) and X.25 and contains the mandatory spiel on the OSI Reference Model. The three X.25 peer protocols get a more detailed treatment in this chapter, but this can be skipped by those interested only in the high-level view.

Chapters 4, 5 and 6 describe the type of applications for which X.25 is appropriate, the relative merits of private and public networks, and the potential benefits of an X.25 networking solution. The examples given are typically based around multi-drop SDLC networks, which may not be to everyone's taste. The final chapter gives a brief description of some present and future applications of packet switching.

The only real criticism I have to make of this book is that it appears to be a low-key advertisement for Motorola Codex products. There are a number of sidebars which have hand-drawn pictures of equipment with captions like "The Codex 6500 series is a high-performance family of packet data networking

products''. I didn't find this too obtrusive, but I do think it detracts a little from the credibility of the book.

In conclusion, I would recommend this book to those who want to acquire a basic level of understanding of X.25 packet switching, especially if time is limited. For someone making managerial decisions about incorporating packet switching technology into a traditional IBM-style network, I feel it could be quite valuable.

Inside SCO Unix

by Steve Glines, Peter S. Spicer,
Benjamin A. Hunsberger and
Karen Lynn White
New Riders Publishing,
ISBN 1-56205-028-1

Reviewed by
Taso Hatzl
<tasom@munniari.oz.au>

The authors state, "Inside SCO Unix is written for beginning and intermediate UNIX users who want clear and complete background information plus practical examples illustrating the essential components of UNIX".

The book's "Contents at a Glance" page should give the reader some idea as to whether the book is likely to be of any interest to them.

1. Getting Familiar with SCO Unix
2. The Organization of a SCO Unix System
3. Fundamentals for the First-Time User
4. Working with Unix Shells
5. Working with Files and Directories
6. Using Pipes, Filters, and Redirection
7. The Role of the System Administrator
8. Working with System Information
9. UNIX Daemons
10. Working with the Visual Editor
11. Using the Office Commands
12. UNIX UUCP Communications
13. Common Applications

14. Creating Useful Shell Scripts

15. Using Programming Tools: awk and sed
+ SCO Unix Command Reference

This book is a 700+ page broad brush commencing with some UNIX pre-history and ending with a 200+ page UNIX Command Reference. In between, the reader finds out how to log on, some useful information about shells, disc partitioning, file permissions, pipes, redirection, creating accounts, auditing, backups, system configuration, the init process, printing, batch jobs, vi, mail, UUCP, WordPerfect, Lotus, writing shell scripts, grep, awk, sed, lex and yacc, (phew!).

That's a lot of material so naturally, the depth of coverage is not great. Topics that are covered in tutorial fashion are satisfactory for the intended audience and also make a unique and positive contribution to the reader's knowledge of the subject. (Anyone who undertakes the task of explaining how to configure UUCP deserves some credit.) Vi, mail, UUCP, cron, shell programming get this treatment.

The book let's itself down in being just too fat with material which is so elementary that one suspects it was put in for padding, or which is to be found in the SCO Unix manuals. The book is certainly not unique in being overweight - just take a look at the computer section of any bookstore. I guess it caters for the consumer who picks up a volume, and by its sheer weight, judges it to contain all that he would ever need to know about a subject.

THE X WINDOW SYSTEM AND MOTIF : A FAST TRACK APPROACH

by Jan D. Newmarch
Addison-Wesley Publishing Company
ISBN 0-201-53931-4

Reviewed by
Michelle Ritchie
Andersen Consulting

This text has been written to assist those of us, like myself, who are dealing with X Windows for a first time. The text is just over 200 pages in length - not exactly what you would expect from a text dealing with such a complex subject ! It is refreshingly easy to read with clear explanations, simple relevant examples and associated screen

shots.

The text is divided into 3 key sections :

1. The X Environment
2. Xlib Programming
3. The Motif Toolkit

The author, Jan Newmarch (a lecturer at the University of Canberra), has taken a tutorial type of approach in writing this text with each section containing a set of related exercises to reinforce and expand upon the content discussed within the section.

The author begins with covering the history of X windows and introduces the relevant jargon (eg. Widgets, Intrinsics), object oriented concepts, the standard tools and twm and motif windows managers. For the remainder of the text the focus is on the Motif windows manager, selected for it's similar look-and-feel to the IBM Common User Access interface providing compatibility with Presentation Manager and Microsoft Windows.

The author proceeds to cover only the fundamentals of Xlib programming with and extension of Fonts, Colour, Icons and End- user Configuration/Control. By keeping the subject matter simple, yet realistic, the author avoids saturation and discouragement of the new X

Windows programmer. The aim of writing the text is not to provide an exhaustive technical manual for X windows programming, but rather to act as a guide providing the basic skills which can be expanded upon with experience.

The remainder of the text explains the object oriented approach to X and Motif programming and demonstrates how to build a simple, followed by a more complex, application. The applications and associated examples of code could be real-life examples which are available on the Internet.

Reading and understanding the contents of this text will develop a reader with no exposure to X or GUI programming into a programmer capable of developing a full application. The underlying assumption is that the serious programmer will need to delve into reference manuals and larger texts to implement a more complex application than outlined in this text.

I believe this text achieves the author's purpose, that is to provide the reader with the basic knowledge to allow him/her to productively work with the X windows system. It's simplicity, practical examples, and carefully chosen exercises are what makes this text a success and what prompts my recommendation to anyone who needs to program in the X windows environment for a first time.

Wizard's Bookshelf

Working at larger sites quickly demonstrates that it isn't sufficient to simply know about UNIX security — it is equally important that you have some breadth of knowledge too.

I have just finished a book that will hopefully provide some of this more general information on computing security - *Computer Security Basics*.†

This book brings much useful and interesting information together into one place. It starts with fairly simple topics (password protection, the differences between a virus and a worm), briefly describes secure system planning and administration, and then goes into some depth discussing security concepts as detailed in "The Orange Book" and other computer security sources.

These concepts include the alphabet soup of security ratings (D, C1, C2, B1, B2, B3, A1) and what they actually mean, encryption, network security, physical security, and the TEMPEST program. As they say in the book (about TEMPEST), "This is the real spook stuff!" - even the origin of the name TEMPEST has been classified for decades.

This information is followed by five appendices which comprise over one third of the book. Unfortunately, much of the information in these appendices is very US-centric (such as a description of computer security-relevant US legislation). Still, I derived some amusement from the list of acronyms — the US Government certainly thrives on them.

In all, an indispensable reference for anyone who is involved with computer security in any way, especially higher-security ("spook") environments. However, it isn't a book that you are likely to finish in one sitting (unless you are *very* interested in this field) — try as the authors might, it is still a large book covering a fairly dry area.

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

Open System Publications

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

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

For example:

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

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

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

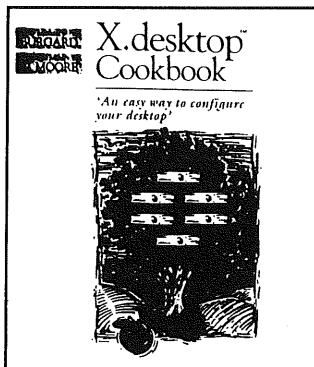
AUUG Inc.
Open System Publication Order
PO Box 366
Kensington, NSW, 2033
AUSTRALIA
Fax: (02) 332 4066

† *Computer Security Basics*
Deborah Russell & G. T. Gangemi Sr.
O'Reilly & Associates, Inc.

AUUG BOOK CLUB & PRENTICE HALL AUSTRALIA

20% DISCOUNT TO AUUG MEMBERS

Please send me a copy/copies of the following book —



Burgard/ The X Desktop Cookbook

ISBN: 013978537X Paper 1992 RRP \$83.50*

*Deduct 20% from listed retail price

Name: _____ Organisation: _____

Address: _____
(Street address only)

Telephone: _____

Please send my book/son 30-day approval (tick box)

Enclosed cheque for \$ _____ (Payable to 'Prentice Hall Australia')

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 2100

OR  Use our FAST PHONE SERVICE by calling Sandra Bendall.
SYDNEY (02) 939 1333

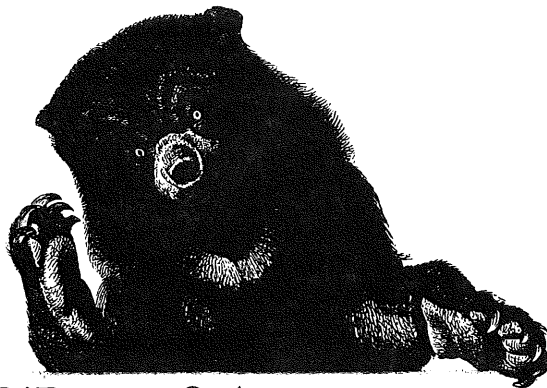
A.C.N. 000 383 406



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

A Paramount Communications Company

THE WORLD'S LEADING UNIX PUBLISHER



O'REILLY & ASSOCIATES, INC.

HAS MOVED...







& THINGS HAVE CHANGED!

O'Reilly and Associates are now represented in Australia by WoodsLane Pty Ltd.

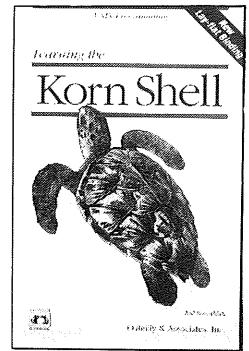
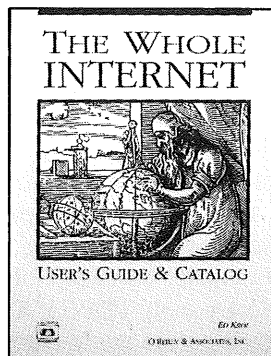
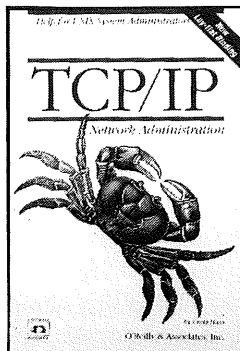
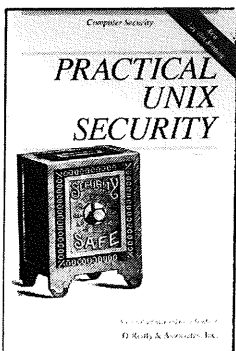
For those of you who have not heard of us, I would like to let you know how we do business. We airfreight all our books into Australia so you can expect new titles and backlist to be available constantly. We provide our booksellers with 48 hour turnaround, so if they happen to be out of stock they can access the books from us quickly. You will also find that we will promote the books to you so you will see regular advertising, information and reviews about O'Reilly & Associates books.

If you have any problems purchasing O'Reilly books, please call me, Tricia Wilden, at WoodsLane.

AVAILABLE FROM AUSTRALIA'S LEADING BOOKSELLERS

- | | |
|---|---|
|  NSW: Dymocks George St Sydney 224 0411
Dymocks Pocket & Technical
Wynard Sydney 224 0444
Hotline Chatswood Sydney 415 2265 |  QLD: Dymocks Edward St Brisbane (07) 220 0152 |
|  VIC: Tech Books Melbourne (03) 663 3951
McGills Melbourne (03) 602 5566
Dymocks Melbourne (03) 663 1344 |  SA: The Bookshelf Adelaide (08) 223 3899
Dymocks Adelaide (08) 223 5381 |
|  WA: Rellim Booksellers Perth (09) 321 5684
Boffin Bookshop Perth (09) 321 5755 |  A.C.T. Uni Co-Op Bkshp
ANU Canberra (06) 257 2673 |

Or call WoodsLane for the stockist nearest you.



Distributed by **WoodsLane Pty Ltd**

Phone: (02) 979 5944 Fax: (02) 997 3348 EMail: woods@tmx.mhs.oz.au

A.C.N. 003 667 549

The Electronic Interviews

This marks what will hopefully be the first of a series of interviews conducted by electronic mail with well-known people in the Australian UNIX arena.

These interviews are the start of a process which will hopefully culminate in a significant ‘Twenty years of UNIX in Australia’ work (targeted for completion during 1994).

The interview series starts with **Greg Rose**, who first began using UNIX in 1974.

When and what was your first involvement with Unix?

It was in late 1974, at the University of New South Wales, just after final exams. Chris Maltby and I (just finished first year) were wandering around putting bigger and better jobs into the third-year Comp Sci card drawers (Fortran and Algol on the IBM/360) when one of the Honours students (Jenny Something) asked if we wanted to see this new operating system. It was clear as mud at the time. I remember that I thought C was PL/1 because that was the only thing I’d ever seen with the /* comment convention */.

During that break, the IBM/360 was replaced with a Cyber 72 (which was already operational, it was just the changeover). The PDP-11 was one of twelve bought to function as remote batch entry stations for the Cyber. Of these, four were PDP-11/40s with some real capability, and the aim (of the Computer Services Unit) was to run RSX-11D on them, with some software which enabled them to emulate CDC’s U-200 RBT protocol. The Computer Science Department thought that they’d be able to run Unix on theirs. But it still had to function as a remote job entry station too.

A bunch of honours and postgrad students dived in and wrote device drivers for the network, printers and things, a queueing system, a better spooler, and so on. The spooler printed ‘JIGI JIGI’ on the paper perforation, as a spoof on the Computer Services Unit. This was the initials of John Wainright, Ian Johnstone, Greg James, and Ian Hayes.

As well as the desired functionality for remote batch operations, CS wanted to have a batch assembler service for teaching the second year course in second session. But all the Honours students were busy getting their degrees, so Chris and I got the task of writing the local assembler batch system, complete with a run-time library and core dump printer. Other promising second and third year students were drafted for other aspects of the task, and eventually we all became de-facto operators of the system. In second session, 1975, we had the dubious honour of being asked to submit assembler assignments using the system we had written a few months previously. At least by this time it was easy, and of course we had left in a provision to jump the batch queue.

Did you like the look of Unix, and so started doing these developments, or was it the other way around?

Well, Unix looked better, and was more ‘hands on’ than Kronos (the Cyber operating system). Also, the problems I was being given in the courses were much simpler than the problems I could choose for myself or have thrust upon me on Unix. This was also true for the more advanced people, since most of them had never had a chance to fiddle with machines at the hardware level before.

Was this hive of activity the catalyst for the formation of AUUG?

Basically, yes. Interest in using Unix at the more capable batch stations increased, mainly because the remote batch emulator written for Unix used a back door approach to the protocols (which assumed that operators would type commands to make things happen -- the U200 device driver typed these commands automatically and very often...). The Computer Services Unit itself started to use Unix, the Australian Graduate School of Management got a very early PDP-11/70, Sydney University started to use it, and The University of Wollongong started their port to the Interdata 7/32. Because of the rapid development, we were having meetings about weekly to exchange ideas and problems. Somewhere along the line, we started calling these meetings ‘AUUG’. John Lions was editing a UNSW internal newsletter called

“CUSwords” (Computer Users Society I think), and this didn’t take too long before it started promoting Unix. There were plenty of criticisms to level at Kronos, and it was getting boring until Unix came along.

What can you tell me about the formation of AUUG?

The real formation of AUUG started, I suppose, when we tried inviting people to speak as they passed through the country. This justified the use of bigger meeting rooms, a biscuit budget, and open notification of the meetings. My memory of exactly what the timing was is a bit hazy, as I was busy with the course load of a third year student, doing operations and programming on the Computer Science 11/40, and working 4-midnight as an IBM 360 operator to eat. Basically this transition happened in late ’75 and early ’76, I think.

So there isn’t a specific date you could give for the formation of AUUG (or even of their first meeting)?

Not that I can give, no. I wouldn’t be surprised if John Lions could tell you though.

Do you have any interesting recollections of those first meetings?

Yes, but I can’t tell you :-). Seriously, though, it was a time of excitement and intense activity, which often led to frayed nerves and short tempers. One of my most vivid memories was when Dave Horsfall, then with the Computer Services Unit, introduced a parallel set of system calls which operated on inode numbers rather than file names, and as a regrettable side effect bypassed all of the intervening directories’ permission bits. The term “necro-nurd” echoed down the corridors, coined by Chris Maltby.

Then there was the time I was giving a presentation for my Honours degree about “Why Unix is too slow, and what we plan to do about it”. At the time I was cycling to the University, and this particular day there was a really big thunderstorm, and it was hot and humid, so I stayed late. The only thing to drink in the AGSM [Australian Graduate School of Management - asb] that evening was a flagon of cheap wine, so at 11:00 the next morning I presented a talk about “...what we did about it”. The interesting thing is that Chris Maltby, Ian Johnstone and I had basically put hashing and virtual memory buffers into the Unix kernel that night, and I was talking about it while drunk (it wasn’t even a hangover, yet) and having had no sleep for 36 hours. Professor Allen said it was the best presentation he’d heard from me...

Can you name some major names (and their role) in AUUG at the time?

I’ve mentioned JIGI, John Lions and Chris Maltby already. Another major player at the department of Computer Science was Peter Ivanov, who has been in charge of their computers almost continuously since those days. Andrew Hume was a year behind Chris and I at UNSW, but was extremely precocious. Dave Horsfall at the Computer Services Unit was a bastion of Unix support in an otherwise hostile environment, even if we didn’t agree with some of the things he did. Piers Lauder from Sydney Uni. Juris Reinfelds, professor at Wollongong University, was probably the first person to really believe in a future for Unix, and he made sure that Richard Miller, an expatriot Canadian, was pointed in that direction; the result was the amazing port of Unix. Others at Wollongong were Ross Nealon (sadly deceased from leukemia) and Tony McGrath.

It was a while before we really heard much from the Victorians, but very early on Robert Elz at Melbourne University and Ken McDonnell at Monash picked up on the Wollongong software and started to be very active. We were always expecting another talk about “tty drivers” from Robert Elz.

I’m probably leaving out a bunch of worthy people, but by that time (say 1977-78) the group was growing rapidly and I was trying to get a degree and get married.

Who were AUUG’s first international speakers?

The very first I think was Jeff Rottman, a most amazing guy who happened to be passing by and had

made a name for himself in the Unix community. The next I can remember was when Dennis Ritchie came out in 1978 to speak at a programming languages conference. He and Niklaus Wirth were pitted against each other -- the conference even had a humorous Pascal versus C debate (in which I spoke last for Pascal!). There's a bit of a gap then, and the next I remember was Bill Joy in 1980 or 81. Bill Joy was a study in arrogance. He hadn't bothered to arrange a passport, let alone a visa! He brought with him a first release Sun, with no media except a fixed Winchester disk, with the system already loaded on it. He gave it to a hardware engineer to set up (while he waited for the diplomats to sort out his situation), and the engineer's first action was to run diagnostics on it, including formatting the disk, so the demonstrations were a bit poor. Since that time, there's been a fairly regular stream.

Could you describe your professional career path from when you got your degree? What would be the main highlight(s) of your career to date?

I finished the computer science honours degree at the end of 1977, and without a clear decision to do so, ended up enrolled in a Ph.D. program. I'd been working in the computer industry at least part time since 1975, and managed to rationalise that into a part time tutoring position at UNSW. The PhD petered out at the end of '78, although I didn't notice until mid-'79.

John O'Brien had joined a small software company, and was both enjoying himself and making money, so I interviewed with his company and joined for about four months. Then the two directors of Link Computer Systems had a falling out. John and I had been programming mostly in RPL (Rapid Programming Language, an implementation of FileTab), but for some tasks wanted to use C. Link had organised a license for Whitesmith's C compiler, and we were about to try and negotiate to get distribution rights for that and their Unix look-alike system Idris, when we were told by the remaining director to "get back to work". Within a week, John and I had joined with the other director, Allan Moore, and formed Fawnray Pty Ltd, to pursue the Whitesmiths opportunity.

I like to think of Fawnray as one of the successes, even though it eventually went out backwards. We established a comfortable business and a fun place to work. In 1983 we were asked to look at a project to port Unix to the ELXSI, a 64-bit multiprocessor computer that was way ahead of its time. We came up with a project plan, but initially lost the job to a Silicon Valley based company (coincidentally with an Australian name, but that is another story). The ELXSI had its own operating system already, called EMBOS, and the job given to the other company was to host Unix on top of EMBOS. On the day their first milestone was due -- they were to demonstrate the shell running "ls" -- ELXSI got a phone call asking for documentation of EMBOS's "fork" system call. It didn't have one. In two weeks I was over in Silicon Valley planning the porting project. We eventually made a very successful port of Unix by carving EMBOS into a sort of virtual machine operating system and then hosting the rest of it in parallel with a port of real Unix on top of that. It was a lot like the architecture of MACH today.

Some of the things we were doing attracted the attention of venture capitalists such as the Australian Industry Development Corporation, who felt that it was a good time to create a general purpose Unix software house. During late '84 and '85, Fawnray grew, merged to become Fawnray Prance Ltd, changed its name to Neology, expanded by a factor of six or seven, and exploded. Getting that venture capital was the worst thing that ever happened to us, in retrospect. The core business, Unix-related consulting, was still going fine though.

John O'Brien took over the Whitesmiths related products and consulting, and Whitesmiths Australia Ltd has far outlasted Whitesmiths Ltd. Chris Maltby, who I finally enticed away from Sydney University in 1984, and I formed Softway in early 1986 and pursued a Unix porting project with the CSIRO, funded by Techway (who represented ELXSI). Softway from the start concentrated on what we did well, and aimed to establish a significant reputation in a specialised area; I think we succeeded. We had numerous contracts in the United States, mostly helping companies in their last ditch efforts to resurrect minicomputer products in the face of competition from cheap microprocessors and first generation RISC machines. They didn't succeed, but that wasn't our fault.

In 1990, I realised that I was getting stale, since I'd been doing essentially the same thing for ten years. Anyway, I was getting to be a cranky old man who had a company that no longer needed him. So I left

Softway and found a sabbatical position that happened to be at IBM Research at Yorktown Heights, New York. To me, being a programmer with no staff to oversee and no product to develop was a thirteen month holiday; it paid about as well too, since the pay assumed that I had an academic position also paying me. I learned an enormous amount during that time, and if the job was the only consideration I'd be there now. Professionally, what I achieved was to help build a successful, real world clustered computing facility out of workstations.

Instead of staying, I stuck to the plan and returned to Australia where in early 1992 I joined the Australian Computing and Communications Institute. Not much to say about that yet.

You are actively involved in several groups such as AUUG, SAGE-AU†, and USENIX. What do you see as the biggest positives and negatives of these groups?

To handle them individually:

AUUG's biggest positive is fundamentally linked to its biggest negative. Basically, I don't think that UNIX (note the change in capitalisation) users need a group. The less seen of UNIX the better for the vast bulk of users. So the group really serves MIS in large organisations, computer hardware and software vendors, and the UNIX traditional techies. Its biggest advantage is that it is a single organisation serving all three of these groups, which allows for some synergy and interaction that is not present in any of the other long established groups to my knowledge. But this is also its disadvantage, as it puts a lot of strain on the organisation to avoid the natural tendency to bigotry and overservicing one segment at the expense of the others. As for myself, with a leaning to the technical side of the industry, I must say that I'm looking elsewhere. UNIX is not the correct focus for development any more. (That's a hint for the next question.)

SAGE-AU (and I'm somewhat involved in SAGE-US as well) represents both what I'm personally most interested in, and also what I believe to be the cutting edge of the technology. We now have a reasonable handle on how to write compilers, build GUIs, and stuff like that, but we still have no real idea how to manage clusters of thousands of computers which have aggregated over time, and are almost never the same as each other. System administration costs are increasing, not decreasing, and particularly with reference to the costs of the things being administered such as computers and networks. The hardest problem facing SAGE-AU is how to embrace the leading edge of these technologies to solve its own problems, and still meet its primary goal of servicing the community of existing-practice system administrators with their isolated LANs. I don't know the answer to that problem, but I expect to spend the next few years contributing to it.

USENIX is in an interesting state at the moment. It is an extremely well recognised forum for presenting new technical developments in and around the Open Systems arena. Increasingly it is turning to specialised conferences and workshops to meet specialised needs. I think that the "focus on focussing" is leading to a starvation of people presenting new developments at the main conference, which can form a vicious cycle. I hope to get more involved in USENIX over the next few years, workload permitting.

What do you see yourself doing in five years' time? Ten years?

Telecommuting is working well enough for me at the moment, that I think in ten years time I won't have to leave my home in Mudgee hardly at all. How I'm going to get to that position (ie. having a home in Mudgee or somewhere) I have no idea. I just hope it will continue to be interesting.

† SAGE is a relatively new group, the "Systems Administrators Guild", formed by USENIX. SAGE-AU is an independent group formed in Australia, with similar aims.

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

Ahhh, the general purpose soapbox question! Well, I think that it is time for a new operating system to emerge and supplant Unix, but I certainly don't think Windows-NT (or OS/2 :-) is it. Unix development has reached the point of diminishing returns, as the system is now way too large for any coherent development.

That is not to say that the new system (whatever it is) won't still feel somewhat like Unix. The operating system interface standardised by POSIX is pretty good really, and any future system, to be successful, will have to have an interface just like that (or at least capable of being disguised to support that). So applications should port over, or simply install, very easily. That is a necessary criterion.

Also any future operating system will need to really be integrated with the networks, in a much more serious sense than Unix, Netware and so on are. As an example (borrowed from Plan 9 I admit) there really shouldn't be any distinction between files accessed over the network, and a local file system; in fact, there shouldn't be a local file system at all, even if it just happens to reside on the same computer. Diskless workstations are a hack to simulate this fundamental change in behaviour.

We need to understand authentication a lot better. This has to be cryptography based, and there is a lot of work going into figuring out just how to identify a user from a remote administration domain.

In any multi-computer site (like many homes in a few years...) there will need to be a movement towards recentralising the administration. One of my nightmares is being asked to integrate ten random workstations which have been administered by their individual owners, into a coherent facility. That is very much harder than planning to support a hundred users (even with the same set of random hardware) when you can absolutely design the setup. Cost effectiveness will force this change in anything but the smallest sites.

Note that I very carefully said "recentralise the *administration*", but I didn't say anything about "control". Too often the authority to determine *how* a machine, user or application package gets into the system somehow gets confused with the authority to determine *why* or *whether* it gets done. This is not acceptable. We need to learn more about the social impact of these larger and better connected systems, and somehow separate the functions of setting policy, administering within the policy, and enforcing it. At the moment, policy setting and enforcement on the Internet is anarchy and peer pressure, and it is working much better than I would have dared to hope, but I am not convinced that this will continue to be the case.

Thanks for the interview. It's been fun.

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

The following paper is re-printed from AUUGN Volume 1 Number 4. Greg, is this the report that you presented that morning? (see The Electronic Interviews in this issue.)

A brief note on UNIX System Performance.

30th May 1979

Recently at UNSW a concerted effort has been made to obtain the the best results from the PDP11/70 in Computer Science. Rather than approach this problem in the usual ad-hoc way it was decided that a systematic approach should be used. A novel idea, you might say.

With the help of a tame electrical engineer a programmable clock to suit our needs was designed and built (at a cost of \$100 dollars approx). This clock is capable of interrupting the CPU at a rate settable between 500ns and 250ms.

Software for the system was written which uses interrupts at a rate of 317HZ. When an interrupt occurs, information describing the state of the machine is saved in a large buffer. Noting the program counter at interrupt time allows a profile of the system's execution to be taken, also time spent at each hardware priority level was recorded. The cost of these changes is less than 1% of CPU time plus a large chunk (10%) of our available memory. The memory chunk is of a variable size, depending on how detailed the profile is to be. In the system we used, one long word (32 bits) was used for every four bytes of instructions. This allowed identification of individual C procedure usage since each begins with a four byte instruction and thus its entry point would have a unique entry in the profile table.

Besides the system code changes, there are only two other programs required to make use of the profiling system. The programs 'suck' and 'examine' are used to extract the profiling information from the system, and analyse the results.

We began by profiling the system that was currently in use. It was a mapped buffers system, with 96 buffers, 170 processes, and 400 inodes. The system in this configuration reached saturation (subjectively) with 42 users logged on. The initial results gave us immediate hope. Over periods of up to several hours in the busiest time of the day, kernel cpu time was seen to be 85% [all percentages given are approximate, full and complete results are in the pipeline] of the time available, with no idle time. Of the time spent in the kernel mode, 20% went in servicing output interrupts on the terminal multiplexors (7 DZ11's), 15% was used by the routine 'swtch', the rescheduler, and a further 12% was absorbed by procedure call overhead.

The immediate thrust was to reduce the time taken to reschedule by chaining ready-to-run processes together, (as in PWB/UNIX). This had the effect of reducing the time spent in 'swtch' to about 1%. As a result, the proportion of kernel time declined to 75%.

The 'dxxint' routine was the next target. By recoding to use an external synchronization (a KW11-P), the time spent in this routine was reduced to 13% of the kernel time, which now fell to about 70% of total time. This effort showed that although the DZ11 apparently is an acceptable multiplexor it is in fact far from it.

The routines which now seemed to be using most time were: dxxint 13%; procedure overhead 14%; getblk 10%; copyseg 6%; and wakeup 3%. A change to getblk, the buffer pool managing routine, which hashed the 'dev', 'blkno' for each buffer in the pool, with a hash table size of 113, reduced the time consumed in getblk to 1% of kernel time. In conjunction with this, a change was made to copyseg which halved the time spent in this routine. As a result of these two changes, kernel time dropped to an all time low of 65%.

A further reduction without a loss of service was gained by reducing overheads on certain interrupts by not rescheduling after interrupt processing was complete.

Subjectively, the system now seemed to reach saturation (still) with an extra five users, or at around 47. These results are probably close to the limit that can be achieved by this technique. Changes are still in the air for the DZ11 driver, as are slight improvements to wakeup (by hashing). It seems unlikely that any improvement can be made to the register save routines 'csv' and 'cret', except by perhaps getting at the 11/70 microcode. Neither is the effect of the UNIBUS disk controller being accounted for. Due to the design of the 11/70 memory system, UNIBUS disk I/O can degrade the speed of the CPU, even with maximum cache hit rates, by upto 50%. Under normal running we calculate the degradation from this source to be of the order 20%. The only practical solution to this problem requires a considerable capital outlay (the purchase of MASSBUS drives).

Ian Johnstone
Chris Maltby
Greg Rose

The First Port is the Deepest.

Tony McGrath

Uniq Professional Services Pty. Ltd.

ABSTRACT

There are many events that happen to each of us that help define exactly who we are and why we are the way we are. For me, one of those seminal events took place fifteen years ago while I was an undergraduate student at the University of Wollongong, and the event was the first UNIX¹ port. Looking back over those fifteen years, it has become difficult to accept just what it was that I was involved in, and what effect it would have on myself and the whole UNIX world. This is a short, somewhat biased look at those events, and the people that made them happen.

In the Beginning...

There was UNIX, the child of Ken Thompson, one of the many children that Ken would give the world. It ran on a PDP-7, one of the processor children that Digital Equipment Corporation would give the world. And behold, it was very good.

As time went by, UNIX grew and grew, until it could no longer be supported properly by the humble PDP-7. Yea, verily, it was rewritten in the programming language C, child of Denis Ritchie, and moved to the PDP-11.

And there was much rejoicing.

Where were you in '75?

1975 was an interesting year in many ways. Edition 6 of UNIX became readily available, the University of Wollongong was created from a college of the University of New South Wales, and I was a Year 11 student² who was falling deeply in love with primitive computers.

With the creation of the new university came the creation of the Department of Computing Science under the leadership of Juris Reinfelds. Juris had the interesting task

¹UNIX is a trade mark of a number of different organisations that have been responsible for it over the last 20 years. You all know who you are.

²How is your arithmetic? Don't bother, it makes me 34.

of getting the fledgling department staffed, creating new courses, and buying the initial computing resources for its students. The university already owned a UNIVAC 1106 mainframe computer, quite a powerful machine in its day, but Juris perceived that the department should have its own computing facilities.

Juris had looked at a number of systems, and he was very impressed with UNIX running on the PDP-11, but he had one small problem: the department could not afford a PDP-11 system that would provide enough power for their needs. What the department could afford was an Interdata 7/32 system, a 32 bit minicomputer that was designed around the IBM 370 architecture, and it didn't run UNIX at all.

One of the people that Juris had brought with him to Wollongong was Richard Miller, a Canadian that had worked with Juris on a number of projects. Richard put forward an interesting proposal, porting UNIX to the Interdata system. This would not be considered a difficult activity today, but in 1975 there were no existing serious operating systems that had been moved from one computing architecture to another without significant resources being required, and Richard was basically proposing that he would do this work alone. In fact, the hurdles that needed to be overcome were significantly more than simple porting, as we will see.

The Interdata 7/32

Before we get too involved in the mechanics of porting UNIX, it would be worthwhile to spend a little time looking at the Interdata 7/32 minicomputer. This was a machine specifically designed to be familiar to anybody who had used any IBM 370 system, and its machine language was similar to the IBM 370. In its particular incarnation at Wollongong the system comprised a main memory of 256 Kb and a dual fixed/removable disk subsystem with a total disk capacity of 20 Mb, 10 Mb of which was removable.

The operating system for this machine was called OS/MT-32, a partitioned multi tasking operating system that supported a number of fixed memory partitions that could be allocated to running processes, or shared by a number of processes if required. Processes were allocated to partitions using a command processing language that used the system console directly. Terminals could also be attached to running processes from the system console, allowing interactive programs to be used by a number of simultaneous users. For student use, the system typically ran a version of BASIC that contained a single executable copy in one partition with separate data partitions allocated for each user.

Despite the cumbersome memory organisation, OS/MT-32 provided a good set of services and was quite an advanced system of its type. Unfortunately, the design of the command interface assumed that all commands were entered from the system console and that interactive programs were started from the console with an appropriate terminal to use for interactive work. This type of arrangement was perfectly fine for an organisation that had a fixed set of programs that were dedicated to particular terminals, but it was very difficult for true interactive computing.

In terms of processing capacity, the 7/32 could be compared to the PDP 11/40. The architecture was inefficient compared to the PDP 11 architecture, so programs required more resources to perform similar work.

MOTH

It is important to remember that the original purpose of the 7/32 system at Wollongong was to provide a computing facility for the Department of Computing Science, and that the UNIX port was considered to be secondary to this need. It was also obvious to anybody who had attempted to use OS/MT that it did not provide the interactive facilities that would allow the system to be used to its full capacity.

With this in mind, the first steps that Richard Miller undertook was to get a better understanding of the 7/32 environment as a prelude to the porting effort. He decided to design and implement an interactive command system that allowed users to dynamically execute programs without the need to use the complex system console command suite. This system was called MOTH (Miller's Own Terminal Handler) and it provided a simple command shell that could load and execute programs directly from an interactive terminal. This system provided Richard with valuable insights into the 7/32 system, especially with OS/MT, and it set the pattern for how the UNIX port would be undertaken.

MOTH became an integral part of the Department's computing strategy, as it allowed the system to be used by inexperienced student users and it also provided a framework for the development work needed to support the UNIX port, and for more than 2 years was the standard operating environment.

Getting Started

All the pieces were there: the 7/32, MOTH, an Edition 6 source licence, and Richard. Well, maybe some of the pieces were more there than others, and the major source of consternation was immediately obvious.

When costing the 7/32 system, it was realised that it would be prohibitively expensive to configure the system with a reel-to-reel tape drive, and it was impossible to borrow a tape drive from any other source. Unfortunately, all the UNIX sources were only available on tape or PDP 11 disk packs, a small difficulty that was going to make life interesting as time went by. There was one other important thing, the nearest tape drive equipped 7/32 system was in Interdata's Sydney office, 80 km away.

Tapes could, however, be read on the UNIVAC 1106 system, and at a later point the university purchased an 8/16 system to act as a terminal front end for the 1106. Once this system was installed, tapes could be read on the 1106, the data transferred to the 8/16 and written onto OS/MT formatted disk packs.

Once a mechanism was discovered to allow data to be transferred there was still one major obstacle to be overcome before UNIX could be ported, the C compiler.

The Edition 6 C compiler was written for the PDP 11 architecture, produced PDP 11 assembler as output, and only ran under UNIX. Richard needed to have a C compiler that produced Interdata assembler as output and had to run under OS/MT.

The approach Richard used was very simple: he looked closely at the C compiler and made modifications to support the new architecture. These modifications were then transported to the University of New South Wales which had the nearest UNIX system to Wollongong. The modified compiler was then debugged under UNIX and then used to compile itself, producing the Interdata C compiler in Interdata assembler. This assembler was then moved back to Wollongong and used as the template for the OS/MT compiler. It is a credit to the abilities of Richard Miller that he only needed to perform 2 debugging sessions of this type before he had a passable compiler that ran under OS/MT with sufficient power to compile itself.

With a working C compiler running under OS/MT, he then ported some other basic software, such as ed, to make up the basic porting environment.

On The Way

Faced with the problem of getting UNIX to run on a totally different architecture some time was needed to look at the approaches that could be taken. Basically, there are two ways to get a new operating system running on a new architecture - write the whole system from scratch or implement the system on top of an existing system. The first method, writing from scratch, works very well if you have a development system and a target system that are physically close together so that the development system can be used for cross-compiling and other uses. Unfortunately, the nearest UNIX system that could be used for development work was 80 km from Wollongong, and the fastest modems that were available at that time were 300 baud or 1200 baud.

Faced with these facts, it was impossible to consider the full implementation as the first target. The initial target was to develop a UNIX kernel that ran as a privileged task under OS/MT and that used OS/MT's device drivers to perform I/O instead. This approach had a number of benefits: it allowed development to move faster in the single system environment, the OS/MT system allowed greater control over debugging, and less time was required to develop device driver interfaces.

The resultant system used a modified Edition 6 UNIX kernel which used OS/MT files as disk devices. This configuration allowed a complete implementation of the Edition 6 file system to be used. Hooks were provided within the kernel to access normal OS/MT files from within UNIX and other hooks allowed OS/MT tasks to be executed from UNIX as required. The major impetus for the latter facility was the need to use the OS/MT assembler since there was little time to write a new assembler, and the need to support the BASIC interpreter which was written in Interdata assembler and would only run under OS/MT.

It was about two thirds of the way through this development that Richard Miller made a trip to Canada and the United States, during which he made a visit to the Bell Research Laboratories to see Ritchie and Thompson. It appears that what he had to say impressed them sufficiently that a short time later the Research Group purchased an Interdata 8/32, the next larger model after the 7/32, and started a porting project of their own. They had the advantage of having their 8/32 next to their PDP 11/45, so they used the "scratch" method for their port.

Since the Research Group knew that they would be porting to a full 32 bit architecture, some effort was made to clean up various UNIX sources. The C compiler was extended to support 32 arithmetic and the kernel was rewritten to make it more portable. The final result of this endeavour was Edition 7, arguable the best UNIX kernel that was generally available.

Touchdown

The modified UNIX system was essentially completed late in 1977 and the day it was booted it had four users: Richard Miller, Juris Reinfelds, Ross Nealon and Tony McGrath.

Ross and myself were undergraduate students who were doing a Computing Science degree, Ross in his third year and myself in first year. We had been performing a number of tasks as 'defacto' system support personnel for some time and we were allowed to help Richard by porting a number of system utilities over to the new system. This was not a trivial exercise as many of the utilities had been written to take advantage of the fact that UNIX was designed for the PDP 11 architecture and no attempt had been made to write code portably.

I paid the price for my enthusiasm as Richard decided to give me *nroff* to port first. Unfortunately, *nroff* was originally written in PDP 11 assembler and had been translated into C in a hasty manner without any comments to guide one through it. This is not recommended as the first major piece of C code for a novice to understand, especially when one is attempting to port the code to a totally different architecture. I found the exercise daunting and, ultimately, uncompletable, but I did learn a lot about C.

Once the UNIX-OS/MT environment was available, work was begun to allow UNIX to run as the native operating system, as task that was simpler once some form of UNIX environment was available for development. It was in early 1978 that the native UNIX system was finally made available for general use at Wollongong.

Later in 1978, Edition 7 was released, and Wollongong received their copy hand delivered by Dennis Ritchie himself. It was quickly ported to the 7/32 and became the basis of what became the commercial data UNIX system.

Aftermath

All good things must pass, and so it was with the group at Wollongong. Richard Miller decided to move back to Canada where he worked on UNIX related projects for a number of years. He is no longer interested in UNIX any more, a comment that also holds for members of the Bell UNIX Research Group.

Juris Reinfelds continued to run the Department of Computing Science for a number of years, but he eventually moved to the United States, having spent a considerable amount of time trying to convince others of the future of UNIX, a future that has actually happened much as he predicted.

Ross Nealon took over from Richard Miller as the major UNIX development person at Wollongong, but his long fight with cancer was lost a few years ago, a very sad event to those who knew and worked with him.

I am still active in the Australian UNIX scene, the only member this group that is still working closely with UNIX. I am still amazed by what we did fifteen years ago, and I doubt that I can predict what the next fifteen years will bring, but I am sure that UNIX will still be there, either cursed or praised.

386BSD in the Shrink-Wrap World *

Peter J Billam

Treasury, and Department of Premier and Cabinet, Hobart

ABSTRACT

386BSD offers to the commercial user significant advantages over traditional LAN products: low cost, legality, connectivity, and routable networking protocols. It also has various resources which can be used to increase ease of administration and security, resources which are less available to users of bought software. The author also discusses an administerable user-friendly Unix login shell, and recommends system servers to improve the administerability of networked computers.

Disclaimer

The opinions herein expressed are those of the author only, and are not necessarily the opinions, let alone the policy, of his employers. Some of the features described have been fully implemented, some await complete implementation.

A very typical network

We have about 50 PCs, 50 Macs and a few terminals, and 7 Unix boxes. The PCs and Macs run mostly Word and Excel. This, I imagine, is a very common situation, and I expect most sites with this kind of setup would be running some LAN protocol such as Novell do perform their file-serving. One feature of our setup is that we don't have a separate LAN product, we use NFS for the PCs and the Columbia AppleTalk Package for the Macs. Another unusual point is that although two of our Unix boxes are sVr4 name brands, the other five, and most of the file-serving capacity, are generic 486 PCs running 386BSD, the public domain implementation of BSD Unix stemming from Bill and Lynne Jolitz.

Login in, but not to the command line.

As Unix makes its long march from the academic to the commercial culture, one of the major differences it must cope with is the fact that commercial users, by and large, do not want to come to the command line; they want a login shell that is an application, or, more generally, is a choice of the available applications and utilities.

To proceed further it is necessary to introduce *choose*, a public domain utility written by the author, which allows the user to choose one or more items from a list. *Choose* is a filter just like *grep*, except that in *grep* the choice is pre-determined by the shell programmer, whereas in *choose* the choice is made interactively by the user, using arrow keys and return, the space bar to mark and "q" to quit. *Choose* also maintains the last choice made in files in the directory *\$HOME/choices/*. Versions of *choose* currently run on *curses* and *DOS*, and I am working to extend it to *X*.

In order to be able to offer the user the available choice in an administerable way, we have given most of our users */usr/local/bin/menush* (†) as their login shell. This involves the following executable shell scripts:

(†) login shells that do not end in "sh" are unkind to *su*.

* This paper was presented at the SAGE-AU conference.

/usr/local/bin/mainmenu

is a script which outputs (one to a line) the options that user is currently allowed on this computer. For example, "Internet Mail", "Directory Services", "Text Processing", "About ...", or "Change Preferences". *Mainmenu* therefore centralises, into one easily administered and audited file, all questions of which login users are allowed to perform which functions.

/usr/local/bin/menush

is the main login shell for our users; it uses *mainmenu* piped into *choose* to determine what the user wishes to do, performs logging for accounting purposes as desired, and then runs the appropriate software. For example: line right ; line left line right 5.2; arc; line up 2; arc line left 5.2; arc; line down 2; arc

Host: adminbox.tres.tas.gov.au User: oliveo pjb

Office Automation	Database Application	Talk
Internet Mail	Send Internet Mail	Internet News
Internet File Transfer	Internet Remote Login	Check Load on Adminbox
Send Documents	Receive Documents	File Manager
Chess	Squares	Change Preferences
About ...	Contact Administrator	Directory Services

Use: up, down, left, right, mark, reject, accept
Or: k, j, h, l, space, q, ^C, ^Z, return

/usr/local/bin/telnethosts

is a script which outputs those *telnet* hosts that the user is allowed to access in pursuance of their duties, and pleasures, if any.

/usr/local/bin/easy_telnet

uses *telnethosts* piped into *choose* and *telnet*'s to the result.

/usr/local/bin/ftphosts

is likewise a script which outputs those *ftp* hosts that the user is allowed to access.

/usr/local/bin/easy_ftp

uses *ftphosts* piped into *choose* and *ftp*'s to the result, logging in correctly as *anonymous*.

/usr/local/bin/preferences

is a script which outputs those Preferences that the user may usefully change; typically "Password", "Printer", "Editor", etc.

/usr/local/bin/change_preferences

uses *preferences* piped into *choose* and reacts accordingly, for example running *passwd*, offering a choice of printers or editors, editing *\$HOME/.nn/init*, etc. This script should be dotted (sourced) so that it can set *LPDEST* and *EDITOR*.

The files *mainmenu*, *telnethosts*, *ftphosts* and *preferences* are shell scripts rather than the more conventional passive text files because this is far more flexible; they can consult text files, but can also react to the presence or absence of various files, the time of day, who is logged in, etc.

Also useful are the following directories:

/usr/local/allow

containing files (or links) like *cron*, *telnet-in*, *telnet-out*, *crypt*, *mail-out*, etc.

/usr/local/about

containing text files like *This Computer*, *This Network*, *Internet Mail*, *How to Set up your PC*, *How to Set up Your Mac*, *Smileys* etc.

The 386BSD Fileservers.

We started off running NFS and CAP on the ICL DRS6000, but the time came when we needed more disc, and found out it would be cheaper to buy 2 Gigabytes of disc and a 486PC than to buy 660 Megabytes of disc for the DRS6000.

Our stock 386BSD PC is a 486/33 with AMI Bios, 16 Mb memory, ET-4000 1 Mb video card, 15 inch NI monitor, 3-button mouse, WD-8013 Ethernet card, Adaptec 1542C SCSI card, external SCSI terminator, and two separate 1 Gb Seagate ST1120N discs.

Normally, these two discs are arranged so that one is used to back up the other; thus if one disc were to fail, restoring would simply be a matter of removing the failed disc for repair and possibly changing the SCSI ID of the remaining disc. This means the file-servers normally work as 1 Gb file-servers with built-in backup. It's also a flexible system, in that if something like an Exabyte is purchased and stuck on top of the PC, the second disc can be relieved of its backup duties and deployed to give a 2 Gb file-server.

If you want to run 386BSD, you'll probably have to specify hardware with that in mind; the chances of getting it running on a random desktop PC are not so good. We have had trouble with Quadtel BIOS and Cyrix chipsets.

On the software side, I come from a more SysV background, and have problems with *BSD curses* and *regexp*; but overall we have had good experiences with 386BSD, and find it to be as robust as the sysVr4 that we run. In particular, when running *NFS*, *AUFS*, *sendmail*, *news* and *X*, it seems to have no problems.

Our next step with the file-servers will probably be to fit a second Ethernet card and configure them as routers between the workgroup and the backbone.

CAP for the Macs

The ports of the public domain CAP software to 386BSD, giving Macs very economical file-sharing and printer-sharing, and also the port of CAP to the ICL DRS6000, which was to our knowledge the first port to any sVr4 platform, were performed by my assistant Michael Brown. The CAP software seems totally reliable, and I would recommend it.

Our "Add AUFS User" script provides our users by default with access to three file-server Volumes; an Individual Volume named after their login name, a Group Volume where all that group can read and write, and a Public volume which is read-only. Some users find three icons to be too fussy, and we might at some stage consolidate the Individual Volume as a folder in the Group Volume.

NFS for the PCs

NFS for PCs suffers from a major administerability problem; the details of the server hostname and the absolute pathname of the user's file-server directory are needed by a *net use* command on the PC, whereas this information should be administered centrally, for example to allow data to be moved between discs. On the other hand, other information such as the user's name, and the user's preferred printer, does properly reside on the PC. Then again, the list of available printers, or authorised users, that the user is to choose from, should reside on the server.

In order to be able to hold these items of information in their appropriate places, I wrote a DOS batch file *connect.bat* which resides on the server and is summoned up from *autoexec.bat* as follows:

```
net use k: servername:/var/config/pcnfs -ro
call k:\bin\connect.bat
```

Connect.bat then uses the DOS version of *choose*, and a DOS program called *setenv* which overcomes BATCH's lack of backquotes, to offer the user the following choice ... line right ; line left line right 5.2; arc; line up 2; arc line left 5.2; arc; line down 2; arc

```
User = oliveo, LPT2 = ps_admin                                pjb
```

```
Connect to the network          Change user
Change printer                  Do not connect to the network
About the network
```

```
Use: up, down, left, right, mark, reject, accept
Or:  k,  j,  h,    l, space, q, ^C, ^Z, return
```

If the user chooses "Change user", they are offered a choice of legitimate users from */var/config/pcnfs/menus/users*, and the result is stored on the PC in *c:\nfs\user.bat*. Changing printers is analagous. If the user chooses "Connect to the network", */var/config/pcnfs/users/oliveo.bat* registers the user with *psnfsd* and mounts the user's correct Individual, Group and Public drives.

Since the introduction of *connect.bat* we find that *PC-NFS* is as self-administering as *AppleShare*.

Macs and PCs sharing the same file-server volume

An *AUFS* directory can easily be exported by NFS also, providing *DOS* filename standards are respected, or enforced. We have programs, written by Michael Brown, which truncate Mac filenames unambiguously, and add Mac Icons to PC Word and Excel documents. Running these programs nightly over all new files in the directory results in a volume shareable between Macs and PCs.

When Administering a Network, Cost and Worth are Opposites.

A public domain operating system such as 386BSD is inherently more useful (ie. worth more) than a purchased one with similar features:

- Its useful lifetime is not limited by licencing, or by maintenance expense.
- Its security is improved because of the availability of the source code, both because the code can be audited, and because it can be modified (for example so that shell escapes go to the login shell rather than */bin/sh*).
- It is easier to administer, because to know the state of a system it is not necessary to keep track of many small incremental changes; the operating system can simply be deleted, and restored to a known state; a much simpler operation.
- That state only needs to be maintained on one machine, because the others may be freely copied or mounted from it.
- It is easy to produce highly secure modified or stripped-down versions of the operating system for particular purposes; for example a file server does not need *su*, nor *fingerd*, and *telnetd* can be modified to accept logins only from IP addresses mentioned in a */usr/allow/telnet* file. Master copies of these stripped-down versions of */usr* should be kept on a System Server, see below.

Most of these advantages also apply to other public-domain software, such as *Eudora* and *PCEudora*.

System Servers

If you have 50 PCs running Word, of course you have to buy 50 copies of Word. However, there is no need to perform the useless work of actually unpacking and installing them. All you have to do is to install one, copy that hard disc to */u1/pcnfs/exports/word* and use */etc/exports* to export */u1/pcnfs/exports/word* read-only to precisely those 50 PCs. The PCs then mount that */u1/pcnfs/exports/word* on *E:* and all they need on drive *C:* is an NFS client. The remaining 49 copies of the software should either be locked in a safe, or, to save safe space, destroyed in front of witnesses who sign an document to that effect.

Using this system, great savings of disc space and upgrade work are made, virus resilience is much improved, and licencing legality and access control can both be easily enforced from one file, namely */etc/exports* on the System Server.

Of course, the same principle applies to Unix boxes, so that */usr/exports/firewall*, */usr/exports/router*, */usr/exports/fileserver+router*, */usr/exports/mail+news* and so on can be exported read-only to appropriate IP addresses. The client box then, from */etc/rc.local* or equivalent, mounts the appropriate directory from the System Server over the top of its local */usr*. */etc/rc.local* should also call a script in */usr*, say */usr/local/rc*, so that the System Server can, with superuser privileges, verify the state of the local root partition, including */etc/passwd*, */etc/services*, and so on.

The System Server uses the following directories:

/usr/exports

whose various sub-directories, for example *firewall*, *fileserver*, *router*, *mail+news*, etc, contain complete */usr* filesystems for their respective clients, and

/u1/pcnfs/exports

whose subdirectories */word*, */wordexcl*, etc, contain complete *E:* drives for their clients.

I am convinced that file serving is far more valuable to the administrator when used in this way, to export software, than when used to provide big discs and regular backups for user data, because it reduces the task of administering hundreds of computers to the task of administering one computer.

If the goal of the System Administrator is in any sense to deliver as much functionality for as little money as possible, then the mythical ideal System Administrator would deliver complete functionality for no money. *386BSD* helps us strive for this goal.

Peter J Billam
August 10, 1993

Open, Distributed OnLine Transaction Processing †

*Anthony D. Hooten
IBM Corporation*

Online Transaction Processing (OLTP) is the backbone of businesses throughout the world. Advances in distributed computing and transaction processing software, standards-based computing, and powerful workstations have led to a migration towards open, distributed transaction processing. This paper provides an overview of the evolution to open OLTP and of IBM's open, distributed OLTP structure. This open OLTP structure is built on the Open Software Foundation's Distributed Computing Environment (DCE), utilizes the Encina transaction processing technology from Transarc, and offers a modern, state-of-the-art implementation of IBM's CICS OLTP monitor.

Online Transaction Processing (OLTP), or simply Transaction Processing (TP), is the backbone of businesses throughout the world. TP systems, sometimes referred to as TP Monitors, offer businesses a secure, reliable means of recording orders, sales, transfers, reservations, and any other updates to data that is shared, potentially among many thousands of users. OLTP systems guarantee the consistency and retrievability of that revised data, ensuring that an update is never lost. OLTP systems guarantee access to and the integrity of the mission critical data that is used to run an enterprise, whether that be account balances, inventories, flight bookings, or other business data. The most important attribute of a TP system is that it ensures additional semantic guarantees for accessing and updating shared data. Through these guarantees, transactions greatly simplify the programming of applications that employ concurrency and that recover from failures.

OLTP, now entering its fourth decade, has a very rich history in systems like IBM's CICS and IMS. These and other TP monitors evolved as system designers recognized common requirements in the online systems they were building. These requirements included: the assurance of reliable, real-time access to large databases; the preservation of data integrity even with concurrent updates and system failures; the scalability to admit large numbers of simultaneous users without sacrificing performance; and the security of limiting data access to authorized users only through specified routes or audited applications.

Today, most OLTP applications and systems run on mainframe computers that serve large data processing centers, such as airline reservations and international banking. However, TP

† This paper is a reprint from the proceedings of the UniForum NZ 93 conference.
AUUGN

monitors are increasingly being used for many other diverse computing needs, such as retail point of sale, manufacturing control, telecommunications, government, health care, insurance, legal services, and image processing. Advances in distributed computing and transaction processing software, standards-based computing, and powerful workstations have led to a migration towards open, distributed transaction processing.

The migration to open, distributed transaction processing is the culmination of four technology evolutions in computing: distribution, open systems, client/server and object orientation, and transaction processing technology itself.

Today, businesses are becoming more decentralized. At the same time, smaller, more affordable networked computing systems are offering services previously performed solely by large, centralized systems. By permitting computing services to match and evolve with the organizational structures of business, distributed systems allow autonomy for the resources which form the enterprise computing base. The result is decentralized procurement, development, and management. An enterprise can achieve greater configuration flexibility because applications and data can be placed where they are being used. Interconnecting systems facilitate real-time, wide-area sharing of services without requiring secure data or programs to be duplicated. Distribution also enables the incremental growth and evolution of an enterprise's computing base; PCs, workstations and servers can simply be added or replaced as the need arises. The most obvious reason to migrate toward distributed computing is to take advantage of the improved price/performance of microsystems. But distributed systems can also provide greater reliability and availability, through redundant hardware and software, and worker productivity gains, through advanced user interfaces and tools which now come with powerful desktop computers.

Although distributed computing addresses many business challenges, it also can introduce further complexity into the computing environment. The main obstacle is linking heterogeneous computing platforms in a way that applications can communicate with one another, can be run on different machines, and can be effectively administered. By emphasizing common application program interfaces, open systems permit interoperability, the ability for applications developed on one platform to run on others. Open systems also permit distributed applications, using common communication protocols, to work together by sharing data and/or computing resources. Open systems provide the flexibility to meet changing computing requirements with little sacrifice of investment.

Client/server technologies permit applications to be separated into presentation logic, implemented in front-end clients, and business logic, implemented in backend application servers. Well-defined application server interfaces allow sharing of business services and data among a potentially diverse set of clients. Client/server allows application designers to choose the interface between the client and the server. This offers greater program modularity, increased ease of evolution, better performance, and stronger security.

Transaction processing technology has changed significantly to meet the needs of open, client/server, distributed computing. Traditionally, a transaction was thought of as the

mainframe-based application procedure that terminals invoked to process a screen, as business logic and presentation logic were packaged together as a single host transaction. A transaction is more broadly defined in the distributed client/server model. A transaction could just as easily be the part of a client or a server application. Each operation could be a local procedure call to services within the same program or a Remote Procedure Call (RPC) to services running on remote nodes. A single transaction can encompass applications and data distributed over local or even wide area networks. The underlying system takes care of managing conversations, packing messages, routing requests, converting data, and how the operations together as part of the same unit of work. A transaction could also include database commands, queuing commands, or communications with other TP monitors.

Integrity is particularly critical in a distributed TP system. On a mainframe, if one piece fails, the entire system fails. In a distributed environment, any one component of the system could falter while the remaining pieces continue working to complete a transaction that should have been canceled. To overcome this challenge, the two-phase commit procedure was developed. Two-phase commit makes sure a transaction is either committed correctly or is aborted. In the first phase, called the prepare phase, all participants in a transaction are polled to see if they are prepared to commit to a transaction. This polling is done by the transaction coordinator. An analogy would be a travel agent trying to make plane, hotel and rental car reservations at once. The agent doesn't want to book a flight if a hotel room can't be reserved, or request a rental car on Tuesday if the passenger can't get plane until Wednesday. So the agent finds out if all of the reservations can be made before committing to any of them. The second phase is called the commit phase, in which the transaction is either committed, if all participants respond affirmatively, or rolled back if any of the participants respond negatively.

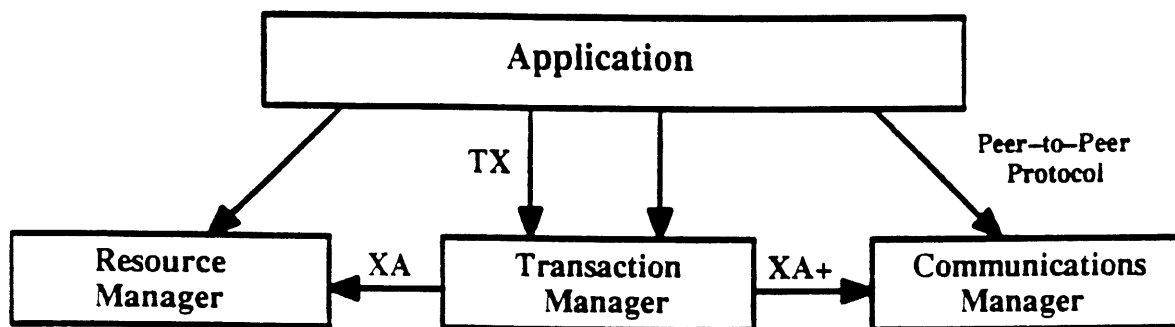


Figure 1: The X/Open Distributed Transaction Processing (DTP) Model

The standard in open systems for two-phase commit is X/Open's XA interface. X/Open's Distributed Transaction Processing (DTP) model includes several other interfaces which have yet to be fully defined. The model itself, see Figure 1, consists of applications, resource managers, a transaction manager or monitor, and communications resource managers, such as networking protocols and remote procedure calls. The XA interface is between the resource manager and the transaction manager. This XA interface enables a standard way for

transaction managers to coordinate transactions, using two-phase commit, with resources managers, such as relational databases.

There are four major transaction processing environments which are X/Open XA-compliant, are available for open systems platforms, and are expected to be the dominant TP environments in this new age of open, distributed, OLTP. These TP environments are: CICS from IBM, ENCINA from Transarc Corporation, TUXEDO from UNIX Systems Laboratories (USL), and Top End from NCR. Each of these TP technologies can be licensed by vendors and ISVs and each TP environments is, or will be, available across multiple open systems platforms. All of the major relational database vendors have announced support for XA and many are currently shipping XA-compliant databases. Some of these include Oracle7, Informix-Online 5.0, IBM's DB/6000, HP's Allbase/SQL, and USL's Tuxedo System/D.

The way to make applications portable and interoperable across a wide range of platforms is to adopt a standard distributed computing infrastructure that can be layered upon alternative networking and operating systems. This shields applications and their administration from the underlying complexities of heterogeneity. Figure 2 shows a general architecture for distributed transaction processing which incorporates the X/Open DTP model.

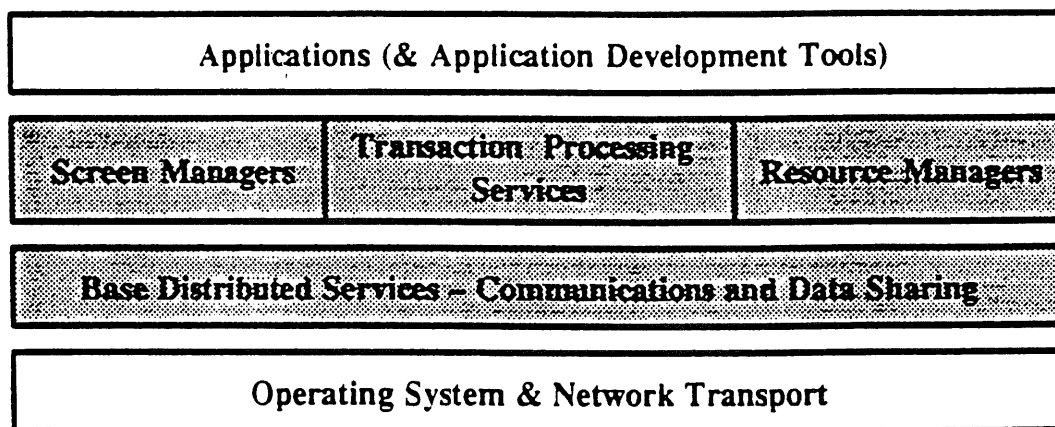


Figure 2: Open, Distributed, OLTP System Architecture

IBM and other open systems vendors are utilizing OSF's Distributed Computing Environment (DCE) to provide the base distributed services in this open, distributed OLTP architecture, and for other leadership distributed applications. The Open Software Foundation's (OSF's) Distributed Computing Environment (DCE) provides the base set of distributed services and tools that support the creation, use, and maintenance of distributed applications in a heterogeneous computing environment. DCE is based on three distributed computing models: client/server, remote procedure call, and data sharing. The client/server model is a way of organizing a distributed application. This model permits applications to be separated into presentation logic (implemented in clients) and business logic (implemented in servers). The

remote procedure call model is a way of communicating between parts of a distributed application. The data sharing model is a way of handling data in a distributed system.

Architecturally, DCE is a software layer between the operating system and network on one hand, and the distributed application layer on the other (see Figure 3 – DCE Architecture). DCE is sometimes referred to as *middleware*. DCE provides the services that allow a distributed application to interact with a collection of possibly heterogeneous computers, operating systems, and networks as if they were a single system. There are several technology components which work together to implement the DCE layer. These components include DCE Threads Service, Remote Procedure Call, Directory Service, both Local and Global, Time Service, Security Service, and Distributed File Service.

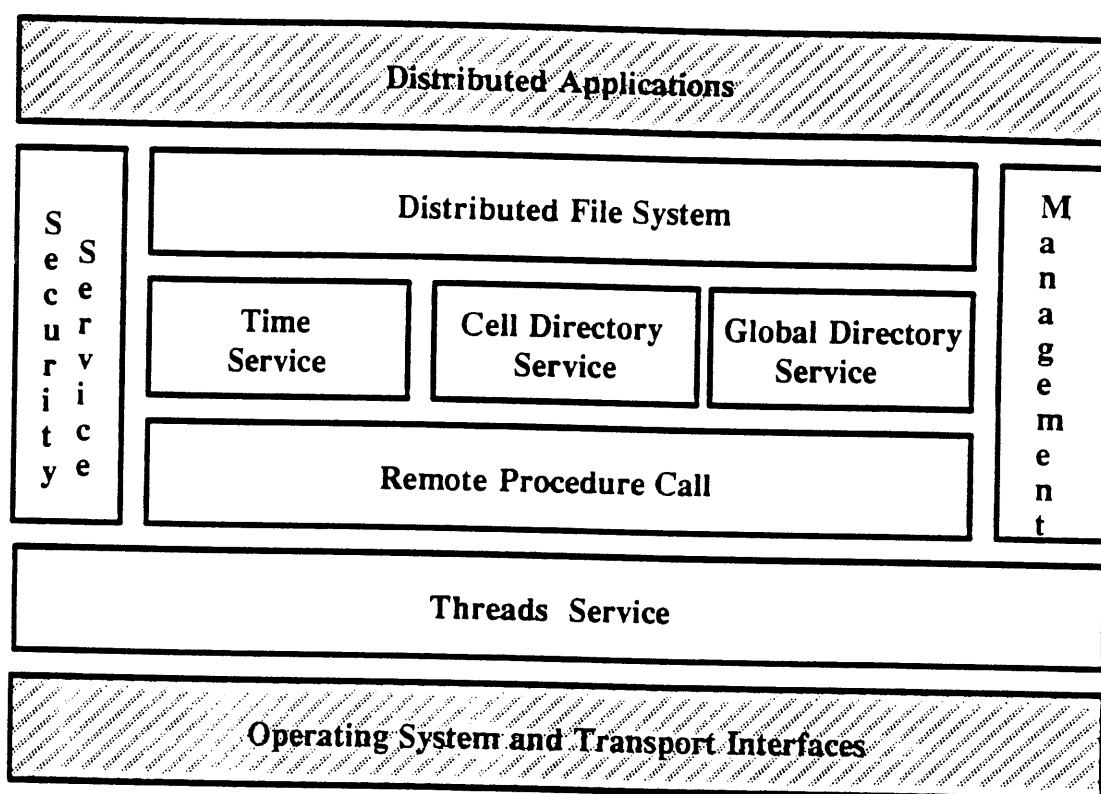


Figure 3: DCE Architecture

DCE Threads supports the creation, management, and synchronization of multiple threads of control within a single process. The Threads service provides portable facilities that support concurrent programming, thus allowing an application to perform many actions simultaneously. The Threads interface conforms to the POSIX 1003.4a draft standard.

DCE Remote Procedure Call (RPC) distributes application execution. It extends the familiar local procedure call model by supporting direct calls to procedures on remote systems, thus enabling programmers to develop distributed applications as easily as traditional, single-system programs. RPC provides for consistent client / server interactions. It handles

translation of disparate data representations, and important capability in a heterogeneous environment. RPC also provides integrated error handling and recovery procedures.

DCE Directory Services provides a single hierarchical naming model throughout the distributed environment. They allow users to identify by name resources such as servers, files, disks, and print queues, and gain access to them without knowing where they are located in a network. The DCE Directory Services is comprised of a Cell Directory Service and a Global Directory Service.

The **Cell Directory Service** stores names and attributes of resources located in a DCE cell. It is optimized for local access. The **Global Directory Service** is used for looking up names outside the local DCE cell. Global names can be either located in the Internet Domain Name System name space or in the X.500 standard directory service. Traditional UNIX file system naming is used once a name crosses into the file system. DCE applications are accessed by traversing this name space. DCE names are independent of the named object's actual location. Users and applications can use the Directory Services to store, retrieve, and manage information about resources within the Distributed Computing Environment.

DCE Time Service synchronizes all clocks on a network to a widely recognized time standard. This provides precise, fault-tolerant clock synchronization for systems in both local and wide-area networks. The Time Service also interoperates with the Network Time Protocol (NTP) found in many existing network environments.

DCE Security Service provides for authentication, authorization, and user account management. The authentication service guarantees a principal is who it claims to be, while the authorization service determines whether a principal has privileges required by a request. Together these services offer a secure means of communication that ensures both data integrity and privacy, and prevents unauthorized access to a distributed environment. Multiple protection levels are supported by the Security Service. This allows a user to determine when authentication is performed and how much encryption should take place. The authentication service is based on MIT's Kerberos systems, a commonly used and well-understood authentication mechanism. DCE authentication is accomplished without sending passwords in the clear, unlike most UNIX applications. Authorization is based on Access Control Lists (ACLs), which limit who is allowed to use certain services or look at certain files. The Security Service is integrated with RPC to provide for secure communication.

DCE Distributed File System (DFS) is the data sharing service for use within the DCE environment. It gives users a uniform means of accessing remote files, regardless of their location. DFS allows for sharing of data transparently. By joining the file systems of individual workstations and providing a consistent interface, it makes global file access as easy as local access. DFS utilizes a technique called caching to provide better service to the user and lower the active load on file servers. This allows for a higher client-to-server ratio and supports increased scalability. DFS provides a strong cache consistency protocol to obtain local file

system semantics, which means it works as a local UNIX file system would act. Because DFS fully exploits the other DCE services, most administrative operations can be performed remotely, substantially decreasing the administrative burden for many sites. DFS provided for high availability via replication and supports co-existence with NFS. DFS is based on CMU Andrew / Transarc AFS / and DEcorum technology.

DCE's set of services is integrated and comprehensive. Not only does DCE provide all of the tools and services needed for developing and running distributed applications, but the DCE components themselves are well integrated. They use one another's services whenever possible, since many of the DCE components are themselves distributed applications. For example, the RPC facility uses the Directory Service to advertise and look up RPC-based servers and their characteristics. RPC uses the Security Service to ensure message integrity and privacy. RPC uses the Threads Service to handle concurrent execution of multiple RPCs. The Distributed File Service uses Threads, RPC, Directory Service, Time Service and the Security Service in providing its file service. In addition to supporting the development of distributed applications, DCE includes services that address some of the new problems inherent in the distributed system itself, such as data consistency and clock synchronization. DCE also includes management tools for administering all of the DCE services and many aspects of the distributed environment itself.

In September, 1992, IBM announced state-of-the-art transaction processing on its RISC System/6000 family, based upon the open distributed TP architecture presented earlier, using the Encina technology from Transarc Corporation and a version of IBM's Customer Information Control System (CICS) for AIX. See Figure 4 for IBM's open, distributed OLTP architecture.

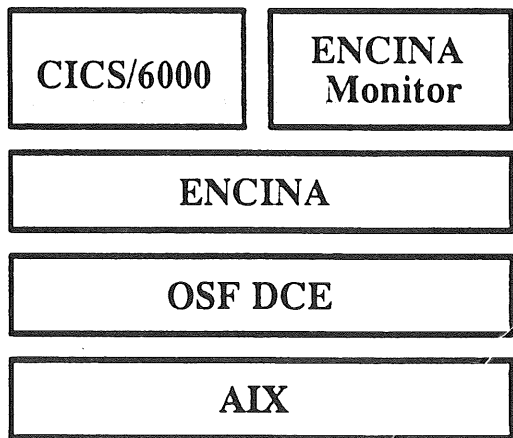


Figure 4: IBM Open, Distributed OLTP Structure

IBM's Encina and CICS/6000 products are built on and utilize the distributed application services provided by OSF's DCE. Encina and CICS are the first transaction processing environments built on this widely adopted standard for heterogeneous, distributed computing. These products provide new levels of capability in the open systems market, provide for future

growth and enhancements, and help customers capitalize on their existing OLTP investments and user skills. The IBM offerings feature a modular approach that delivers a core set of basic, standards-driven functions together with a choice of two transaction monitors, a key component that helps users manage the performance and administration of their systems. Customers can choose from a variety of elements to tailor a solution to fit their requirements, including one or both of the transaction monitors, AIX CICS/6000 and the Encina Monitor for AIX/6000. This approach offers users more flexibility and choice.

Encina, from Transarc Corporation, is based on the OSF's DCE, and extends many of the DCE services to provide a full-featured TP environment for transactional client/server computing. Encina uses and exposes the base distributed services provided by DCE. By building on this DCE foundation, Encina inherits the benefits of DCE including: transparent access to remote services, enterprise-wide scalability, automatic protection of services from unauthorized access, privacy for applications distributed over unsecure networks, POSIX-standard high-performance concurrency technology, and secure global file sharing. Encina fits centrally within the open OLTP framework with DCE providing the base distributed services. Encina is integrated with both screen managers on the client-side, and relational database management systems (RDBMSs) and other resource managers on the server side.

While the DCE addresses much of the complexities of distributing computation, distributed application developers face further problems for which TP concepts are relevant. Transaction processing is fundamental to enterprise computing because it provides data integrity in the face of concurrency and system failures. Transarc has developed the Encina family of TP products, shown in Figure 5, to simplify the construction of reliable distributed systems and to provide the integrity required for mission critical enterprise computing. The Encina architecture is based on a two-tier strategy:

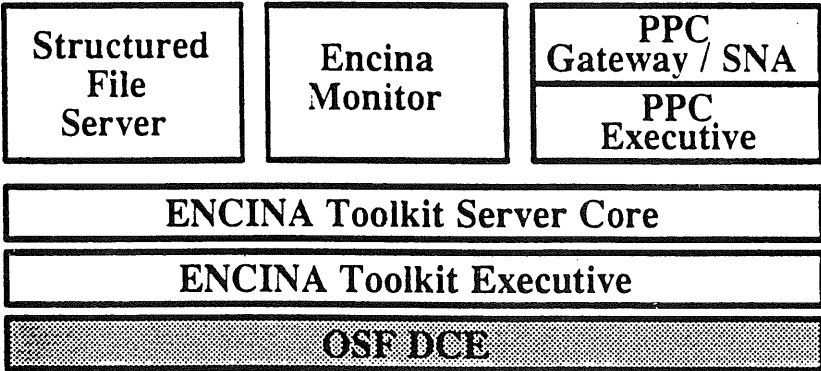


Figure 5: Encina Family of TP Products

- Tier 1: Expand the DCE foundation to include services that support distributed transaction processing and the management of recoverable data.
- Tier 2: Construct a family of high-function TP services based upon this expanded foundation.

Transarc's Encina Toolkit, when coupled with the OSF DCE, provides standards-based, distributed TP foundation services for simplifying the construction of reliable, distributed applications. It is comprised of the Encina Toolkit Executive and the Encina Toolkit Server Core:

The **Encina Toolkit Executive** extends the services of the OSF DCE with core technologies that enable client/server transaction processing. It includes Transactional-C, a high-level application programming interface that provides for transaction demarcation and concurrency management. The Encina Executive also extends the feature-rich remote procedure call (RPC) technology of the DCE to transparently ensure transactional integrity over distributed computations. Supporting these APIs is the transaction service, a powerful two-phase commit engine. The DCE and Encina Executive together permit a workstation to initiate transactions, to locate services, to securely invoke those services, and to commit distributed transactions.

The **Encina Toolkit Server Core** extends the services of the Executive to support the storage and maintenance of recoverable data. It includes a locking library to serialize data access, a recoverable storage system based upon write-ahead logging, an X/Open XA interface to permit interoperability with compliant databases, and a common log.

Encina is unique among commercially available TP systems in its support for nested transactions. Nesting, the ability to define transactions inside other transactions, is crucial to this more general distributed transaction model. Nested transactions simplify the structuring of larger transactions. They are fundamental to modular and object-oriented programming methodologies in which base services are composed together to form higher level services without regard to the details of their implementations.

A wide variety of commercial, distributed systems can be built using the foundation services of the Encina Toolkit and the DCE: TP monitors, database management systems, CASE and other application development tools, recoverable memory managers, recoverable queuing systems, and administrative tools. Transarc and IBM have developed their own family of TP products built on the DCE-based Encina Toolkit. These products include:

The IBM AIX **CICS/6000** product is a new version of IBM's CICS software, which is the most widely used transaction processing monitor in the industry. CICS/6000 provides a complete OLTP environment, including queue management, priority scheduling, character-based screen management, and support for COBOL and C languages and relational databases. CICS/6000 allow users to easily transfer programmer skills and applications from existing CICS systems to the IBM RISC System/6000 environment.

The **Encina Monitor**, which is a full feature monitor providing a powerful, easy-to-use development environment, a reliable execution environment, and an effective administration environment. The development environment supports the programming and integration of distributed TP applications. Front ends may be PCs, ASCII terminals, and/or UNIX workstations. Programmers can utilize the services of the Encina products as well as those of various resource managers and the DCE. The execution environment delivers load balancing and scheduling across heterogeneous computers to ensure high

performance and data integrity. The administration environment enables the configuration and management of the distributed OLTP systems as a cohesive unit.

The **Encina Structured File Server (SFS)**, which is a record-oriented file systems that provides full transactional integrity, high performance, log-based recovery, and broad scalability for very large databases. It is designed to participate in two-phase commit protocols. The Encina SFS is internationalized and support B-tree clustered, relative, and entry-sequence access methods which permit both X/Open ISAM and VSAM-like interfaces.

The **Encina Peer-to-Peer Communication (PPC) Services**, which offer vital enterprise computing services by supporting transactional CPI-C peer-to-peer communication over both TCP/IP and SNA (LU6.2) transports. The PPC services are comprised of the **PPC Executive**, which permits Encina platforms to carry on transactional peer-to-peer conversations over TCP/IP, and the **PPC Gateway/SNA**, which permits systems using the PPC Executive to communicate over SNA/LU6.2 to mainframe systems that support LU6.2, such as IBM's CICS. Encina transactions can either coordinate or be coordinated by transactions using LU6.2.

The Encina products have been endorsed as the basis for strategic, open online transaction processing solutions from leading commercial computing vendors including IBM, Hewlett-Packard, Stratus, NEC, and Hitachi.

With CICS/6000, IBM has combined elements of its established CICS product with OSF structures. Without doubt, the most important combination is the CICS application programming interface (API) with the OSF Distributed Computing Environment (DCE). The CICS API provides an important link with IBM's established TP base and offers the potential to evolve legacy systems to new platforms. It also provides a large skill base which can be harnessed to the task of building the complex new applications which will result from the transition to multivendor systems and networks. To deliver this, substantial parts of CICS were completely rewritten in the C programming language. But IBM recognized that a complete rewrite of CICS would have been unnecessary.

Much of the functionality required by CICS was already available, either within AIX or through third parties. IBM went to Transarc for the additional function not already in AIX. Thus the CICS/6000 monitor sits on top of Transarc's ENCINA toolkit, enabling the CICS API to exploit two-phase commit, transaction scheduling, record logging, and roll-back features. In addition, CICS/6000 uses Transarc's Structured File Server (SFS) to provide ISAM/VSAM-like services and Peer-to-Peer Connectivity (PPC) services to connect to legacy systems.

CICS/6000 also uses ENCINA to get at resources available through OSF's DCE, especially the RPC to enable inter-applications communications. ENCINA includes systems recovery features which are not specified in the native DCE RPC. CICS/6000 supports the XA interface defined by X/Open. This allows CICS/6000 to interwork with other transaction processing monitors, the ENCINA Monitor being an obvious partner. This is desirable if open transaction processing applications are to work across multivendor networks. IBM's modular approach to

CICS/6000 has several other advantages, for example, allowing CICS applications to be distributed across several machines, both IBM RISC System/6000s and other open systems platforms who support this CICS implementation.

The extension of CICS to AIX and other open systems platforms is highly significant and points to a model for transaction management based upon long experience in transaction processing. CICS/6000 provides a high level migration route for current CICS users to alternative hardware environments and provides open systems users access to a rich library of tried-and-tested applications software. This IBM CICS implementation has also been licensed by HP for HP workstations, and is expected to be made available on other open systems platforms in the future. The Standish Group International, a market group which tracks OLTP, projects that IBM's CICS for open systems will be the clear market leader in open systems TP environments by 1995.

It is expected that DCE and distributed TP environments like Encina and CICS/6000 will have a significant impact on both the technology in future relational database management systems (RDBMSs) and on how RDBMSs will be used by customers. There is envisioned a growing level of interoperability between RDBMSs. In addition to interoperability features like a common native interface as specified by the X/Open SQL standard, many RDBMSs will use DCE to provide for heterogeneous communications at a higher level.

One of the important elements of the Encina Toolkit Server Core is support for the X/Open XA interface. The XA interface is an X/Open standard for initiating and coordinating subordinate database transactions. This standard has been endorsed by most of the major RDBMS suppliers, including Ingres, Oracle, Informix, Sybase, and Interbase. The **TRAN/XA Interface** provided by the Encina Toolkit Server Core permits transactional access to XA-supporting databases, such as the majority of relational database management systems. This support for the X/Open XA interface will enable these relational database management systems to be used in conjunction with either the CICS/6000 or Encina monitor environment.

The Gartner Group has predicted that by 1995 more than half of the new applications using relational database management systems will employ a TP monitor. While some RDBMSs alone do supply basic transactional properties, TP monitors are better positioned to give you interoperability between databases, interoperability with other TP monitors, choice of a range of screen managers, highly optimized performance, security, and the scalability to deliver your application to many widely dispersed users.

Summary

Online Transaction Processing (OLTP) is the backbone of businesses throughout the world. Companies have traditionally done transaction processing in large, centralized, computing environments, but as their businesses evolve to include more departmental computing and more global operations, many are looking to add distributed, standards-based computing networks as a cost-effective way to increase capacity and flexibility. Advances in distributed computing and transaction processing software, standards-based computing, and powerful workstations have led to a migration towards open, distributed transaction processing. Built on and utilizing OSF's Distributed Computing Environment (DCE), IBM's CICS/6000 and the Encina family of products provide a comprehensive framework for the development and deployment of distributed transactional applications, and meet new distributed, open systems requirements.

With the combination of Encina and CICS/6000, IBM is offering complementary strategies for open, distributed OLTP. CICS/6000 allows enterprises to preserve their investment in CICS application programs and programmer skills by offering CICS APIs and administrative interfaces. Encina, on the other hand, offers DCE-style program development and administration. Each product shares a foundation of DCE and Encina technologies, and as a result, there is substantial interoperability between them. Together, these technologies deliver a complete, integrated environment for open, commercial-grade, distributed transaction processing.

References

- Dietzen, Scott. "Distributed Transaction Processing with Encina", *AIXpert*, November, 1992.
- Encina Product Overview, Transarc Corporation. 1991.
- Knowles, Anne. "Monitor Maneuvering", *RS/Magazine*, October, 1992.
- Kumar, Ram. "OSF's Distributed Computing Environment", *AIXpert*, Fall, 1991.
- Mason, Tony. "OSF's DCE", *UNIX Review*, January, 1992.
- Schulte, Roy. "Unix TP Monitors Proliferate", *Software Management Strategies*, The Gartner Group, Inc., November, 1991.
- Steiner, Jennifer. *Introduction to DCE*. Open Software Foundation, 1991.
- Storey, Tony and John Knutson. "CICS/6000 Online Transaction Processing with AIX", *AIXpert*, November, 1992.
- "The Opening of CICS and Transaction Management", *The SAA and Open Software Spectrum*, Volume 6, Report 5, 1993.

Trademarks

IBM, AIX, CICS, MVS, AS/400, OS/2, DB2, InforExplorer and RISC System/6000 are trademarks of International Business Machines Corporation.

UNIX is a registered trademark of UNIX Systems Laboratories.

Open Software Foundation, Motif and OSF/1 are trademarks of the Open Software Foundation.

X/Open is a trademark of X/Open Company Limited.

Transarc, Encina, and AFS are trademarks of Transarc Corporation.

DEC and DECstation are trademarks of Digital Equipment Corporation.

Hewlett-Packard and HP are trademarks of Hewlett-Packard Company.

Siemens is a trademark of Siemens & Halske Aktiengesellschaft.

Ingres is a trademark of Ingres Corporation.

Informix is a trademark of Informix Software, Inc.

Oracle is a trademark of Oracle Corporation.

Sybase is a trademark of Sybase, Inc.

X/Open and Interoperability

[Tech123][Tech151]

Abstract: The range of issues of concern to interoperability are discussed in the context of the standards needed, and the work of X/Open in finding a comprehensive set of answers. This article is taken from the X/Open publication "X/Open and Interoperability"+

Petr Janecek

At present, interoperability is at the forefront of demand in computing. Now is therefore a very appropriate time to offer a guide to "X/Open and Interoperability". At the same time, interoperability is an immensely complex subject. This Guide is not a treatise on details of communications and other aspects of interoperability. It is aimed at corporate decision makers who wish to get a brief orientation about:

- what X/Open understands by interoperability;
- what it has done and does to make interoperability happen;
- how developers can design an open systems strategy by endorsing XPG specifications;
- how procurers can devise a procurement strategy based on X/Open branded products.

What is an "Open System" ?

An open system is one which conforms to international computing standards--and is available from more than one independent supplier. In other words, it should be both "standard" and "multiple sourced".

In this context, a system might be either a complete computer system, or a discrete hardware or software component of such a system.

To measure whether a system is open, it is important that both the "standard" and the "multiple source" criteria are used . Genuine openness and the vital goal of interoperability cannot be assured in technical and commercial terms unless both criteria are met.

In practical terms, an open system is exchangeable for any other open system with a compatible set of standardised features.

The Benefits of Open Systems

An open systems strategy guarantees low risk and secure investment for the future through:

- **application portability**--an application running on one open system will run on another (regardless of the supplier) with little or no change;
- **interoperability**--an open system will work with any other open system which has a compatible set of standardised features, regardless of who supplies it;
- **secure investment in data**--users' data is one of their most valuable items; if it is held in an open system, it becomes portable to another open system and it also remains valid and usable through successions of future system upgrades;
- **secure investment in people skills**--staff skills and training remain valid, in a stable environment and predictable growth path;

--

- **controlled development**--users retain full control over the pace and costs of their information technology operation: with open systems, planned and integrated development is a reality;

- **stability**--because it is based on implemented and stable international standards, an open system is inherently:

- of sound design,

- long-lived, not prone to unforeseen updates and "fixes",

- compatible with future releases;

- **vendor independence**--open systems do not tie the user to a single supplier;

- **competitive pricing**--open systems compete in a multi-supplier market.

The alternative to open systems strategy is to tie oneself into a proprietary system. In every respect, however, the user's options for portability, interoperability with other systems, data portability, future changes/upgrades, stability, and competitive pricing would then be severely restricted. As a result, choice of options, availability of products, and cost control over future change are all heavily compromised, if not totally lost. It is therefore clear that any advantage a proprietary system might offer must be very large indeed to outweigh the benefits of open systems.

X/Open's Role

The corporate mission of X/Open is "To bring greater value to users from computing through the practical implementation of open systems". In other words, X/Open strives to promote practical open systems.

X/Open attempts to provide answers to needs of users of information technology. It therefore closely monitors user needs, hardware and software technology development, and system and software vendors' requirements. It does this through its Xtra programme and publishes the results in a document called the Open Systems Directive (OSD).

X/Open then responds to the needs identified in the OSD in three distinct ways:

- It develops technical specifications which meet the requirements achieving industry consensus, and compliance with international standards. These specifications are then published and referenced in the XPG (X/Open Portability Guide). XPG describes the X/Open computing environment called Common Application Environment (CAE). Software developers then use XPG to develop products which take advantage of these standardised specifications. (The title Portability Guide is, however, only kept for historical reasons and it has become a misnomer: X/Open is no longer only concerned solely with portability, and XPG is a series of normative specifications, not just advisory guides.) This does not restrict the freedom of the supplier to implement his software in the optimal way, nor does it limit his ability to add extra value. Suppliers can differentiate their products from other offerings by, for example, including better performance and/or extended functionality.
- It issues Guides covering important topics which are useful in developing, evaluating, procuring or managing open systems.
- It offers a branding programme. A supplier's software which demonstrates compliance with X/Open's specifications in relevant areas is awarded the X/Open brand. Both whole systems and individual software components can be branded. This X/Open brand is the user's guarantee that a product is genuinely "open".

Products are branded with respect to a particular version of the X/Open Portability Guide. XPG3 was published in 1989 and XPG4 was released in 1992.

X/Open itself is not a formal standards body although individuals active in X/Open working groups also contribute to many committees involved in developing formal standards, on both national and international levels.

X/Open has adopted a standards policy which commits it to aligning its specifications with formal standards and, where appropriate, actively promoting its specifications as a base for international standards. This policy has been successfully applied in X/Open's relationship with the IEEE POSIX activities where three of X/Open's OSI APIs have been adopted as IEEE standards.

Through its XPG and branding program, X/Open offers the most widely accepted specification platform in the industry on which open interoperable systems can be built.

Elements of Interoperability

Portability Connectivity and Interoperability

From the point of view of exchangeability, open systems can support two features of application software: portability and interoperability.

Portability of applications means that applications written to a standard set of interfaces can easily be ported to all systems on which this set is implemented. Applications thus become largely independent of the computer system they run on.

Portability is essentially a feature of a standalone system. In other words, a user benefits from application portability even if he only has one single open system, since his applications become independent of the specific make of computer he currently uses.

Source code portability is achieved through standardising of Application Programming Interfaces (APIs) which are source-level interfaces, usually procedure calls, used by application developers.

The other feature of open systems, interoperability of distributed systems and applications, is an aspect of computer systems in communication. Interoperability is the ability of application processes to cooperate through exchange of information and services.

A minimum requirement for interoperability between two computer systems is that they speak the same standardised communications language, i.e. that they encode and decode data using the same protocols. However, the use of standardised protocols will only result in connectivity, or transfer of information, being achieved. Interoperability is a higher quality which often requires the use of additional conventions such as standardised data formats, programming interfaces, file name mappings, etc.

The relationship between these features is shown at Figure 1.

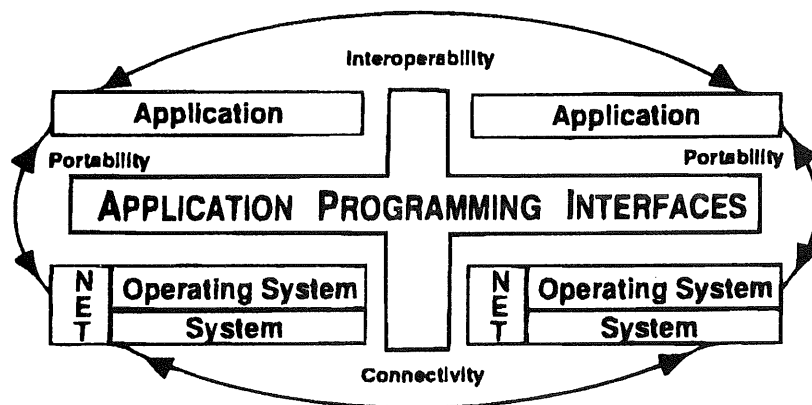


Figure 1. Portability and Interoperability

The Distribution

Interoperability is a desirable feature of products operating in networks of distributed entities. These entities are:

- human users;
- data;
- computing power;
- peripherals.

In order to utilise the distributed entities optimally, application software can itself be split into parts on different nodes in the network. It then becomes a distributed application.

Users of the network need to access its resources either:

- transparently (without having to identify where in the network the resource is located), or
- remotely (explicitly specifying the details of the remote resource being accessed).

User requirements with regard to functionality have led to the identification of the main services which must be provided to the user on various nodes in the network.

The OSI Solution

Interoperability of a number of disparate systems could of course be achieved by a set of one-to-one interoperability solutions, each tailored to one specific pair of systems. This is a costly method when many kinds of systems are involved. Standardisation is the obvious alternative.

For the past twenty years, the United Nations' International Organisation for Standardisation (ISO) has been working on a set of standards jointly called the Open Systems Interconnection (OSI). The OSI Reference Model provides the conceptual framework; its basic features are described in the standard ISO 7498 and its addenda.

The model divides communications into seven functional layers, where each layer relies on services provided by the layers underneath it. In the highest, i.e., the seventh layer, a number of standardised application services are defined, for example:

- Message Handling
- File Transfer, Access and Management
- Virtual Terminal Support

and others.

Together with the definition of services, ISO also defines protocols which are ways of encoding and encapsulating the information being exchanged when a service is used.

OSI was, and still is, being designed to enable interoperability between OSI systems. Interoperability with the plethora of existing non OSI services using various protocols is not its aim and cannot be achieved by it. In this model, in order to achieve interoperability, all systems must run OSI.

X/Open and OSI

X/Open has a stated policy of complying with international standards. The Xtra process has also repeatedly confirmed that users require X/Open to do so. Therefore, X/Open's long term strategy in the area of interoperability is based on OSI.

So far, the OSI standards describe services and protocols. They leave programming interfaces non-standardised as an implementation issue. Since the use of standardised APIs promotes use of OSI-conformant systems, standardised APIs are important not only for application portability but also for interoperability. ISO has left it to the industry to fill this gap in the standard, and X/Open is leading this effort .

Non-OSI Interoperability

For obvious reasons, services and protocols for exchange of information between non-OSI systems will not be specified as standards by ISO. Yet the market would benefit if specifications of at least some of the important non-OSI protocols were published in a vendor-independent manner since that would facilitate emergence of multiple independent implementations and thus increase competition. X/Open provides several such definitions.

Coexistence and Migration

As OSI solutions are gradually being introduced, users are faced with the problem of preserving their investment in the non-OSI technology which is required to coexist with the OSI solutions. Also, organisations that have decided to migrate their networks and users to OSI face specific problems in making such a transition. Planners, managers and users of networks who face issues of coexistence with OSI and migration to it, need advice from the industry on strategies, methods and tools for coexistence and migration of their current networks to OSI. X/Open provides such advice.

Proving Interoperability

Customers' experience makes them automatically suspicious about vendors' claims that cannot be tested either directly by the customer or by an independent third party. Testing schemes and branding of products are therefore important aids to competitiveness in the market.

Since it is limited to a single system, testing of portability is a relatively straight forward matter. Testing of connectivity and interoperability, however, requires a whole host of issues to be addressed.

To start with, a technically good specification, to which a product is to conform, has to be available. Given the importance of OSI, the most serious attempts at conformance testing have been done for OSI protocols. The OSI model provides for seven layers of communications service, each implemented by its own layer and protocol. At each layer, there are service alternatives and protocol options. There are thus many choices to be made in specifying how to interconnect two open systems.

Implementors of OSI products soon realised that, in order to achieve connectivity between products from different vendors, some specific options had to be agreed by the entire industry. They therefore founded regional workshops for implementors:

- OSI Implementors' Workshop (OIW) in the U.S.A.,
- European Workshop for Open Systems (EWOS) in Europe, and
- Asian-Oceanic Workshop (AOW) in the Far East.

These workshops work towards agreements on functional standards, better known as profiles, which they then submit to ISO. When harmonised for all the three regions, these profiles achieve the status of International Standardised Profiles (ISPs).

Given an application and a type of network, a profile specifies a particular set of choices at all levels of the model which:

- completely defines the communication,
- supports the application, and
- works over the type of network.

Thus the user needs only to select the profile corresponding to his application and networking requirements in order to guarantee connectivity between all his equipment as well as with all his communication partners, provided they have selected the same profile.

Conformance Testing

A number of organisations have been founded by the industry to test conformance of a product to an agreed profile, such as:

- Corporation for Open Systems(COS) in the U.S.A.
- OSInet in the U.S.A., and
- EurOSInet in Europe.

Conformance of a product to a profile is normally tested by examining how the product behaves when interacting with a "model" system which has been implemented by the testing organisation.

Testing of Interoperability

Although theoretically possible, in practice it is very difficult to prove that two products will interoperate under all circumstances. Only access to the source code for both, as well as systematic testing covering all the paths leading through each one of the two, would provide a satisfactory answer. Such an approach is, of course, prohibitively expensive and normally impossible for, amongst others, source copyright reasons.

Because of the legal implications and difficulties in determining whose fault it is when two products do not interoperate, vendors are reluctant to give a guarantee of interoperability. However, schemes are being designed for ways of at least registering claims of interoperability between specific products.

The European organisation Standards Promotion and Application Group (SPAG) recently introduced the concept of product profiles and the Process to Support Interoperability (PSI) which encompasses a methodology for the evaluation of interoperability as well as rules for arbitration in cases when two products certified by SPAG as interoperable fail to interoperate.

Instead of testing, vendors sometimes use interoperability demonstrations to convince both themselves and their customers that their specific product will interwork with other vendors' products. For such a demonstration, local networks consisting of dozens and perhaps even hundreds of systems are set up and specific products are run on them (one example is the Connectathon used for demonstrating interoperability of implementations of the Network File System). While giving the most extensive indication of a product's ability to interoperate, this many-to-many exercise has to be repeated every time a new product joins the LAN or a new version of a product appears.

X/Open and OSI APIs

X/Open has produced industry consensus specifications of APIs to several of the OSI Application Services. While the primary benefit from standardisation of APIs is application portability, several of X/Open's APIs actually also facilitate interoperability through definitions of associated packages of OSI object classes accessed and manipulated by these APIs.

The use of standardised APIs also generally promotes use of OSI-compliant systems and thus interoperability .

The relationship between X/Open's OSI APIs and the OSI protocol stack is shown at Figure 2.

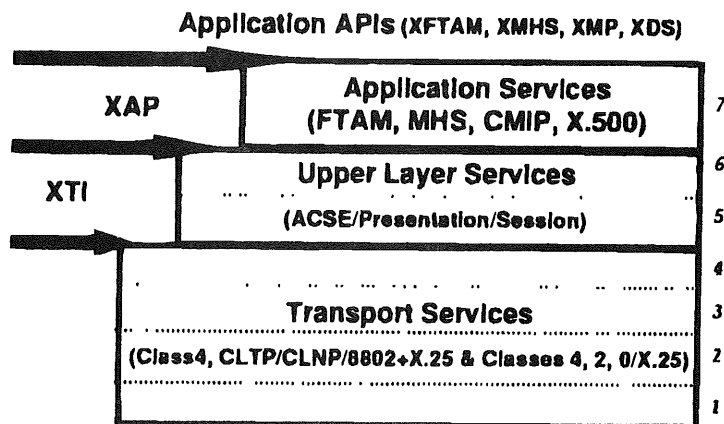


Figure 2 OSI Protocol Stack

Three of X/Open's OSI APIs were recently adopted as base documents for IEEE standards and there is potential for the adoption of more.

The X/Open Common Application Environment currently incorporates the following OSI-related documents:

The X/Open API to Directory Services (XDS)

A directory service provides information about objects in the network. The directory maintains an information base of attributes (e.g. name, address, creation date, etc.) associated with the objects. Objects can be referenced with a naming service or by a set of defining attributes. The directory service will return requested attributes pertaining to the referenced object(s).

The XDS is the interface between a Directory User Agent and an application program (which is the user), by which the application program can access the Directory Service.

The XDS is designed to offer services that are consistent with, but not limited to, the 1988 CCITT X.500 Series of Recommendations and the ISO 9594 Standard.

The XDS has been developed in collaboration with the X.400 API Association.

The X/Open API to Electronic Mail (X.400)

The X.400 APIs specified in this document provide access to, and interconnection of, messaging systems whose architecture is in accordance with the CCITT X.400 Series of Recommendations and the ISO Message Handling System (MHS) Standard. The X.400 Specification defines two interfaces to the functionality of an MHS based on international messaging standards: the Message Access Interface (or Application API) and the Message Transfer Interface (or Gateway API). While they differ from each other in the type of messaging functionality they provide, both interfaces present a model whereby messages, reports, and probes are passed across the interface between the user of the interface, or client, and the provider of the interface functionality, or service.

The X.400 API has been developed in collaboration with the X.400 API Association.

The X/Open API to OSI-Abstract-Data Manipulation (XOM)

Messages, probes and reports are represented at the X.400 and XDS interfaces by data structures called OSI objects. The XOM gives definitions of these OSI objects and of the functions available for creating, examining, modifying or destroying them. The XOM is used, for example, by the XDS, X.400, XMS and XMP APIs.

The XOM has been developed in collaboration with the X.400 API Association.

The X/Open EDI Messaging Package

OSI objects defined by the XOM are categorised on the basis of their purpose and structure into categories known as classes. Related classes can be grouped into collections known as packages. This EDI Messaging Package defines such a set of Electronic Data Interchange object classes. Using the EDI package, objects representing EDI information can be passed across the interface between client and service.

The EDI Messaging Package has been developed in collaboration with the X.400 API Association.

The X/Open Message Store API (XMS)

This interface has been designed for operational interactions with a Message Store. It uses facilities provided by the XOM.

The Message Store Abstract Service is a part of the MHS and is defined in the CCITT X.413 Recommendation. The Message Store acts as an intermediary between the Message Transfer System and the User Agent. Its main function is to accept delivery of messages on behalf of a single end-user and to store the messages for subsequent retrieval by the end-user's User Agent.

The XMS has been developed in collaboration with the X.400 API Association.

The X/Open Guide to Selected X.400 and Directory Services APIs

Using the XOM, XDS and X.400 API definitions is not an easy task for an application programmer with little previous experience of OSI abstract data. This guide has been produced to provide an introduction and tutorial which complements the three API specifications mentioned. The tutorial is illustrated through the use of selected 'C' language programming examples.

This Guide has been developed in collaboration with the X.400 API Association.

The X/Open Management Protocol API (XMP)

OSI provides the Common Management Information Service (CMIS) as a means for performing management operations. Such operations are specified in terms of managed objects, which represent real resources that need to be managed. Objects are described using Abstract Syntax Notation (ASN.1).

The XMP interface provides access to the CMIS service and thus provides a mechanism for the performance of management operations within a distributed environment. XMP uses the XOM interface. XMP also provides access to the Internet Simple Network Management Protocol (SNMP).

The X/Open Management Protocol Profile (XMPP)

To enable interoperability, it is necessary to provide profiles of communications protocols. Profiles specify the use of optional features and ensure that implementations use compatible sets of such options. International Standardised Profiles (ISPs) exist for OSI management.

The XMPP specification refers to these ISPs and thus requires that conformant implementations conform to these profiles.

It also refers to the relevant Internet specifications for the SNMP environment.

X/Open and Interoperability with Legacy Systems

From the commercial point of view, the following are the most important legacy system types to consider:

- Internet Protocol Suite (IPS) systems;
- Personal computers;
- IBM-compatible mainframes.

X/Open has been addressing problems of interoperability in all of these three areas.

The Internet Protocol Suite

The X/Open Guide to IPS (XGIPS)

The Internet Protocol Suite (IPS), popularly known as "TCP/IP", is the current de-facto standard for interworking between non-OSI multivendor systems. It is not a formal standard and its documentation in the Internet Requests For Comments (RFCs) does not have the rigour of a formal standard. It also allows a number of options for implementation. Since implementors of IPS have chosen different sets of options in their specific products, it is not easy to give general advice to users who wish to migrate from IPS to OSI. X/Open has therefore produced the Guide to IPS which describes which IPS functionality can be found in the majority of current commercial IPS implementations. This description also defines the IPS platform from which most of users will be migrating to OSI or at least coexisting with OSI.

The X/Open Guide to IPS-OSI Coexistence and Migration (CoMiX)

This Guide was produced by X/Open in order to help managers, network planners and implementors, and users to understand the issues involved when migrating from IPS to OSI. CoMiX gives first a number of real-life scenarios derived from market research through which the various user objectives and problems are illustrated. Then, descriptions are given of the functionality available in IPS on one side and in OSI on the other side, so that a comparison can be made. (For the description of IPS functionality, CoMiX refers to XGIPS, the guide described above.) Users are further advised on how to design their strategies. Chapters on techniques and tools describe the technical means of use for coexistence and migration. Finally, application of the techniques and tools on the scenarios provides a sanity check for value of the advice provided.

The Byte Stream File Transfer Definition (BSFT)

One of the most used services provided by IPS is the File Transfer Protocol (FTP). In order to ease the migration of users from an IPS-based network to an OSI-based one, X/Open has developed the BSFT definition which preserves most of the interface and functionality of FTP but uses the OSI FTAM protocol profiles underneath. Thus, BSFT simplifies the simultaneous use of file transfer facilities in the IPS and OSI environments and facilitates the coexistence and migration between these two protocol sets. BSFT is fully conformant with the ISO/IEC Standard 8571, Information Processing System--Open Systems Interconnections--File Transfer, Access and Management. It uses the International Standardised Profile ISP 10607:1990, FTAM, Part 3 (AFT11 - Simple File Transfer Service) and part 6 (AFT3 - File Management Service).

The X/Open Network File System (XNFS)

The most widely used architecture for heterogeneous transparent file access between system running IPS is the Network File System, originally developed by Sun Microsystems, Inc. X/Open has recognised the position of NFS as a de-facto standard and published this specification as a temporary but complete solution to the problem of transparent file access between X/Open-compliant systems.

The XNFS specification defines:

- the transparent file access service provided by XNFS,
- the protocols that support this service between X/Open-compliant machines, which can take the role of either servers or clients, and
- the differences in semantics of the XPG3 Volume 2, System Interfaces and Headers, and Volume 1, Commands and Utilities, when they are used "transparently" over the network using XNFS rather than locally.

The X/Open Management Protocol API (XMP)

The Internet Protocol Suite provides the Simple Network Management Protocol (SNMP) as a means for performing management operations. XMP provide access to the Internet Simple Network Management Protocol (SNMP) as well as to the OSI Common Management Information Service (CMIS).

The X/Open Management Protocol Profile (XMPP)

In order to provide interoperability, it is necessary to provide profiles of communications protocols. Profiles specify the use of optional features and ensure that implementations use compatible sets of such options.

The XMPP specification refers to the relevant Internet specifications for the SNMP environment and thus requires that conformant applications should conform to these specifications.

Personal Computer Interworking

Most of the personal computers today run the DOS or OS/2 operating systems which do not conform to X/Open specifications. In local area networks, such systems can be connected as clients to an X/Open-compliant server. X/Open has therefore developed specifications of protocols and interfaces which can be used by such servers.

For Local Area Networks with personal computers, two distinct market segments have been identified by X/Open, one where personal computers are being integrated into an existing network of X/Open-compliant systems which are already running XNFS, the other where X/Open-compliant servers are being added to an existing Local Area Network consisting primarily of personal computers. Two different solutions have been specified for these two markets.

Protocols for X/Open PC Interworking:

(PC)NFS

Where personal computers are being integrated into an existing network of X/Open-compliant systems which are already running XNFS, the (PC)NFS protocol is used. Its specification is a specific version of the XNFS one; it defines the protocol for communication between a PC client running DOS or OS/2 and an X/Open-compliant server.

This version of XNFS specifies only those aspects required to implement a server for a single-user system client, and specific attention is given to issues unique to PC interworking, including authentication, remote print spooling extensions, and DOS file sharing and locking support.

Protocols for X/Open PC Interworking: SMB

The Server Message Block protocol, originally developed by MicroSoft Corporation, is intended for use where X/Open-compliant servers are being added to an already existing Local Area Network consisting primarily of personal computers.

IPC Mechanisms for SMB

Since systems in networks running SMB do not use interfaces defined in XPG, this specification defines interfaces to Inter-Process Communication, covering named pipes, mailslots and messaging. It further defines the necessary Server Message Block protocol extensions for interprocess communication.

For interoperability via asynchronous serial links, X/Open has defined in XPG3 Volume 7 a file transfer protocol identical to the Kermit protocol, as well as a set of features provided on X/Open compliant systems for terminal emulators.

CPI-C

For communication between X/Open-compliant systems and IBM-compatible mainframes, the protocol most in use is the proprietary SNA Logical Unit 6.2. However, in order to ease integration between open systems and such mainframes, X/Open has published the specification of the Common Program Interface Communications (CPI-C), an API based on IBM's definition.

X/Open Transport Interface (XTI)

One of the solutions to the problem of coexistence and migration of networks using different transport providers is to use in applications running in such networks an API that is independent of any specific transport provider. The XTI has been developed for such a purpose. While it is concerned primarily with the ISO Transport Service Definition (connection-oriented or connectionless), it may be adapted for use over other types of provider, and has indeed been extended to include TCP, UDP and NetBIOS.

X/Open and Distributed Computing

X/Open has recently developed the X/Open Distributed Computing Services Framework (XDSCF). The XDSCF is a comprehensive blueprint for a complete system environment that will allow open systems to better address the critical needs in enterprise-wide heterogeneous distributed computing.

The framework specifies:

- the required services;
- the relationship between services;
- the programming interfaces to services; and
- the protocols and data formats that provide interoperability between systems that support the services.

The Framework is organised as a set of four layers, as shown in Figure 3. Each layer providing a level of functionality in an enterprise computing network:

- The Operating System Services provide an environment for running distributed software on each node of the network.
- The Communication Services provide the services that allow applications to communicate reliably across the network independently of the underlying network topology, networking protocols or data representations.
- The Distribution Services provide a set of services that support consistency in distributed applications. These services address the fundamental issues in computing, including the processing model, the naming model, the security model and the management model.
- The Application Services provide the enabling distributed system software to support the development of distributed applications.

While mainly concentrating on interoperability of open systems, the Framework also includes services provided for interoperability with legacy systems.

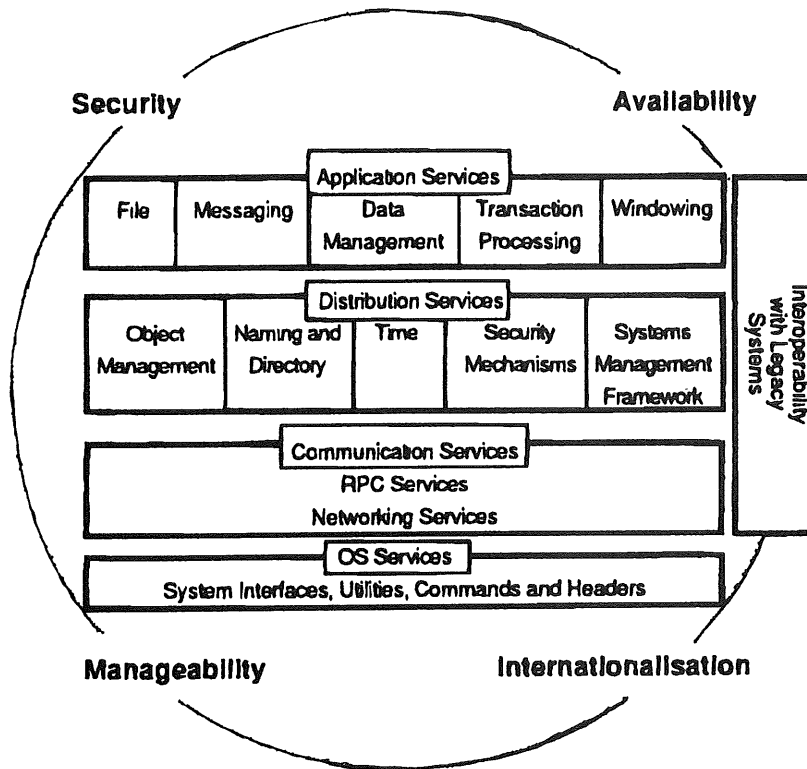


Figure 3: XDCS Framework

Author: Petr Janecek has a doctorate in theoretical physics. After some years at the Nordic Institute for Theoretical Atomic Physics he was employed by ASEA, Sweden on a number of engineering projects which brought him to be closely involved in computer processing techniques. He worked with Ericsson Information Systems as Manager, Software Development Environment and became the company's X/Open Technical Manager. In a broader role with Nokia Data, as the company's Director of Strategic Planning and Architecture, he chaired two X/Open working groups on data management and PC interworking as well as acting as the Nokia Technical Manager working with X/Open.

An Update on UNIX-Related Standards Activities †

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

This report is somewhat shorter than usual, not least because as ;login: goes to press, the IEEE POSIX Spring meeting is taking place, and few of our regular reporters have had time to put finger to keyboard. Stephe Walli, standards report editor for USENIX for the last year, has stood down, and a new fool has been found, too innocent to realize what he was taking on, in me, Nick Stoughton.

I am unashamedly British, and if some of the spelling, punctuation, and use of English looks strange to you, that's because it is correct! Otherwise, I hope none of you notices much change in the style of what is reported here. Having been in the UNIX field since the early days of 6th edition (remember the Lions book?) in 1976, I have been involved with almost every aspect of the operating system, both from an academic point of view, at Queen Mary College (as it was then) of the University of London, and latterly in a commercial services point of view, with Hoskyns Group, also in London.

First Impressions

However you look at it, Stephe is a hard act to follow. Those of you who have been following his comments recently on Test Methods and Language Independent Specifications will know what I mean!

The next installment in these great battles dominated the April meeting in Irvine for many.

As a newcomer on the scene, many people asked me about my first impressions of a POSIX meeting. Imagine two great armies facing each other across a battlefield. Rather than let each side attack the other till one is the clear winner, the whole encounter is far more formalized, more like a game of chess. Of course, this drags out the battle, but is much fairer (and of course politically correct)!

How many standards committees does it take to change a light bulb? Forty two ... one to specify the mechanism for light bulb exchange, one to write a language-independent specification, one to develop test methods, a screw-cap language

binding committee, a bayonet-cap language binding committee, a subgroup on bulb-exchange methodology, a working group on bulb disposal, then of course there's real-time bulb exchange ... hey, this sounds like a great new area for POSIX!

LIS Wars III

The real battle of the week was the Language Independent Specification (LIS) issue. In January, an Ad Hoc Committee, chaired by Stephe Walli, was established to research the whole issue of LISs. The Ad Hoc had spent considerable time between the meetings canvassing opinions, and it was clearly a hot potato. Some countries really believed them a major issue, even if it took twenty years to get there! On the other hand, the real users of POSIX standards, people like you and me, want standards as soon as possible, and we want them written in a language that we can go ahead and use.

The Ad Hoc met frequently during the Irvine meeting, and was well attended. All views were aired, and a consensus reached. Even writers of existing bindings in other languages (FORTRAN and Ada) recognized that the existing, non-LI, methods of deriving their work had worked, and worked well. So a very carefully worded report was constructed to present to the SEC. With Stephe chairing it, what would you expect?

It recognized that the goals of a LIS were good, but that the tools were not yet appropriate for the task. I have lost count of the number of times in my working career where I have been asked, metaphorically, to use a wrench to undo a screw. LIS is just another wrench, a programming language version of Esperanto. Indeed, although it wasn't suggested, if we are to write a truly language-independent standard, the portions currently in English should be rewritten in Esperanto. Actually, thinking about it, I believe it was suggested once, but never mind. A large amount of time and effort has been spent in POSIX over the last three years trying to get LI to work, but we are still not much further ahead than when we started.

The report went on to make the explicit resolution to drop the requirement for LIS, allowing working groups to use it only if they thought it appropriate. It further proposed a resolution to provide an alternative means of ensuring semantic equivalence between different language bind-

† This is a re-print from ;login, the USENIX Association Newsletter, Volume 18 Number 3

ings. The original version, written in a specific programming language (normally C, but not necessarily), should contain conformance rules for future language bindings.

The SEC debated this report at length. Issues such as what to do with existing LI work, in particular the POSIX.1 LIS work, would need very careful consideration.

The chair of the United States Technical Advisory Group to ISO Working Group 15 (better known as the WG15 US TAG), asked that the decision on LIS be deferred until July. WG15 itself meets in Heidleburg in May, and it was WG15 that placed the LIS requirement on the US National Body in the first place. If the US TAG were to present a unilateral fait accompli, their position would be much weakened. Several members of WG15 had made their views known to the Ad Hoc, and were aware of the likely outcome. It was quite clear that in July, the weight of opinion would be behind accepting the report of the Ad Hoc, and that SEC would cease to require LISs. The US TAG delegation were made aware of this. In the end, by the narrowest of majorities, it was agreed to defer the decision. Whether you view this as three more months, or just another three months depends on you point of view.

It is clear from the falling attendances at the POSIX meetings (down to approx 220 from over 300 last year) that many other people feel that far too little happens far too slowly at these events.

The Little Red Standards Group

The following text appeared anonymously on a notice board at the Irvine meeting during the "great LIS debate." It is reproduced here for your enjoyment. Although it represents one person's view, other conclusions are possible!

"The Little Standards Group"

Once a hard-working little standards group lived with a group of international standards organizations, WG15, SC22, and JTC1. One day they decided to make a standard for them all to share.

"Who will help me start the standard?" asked the little standards group. "Not I," said WG15 sheepishly. "Not I," bowed JTC1. "Not I," grunted SC22. "Then I will do it myself," said the little standards group.

When she had cut the interface, she asked "Who will help me edit the standard?" "Not I," growled WG15. "Not I," smiled JTC1. "Not I," snorted SC22. "Then I will do it

myself," said the little standards group.

When the standard had been edited, the little standards group asked "Who will help me ballot the standard?" "Not I," sighed WG15. "Not I," sniffed JTC1. "Not I," whined SC22. "Then I will do it myself," said the little standards group.

Soon the wonderful smell of a freshly baked standard filled the community. "Now," said the little standards group, "who will help me use the standard?" "Not so fast," cried WG15. "Before we can use our wonderful standard, we must first have an LIS for the specification." "Yes!" cried JTC1. "Absolutely true," said SC22. "I can understand that," said the little standards group, "So, who will help me create this LIS?" "It's your standard, so you must do it," said WG15. "This is reasonable," said JTC1. "Naturally it is your task," said SC22.

The little standards group worked on the problem for a while, but was not able to make progress. So the little standards group went back to the others. "I don't think I can do this LIS thing," said the little standards group. "And I didn't even want to do this LIS thing in the first place. Even though you want it done, you are unwilling to help. Why should I do this thing for you?" asked the little standards group. "You cannot call it a standard until we say you can," said WG15. "Regrettably true," said JTC1. "We agree," chortled SC22.

The little standards group said "Fine. Then it will not be a standard for you; it will be a standard for me. When you are ready, I will work with you to create this LIS. Until then, I guess you'll just have to do without." So the little standards group produced useful and portable applications, while WG15, SC22, and JTC1 hungrily watched.

Developing Standards in the Absence of Test Methods – or – What if you threw a party and nobody came?

Many years ago, the POSIX gods decided that it was important for POSIX standards to have test assertions. The purpose of these test assertions was to ensure that implementations of the POSIX standards could be fairly evaluated by potential system purchasers. Unfortunately, the task of creating test assertions, much like the task of testing software and hardware in every organization, was viewed as an onerous, menial task that was just not exciting. At the April POSIX meetings, the POSIX gods rescinded their test assertions

requirement because of considerable pressure from the community that develops the POSIX standards. In this article, I will attempt to evaluate the impact of this fundamental change in the way in which POSIX standards are developed. When reading this article, please keep in mind that it is an editorial, and is one person's opinion. There are a number of other, perhaps better informed opinions in this area. With luck they too will write about this topic.

Why did POSIX develop Test Methods?

Testing is essential to determining the quality of any product. That is not under dispute. While things like Total Quality Management are currently under close scrutiny to determine their overall benefits, the fact remains that without testing it is all but impossible to be confident that the product you are distributing will work to the satisfaction of your customers.

In the open systems community, testing usually takes place during the development cycle. However, large procurement groups (like the US federal government) may require additional, post-production testing to ensure that systems conform to open systems standards. This "conformance testing" is usually done via some sort of test suite that the procurement group has chosen.

This need for conformance testing within the open systems movement has caused a segment of the test development community to focus on developing tests against open systems standards. In fact, the success of POSIX has caused some testing organizations to come into being. These organizations develop conformance tests against the open systems standards, many times in competition with one another for a very limited market. In order to evaluate the quality of a test suite, a potential test suite customer must have some metric for determining that the suite actually tests the standard in question. Hence, test assertions.

Test methods are, in general, simple statements that describe the behavior of an interface under certain conditions. (Note that this is an over-simplification, but is generally accurate.) By agreeing on the test methods for a given interface, the open systems community in effect gives guidance to test suite developers as to the tests that should be performed to ensure conformance to the interface. As mentioned above, these methods can also be used by the community to evaluate the quality of the tests that the test suite developers produce. This is particularly useful for organizations who, like the US federal government, must use objective metrics to evaluate products prior to procurement. Just as computer systems must

be tested to see if they conform to standards, the tests to test the systems must be tested.

So what's the problem?

So far this seems pretty straightforward, right? Unfortunately, the model breaks down because of a market that is too small. While the market for open systems is roughly *really big*, the market for test suites is pretty small. Further, the market for test suites that have to be objectively evaluated against other, competing test suites in the same space is even smaller yet! Consequently, while the market pressure to develop open system standards is very great, the market pressure to develop test assertions is almost nonexistent.

What does this mean? Let's consider an example that typifies traditional POSIX committee patterns, as opposed to ideal behavior. When POSIX committee A starts their work, they have a base interface specification. There is considerable enthusiasm for the work, and it proceeds quickly (in standards terms). Eventually, the draft standard is complete, and is sent out to a broader group of people for formal balloting and approval as an industry standard. This standard represents an interface that is important to the community, and in particular important to the companies that funded their employees to attend the standards meetings and work between meetings to complete the document. It is, in turn, probably important to those companies' customers, since the cost of participation is high enough that it usually requires some sort of higher-level management approval.

At this point in the process, the members of the group that did the work are effectively out of the loop. Balloting is a process that does not generally involve group activity. However, there is more work to be done. The standard must have test assertions developed. Therefore, the members of the group that have just completed work on the draft standard need to start developing a new document that is effectively a detailed design description of the document they just finished. Moreover, this description has to be made in a somewhat formal manner, using stilted, precise, language. On top of this, the document they now must develop is one that most of their management and customers don't need. Consequently, many participants who worked on the initial document suspend their participation until there is some other task in which their sponsor is more interested.

What was the solution?

As I mentioned above, that was an example of a more traditional POSIX activity. Recently the POSIX working groups have come to grips with

this trend by changing their development model to one in which the test assertions are developed at the same time as the interface specification. This results in a higher-quality specification, fewer changes in the specification during balloting, and of course, the test assertions being available in the same time frame as the specification itself. That's the good news. The bad news is that the standard itself takes somewhat longer to develop in the first place, and in the end has all of this stuff in it that wasn't really needed by the majority of the standard's users.

Other groups have addressed the problem by finding some industry group with deep pockets to do the primary development work on the test assertions, and then ratified the work through the working group process, but in a rapidly accelerated fashion. In the three examples of this within recent POSIX memory, the work has resulted in all the benefits of the previous example, but without the disadvantage of taking significantly longer than the development of the base standard. (To be fair, the base standard in these three cases was also developed by an external industry group.)

In both these cases the working groups managed to retain most of their participants throughout the process, and to produce standards and test methods that are reasonably high-quality without delaying the introduction of their standards overly much. So, with the introduction of these more ideal working models, the POSIX community has discovered a way to develop the required test assertions, albeit with some cost to the industry.

What are the implications of not requiring test assertions?

So, what have we come to? Test methods are useful tools for improving the quality of a standard under development. However, test assertions are expensive, and the formal standard version of the test assertions only serves a very small community of "users" — those who develop test suites for open systems standards and those who need to do objective procurement of open systems test suites to use as tools for evaluating other objective procurements.

Given these assumptions, what does the POSIX gods' decision to no longer require test assertion development do to the quality or availability of POSIX standards? My guess: not a lot. POSIX has traditionally provided a useful service, and has not yet succeeded in making itself irrelevant. (Many other standards groups have.) Conse-

quently, POSIX will continue to produce standards for which the market exhibits a need. This development may, or may not, involve the specification of test assertions along the way. Those groups that see utility in developing the test assertions will do so. If the market can agree on the need for test assertions for a given standard, the market may cause those test assertions to be developed. If a large procurement group needs to procure a test suite that tests a standard for which there are no test assertions, it may define the test assertions itself. In any event, the community that has the need will fund the development. In the interim, the standards the market requests will continue to be developed to the level of quality that the industry has come to expect.

Report on POSIX.0: POSIX Guide

Kevin Lewis <klewis@gucci.ent.dec.com> reports on the April 19 – 23, 1993 meeting in Irvine, CA:

Let me say right up front that this was quite a productive week for the group. Our primary goal was to achieve in excess of 75% in our affirmative ballots so as to move into recirculation prior to the next meeting in July. The group was successful in achieving this goal. The chronology is as follows:

- Initial number of affirmative ballots received was 28 out of 58, placing the percent affirmative at 48%.
- The group converted 7 ballots to affirmative prior to the April meeting.
- The group converted 13 ballots to affirmative during the April meeting.
- This places the current percent affirmative at 82% (48 out of 58).

The issue of public specifications, expected to be a highly significant issue, proved to be exactly that for only a small number of balloters (5 out of 58, 3 of whom could be considered negotiable).

The POSIX.0 group conducted a Birds-of-a-feather (BOF) session on this issue of public specifications to ensure that any balloter and any one else with a concern in this area had an opportunity have a dialogue with Dot Zero to ensure an effective exchange of information. Our main concern prior to this BOF was that the way POSIX.0 sees public specs and their use and the way everyone else sees this issue might be at odds.

It became evident after the BOF that the understanding on this was mutual. In fact, there were a very small number of people (3 out of about 20

that attended) who had any major concern with this topic.

The ISO WG15 Rapporteur and the SC22 Secretariat representative were present during the BOF. I queried them on whether or not there was any concern expressed on this issue at their respective levels within ISO. They both indicated that each group was aware of the presence of this topic in the POSIX.0 guide and no one had expressed any concern.

Given that POSIX.0 has reached the goal enabling a recirculation, this will be coordinated with the IEEE with the objective of having this next phase start prior to the July meeting in Denver. The group will be in recirculation during the July meeting. So our discussions will focus on possible future projects, to include a guide in the area of Transaction Processing and an IEEE video conference that would serve as a tutorial about the use of the guide.

Report on POSIX.4: Real Time

Bill O. Gallmeister <bog@lynx.com>, vice chair of the POSIX.4 working group, reports on the January 11-15, 1993 meeting in New Orleans, LA:

POSIX.4 has, for all intents and purposes, split into two separate groups. On the one hand, we have the existing "old" work of POSIX.4: POSIX.4 itself, POSIX.4a (threads), and POSIX.13. These three documents are in balloting right now and are not, for the most part, the concern of the working group. On the other hand, the working group is busy, mostly with POSIX.4b, a proposal for extended real-time functionality. POSIX.4b is what the working group worked on in New Orleans. This report will briefly cover both parts of POSIX.4.

What We Work On

First, here's a brief introduction to what we do in POSIX.4. POSIX.4, the working group charged with making extensions to POSIX to support real-time applications, has a sizable stable of proposed standards in its estate. POSIX.4 is the basic real-time standard; POSIX.4a is the threads extension to POSIX; POSIX.13 is the profiles document that describes POSIX for everything from an embedded controller to a large workstation-based real-time system; POSIX.4b is yet more real-time extensions (stuff we couldn't agree to work on for POSIX.4); and finally, POSIX.4c is the Language-Independent specification and the test assertions for POSIX.4.

The Old Stuff: POSIX.4.

POSIX.4 is the base, real-time standard. This document is so close to finishing, we can taste it! On the last ballot, we achieved an 83% affirmative approval rating. By that metric, we're done. On the other hand, some of the remaining balloters are vociferous and represent large, existing bases of UNIX functionality. For this reason, in New Orleans the technical reviewers were still addressing the remaining objections, trying to remove as many of these as possible. At the close of the New Orleans meeting, the ballot resolution process wasn't completed. However, since then, the resolution process has been finished, and we have in fact sent out POSIX.4 for its final recirculation. Two large changes were made from draft 13 to draft 14, along with several minor changes. The major changes are:

1. The real-time files chapter has been removed from the document. This chapter had become the lightning rod for the majority of the remaining objections, most of which objected to the fact that the facility only seemed able to do contiguous, pre-allocated disk files. More capability and extensibility was desired. A competing proposal, for a thing called *fadvice*, has been made by one of the objectors, and it seems like a better solution to the whole issue of real-time files. We will probably be addressing this area in future work of the POSIX.4 working group. Unfortunately, for now there is no proposal in place for real-time files. I was personally uncomfortable with this action, it being late in the balloting process. At the same time, though, I do like the *fadvice* interface more than the POSIX.4 Draft 13 interface, and some of the Draft 13 facilities were, frankly, incomprehensible to me. Basically, I think this action was OK.
2. Some of the POSIX.4 interfaces feature the capability to set up "something" that will cause the application to be asynchronously notified in the future. For instance, a timer expiration, or asynchronous I/O completion, both result in asynchronous notification. As of Draft 13, "asynchronous notification" meant a signal was delivered. Several objectors wanted the ability to replace signals with other, presumably more lightweight methods of asynchronous notification. (Simply calling a function is a possibility.) Draft 14 has been accordingly modified to support extension to new methods of asynchronous notification. Signals are currently the only known method of asynchronous notification, but other, future

mechanisms can now be implemented in a portable fashion under the POSIX.4 facilities. I am quite in favor of this change, as everyone knows there are faster ways of doing asynchronous notification than signals!

In addition, there are numerous, very minor changes to Draft 14 of POSIX.4. At this writing, POSIX.4, Draft 14 has been sent out for one last recirculation. Draft 14 is not a complete draft; it is merely a set of changes from Draft 13. There are about ten pages of changes. The balloting period has already closed, and we are in the process of totaling up the responses to this last recirculation.

More Old Stuff: POSIX.4a

POSIX.4a is the threads proposal. This document is probably of greater interest to a lot of the USENIX members than POSIX.4 itself. Here's a shocker: POSIX.4a has gone out for a recirculation! After a long while in ballot resolution, the threads proposal was just recently sent out again. It should be in the hands of the balloting group for another recirculation before this report is published. With any luck, this recirculation will go more smoothly, and more swiftly, than the previous two recirculations have. The good news is, this time, POSIX.4a is being recirculated, not reballoted. The previous time around, POSIX.4a was reballoted. What that means is, the entire draft was open for comment. In a recirculation, by contrast, only the changed parts of the proposal can be commented upon. A reballot is required when not enough people deliver an affirmative ballot. Thus, if you need a reballot, you're in trouble. The fact that we are recirculating POSIX.4a is a good sign.

Yet more Old Stuff: POSIX.13

The profiles document, POSIX.13, is currently still in ballot resolution. It seems to be following the POSIX.4a model, wherein the technical reviewers have very little time for technical review. The issues for POSIX.13 are fewer than for POSIX.4a – that's good. On the other hand, the one big issue, subsetting of POSIX.1, is still to be addressed. That's bad. (For those of you who are just learning about the POSIX.4 world, POSIX.13 consists of four profiles, ranging from supercomputer real-time all the way down to tiny, embedded systems. These embedded systems generally run on hardware that is not capable of supporting all of POSIX.1 and POSIX.2 (no disk, no MMU, very little memory, etcetera). For that reason, there is a need for an embedded profile to call out a subset of POSIX.1: it needs a lot of POSIX.1, but other parts are simply irrelevant

or impossible. Stay tuned for more information on how POSIX.13 is doing!

POSIX.4c

Some progress has been made towards a language-independent specification for POSIX.4. However, given the current instability of the whole LI thing, I wouldn't be surprised if we were able to drop this work later on. For now, work continues on LI.

New stuff: POSIX.4b

POSIX.4b is a collection of fairly exotic stuff: timeouts on blocking functions, direct application access to interrupts and different types of memory, sporadic server scheduling, and so on. The facility that is of interest to the mainstream community here is a thing called *spawn*. *Spawn* combines *fork* and *exec* into a single call that *spawns* a new process. This is useful for systems (see above) that cannot support a separate *fork* (because they have no MMU for duplicating an address space), but which can support separate processes (because they have enough physical memory for it). *Spawn* is interesting to the mainstream community because, in general, a *fork* is followed by an *exec* anyway. *Spawn* would be a nice way of combining things.

Other interesting work is *devctl*. *Devctl* is really *ioctl*, but we had to rename it so as to not break existing systems.

During the week in New Orleans, small groups addressed various issues in the chapters of POSIX.4b. An interface for various sorts of typed memory (static RAM, different buses to access the same memory, and so on), was part of the proposal, but due to lack of interest and commitment in this proposal, it was removed from the proposal. Other changes were less drastic. We hoped, at New Orleans, to take the document out for balloting after the Irvine meeting. I personally have my doubts that this will happen. I think it will take a couple more meetings to mature the POSIX.4b functionality sufficiently. On the other hand, POSIX.4b has been substantially written (in Language Independent form plus C bindings, even!), so I could be mistaken.

Report on POSIX.9: FORTRAN Bindings

Michael J. Hannah <mjhannah@sandia.gov>, chair of POSIX.9 reports on the January 11–15, 1993 meeting in New Orleans, LA:

The POSIX.9 (FORTRAN bindings) group, though sparsely attended, did meet in January and made progress towards their next project. While other IEEE standards have been drafted by three people, this is uncommon for POSIX. A committee of such small size implies a very significant reliance on the

with a small group doing the draft, but before such a draft becomes a standard it must be reviewed and examined by a much larger, representative, balloting group. While there may be nothing improper or illegal with this approach, I would certainly like to see more participation in our meetings.

IEEE Std POSIX.9-1992 is approved and was available for purchase at this meeting. This standard completely defines how to access *all* functionality of ISO/IEC 9945-1:1990 from FORTRAN 77, as well as defining a generalized way to access any operating system structure and defining byte-stream I/O for FORTRAN 77. Since FORTRAN 77 is essentially a subset of the new Fortran 90 standard, IEEE Std POSIX.9-1992 is completely usable in a Fortran 90 environment. Like any standards committee that just completed a standard, the committee discussed how to convince vendors to implement this standard, and how to convince users to demand this standard from the vendors.

Actual work was begun on draft 0 of the next project for this committee, POSIX.19, which is a binding between the Language Independent Specification (LIS) of POSIX.1 and the new Fortran 90 language standard. This Project Authorization Request (PAR) was approved by the IEEE standards board in Sept 1992, though approved over a year ago by the SEC. Part of the delay was ensuring complete liaison with X3J3, the FORTRAN Language committee. At their August 1992 meeting, the X3J3 approved an official resolution endorsing the scope of the POSIX.19 PAR and agreed to active liaison with the POSIX.9 committee. This is significant since the POSIX.19 PAR includes in its scope the ability to define extensions to the base language standard of Fortran 90. One of the major unresolved objections in balloting IEEE Std POSIX.9-1992 was that many of the functions could have been defined as simple extensions to the base standard syntax. For example, mode bits could be included as an extension to the FORTRAN OPEN statement, etc. Such an approach is planned for the POSIX.19 work.

The committee began by discussing the overall approach to the new work. In addition to examining the new language features in Fortran 90, the committee discussed how this new binding should relate to the POSIX.1 LIS and its companion POSIX.1 C binding. While this POSIX standards body is focused on portability of applications, as an end user I am particularly concerned about portability for application writers. Whether to point to the LIS or point to the "his-

torical practice" of C is a contentious issue. For example, the LIS describes a procedure called `change_file_mode`, while the traditional C function is called `chmod`. In IEEE Std 1003.9-1992, because any function/subroutine name in FORTRAN 77 was exposed to the loader, and since the IEEE Std 1003.9-1992 routine to change file mode had to be slightly different than `chmod`, we had to give it a distinct name, `PXFCHMOD`. Because of the new features of Fortran 90, module names used by an application are not necessarily exposed to the loader. Thus we *could* now call the Fortran 90 routine `chmod` without fear of conflict with a different C library routine of the same name. Using `chmod` as the name is intuitive to any programmer coming from the C world, but is not intuitive to a strictly FORTRAN programmer. While the argument in this example may be stronger for `chmod` since there is also a POSIX.2 command by that same name, such an argument does not apply to all POSIX.1 functions. If you really believe in the concept of the LIS, and especially if the new C binding is "thin," then a name that is the same as the LIS `change_file_mode` might be more appropriate. A Fortran 90 bindings reader should need at most the POSIX.19 binding and the POSIX.1 LIS. However, many LIS names are more than 31 characters, so some mapping may be required, or an extension to Fortran 90 to recognize uniqueness in names longer than 31 characters. This seems to be something like a religious issue, where parties on each side are *certain* that their position is correct, and the only intelligent position. The current committee default is to use the long LIS names, not the familiar C names, but this may change.

One item of interest is that new constructs in Fortran 90 seem to permit the binding to be completely specified as Fortran 90 code fragments. Whether the IEEE and ISO document structures could be accommodated by this is unclear. For an implementation, you would like to give an electronic copy of the binding (code fragments) to the implementors so they could simply add the rest of the code necessary on their implementation. Since the code fragments completely define the application interfaces, this would be a boon for everybody. However, such a scheme also raises the question among the lawyer folk as to what this would mean with regard to the IEEE copyright of the binding.

Finally, the committee was actively involved in the hot debate concerning whether the POSIX Executive Committee should rescind its mandatory requirement for base POSIX standards to be developed as Language Independent Specifications (LIS). As a language binding committee we

are viewed as the direct beneficiaries of the work to produce an LIS. The members of the bindings committee of both the POSIX.5-Ada and POSIX.9-FORTRAN have strong opinions on this issue.

The POSIX.9 committee is scheduled to meet the first three days of the April POSIX meeting.

Report on POSIX.18: POSIX Platform Environment Profile

Paul Borman <prb@cray.com> reports on April 19 - 23, 1993 meeting in Irvine, CA:

The Reduction Continues

At the April meeting in Irvine, the POSIX.14 group dedicated one day to the POSIX.18 Draft. It was much easier to work on the draft this time, mostly due to its reduced size. As before, the major work done on the draft was reducing the number of words.

First of the areas we attacked during this meeting was the introduction and scope. We decided that even though the content was basically hidden in there someplace, we would do best by just rewriting the introduction and scope instead.

The next section of the document looked at was the definitions section. After reviewing the definitions of conformance, we moved them to the conformance section of the document. We also removed several definitions that were either defined in one of the base-level standards we reference, or were actually simply definitions of English words, such as "portability."

In the actual normative text, we moved some of the pieces in the "Language Interoperability" section to our "Coherency" section. This was done to clarify and not change content. The only actually substantial change was to remove the FIPS 151-2 requirements for CS7, CS8, CSTOPB, PARODD and PARENB, which were added at the last meeting. It was brought to our attention that this was required by NIST to facilitate their RS-232 loop back test. We decided it was not appropriate for this profile to require a particular hardware interface.

We had some discussion on the Fortran Language option when we realized that as the document stood we referenced FORTRAN 77, specified Fortran 90 and required a binding to FORTRAN 77 (POSIX.9). Although we are not sure what to do for the final draft, we are sure that it should be consistent. The issues include:

1. POSIX.9 currently refers to an ANSI standard (FORTRAN 77) and is not an international standard.
2. The International standardized version of FORTRAN is Fortran 90, which is not as widely used as FORTRAN 77,
3. This profile is intended to be forwarded to ISO.

The options that we see ahead of us include:

1. Drop the Fortran Language option (not desirable).
2. Have two Fortran Language options (though only the Fortran 90 option would probably make it through ISO.
3. Wait for a resolution between POSIX.9 and Fortran 90, then do what seems appropriate.

Finally, we actually removed all references to test methods from the document. The SEC has rescinded the requirement for test methods and we had been spending too much time on it without having satisfactory results. This also saves 5 full sheets of paper (10 pages).

Once the new editing job has been done, we will probably be basically ready to go to ballot, but we will have to wait because our document depends on both POSIX.4a and POSIX.1a.

Report on POSIX.19: Fortran-90 Bindings

Michael J. Hannah <mjhannah@sandia.gov> chair of POSIX.19, reports on the April 19 - 23, 1993 meeting in Irvine, CA:

This was an important meeting for anyone following the subject of FORTRAN language bindings to POSIX. After two years of effort to drum up interest in a Fortran 90 binding, the POSIX.9 Working Group proposes to call it quits. The few folks who are left believe that there is an insufficient body of knowledge, practice, or users of ISO/IEC 1539:1991 (Fortran 90) to sustain the effort of producing a POSIX binding at this time, especially in light of a number of outstanding technical issues. Many of these issues concern trying to determine how best to use the new features of the Fortran 90 language in a POSIX binding. However, in a spirit of one last time, the Working Group postponed until July the final act that would disband the Working Group. At the July POSIX meeting, the Working Group will meet for one day only, Monday, July 12. Barring a large turnout desiring to retain the Working Group, the Working Group will recommend that the executive committee of POSIX withdraw sponsorship of the Fortran 90 binding project and disband the group.

The group is circulating a draft of a final report among members who have participated in the effort so far, and will present the completed final report at the July meeting.

The probable demise of this working group raises several questions about POSIX and language bindings:

1. How many different languages are likely to bind to POSIX? If this is only a few, does this imply that POSIX is less valuable?
2. Is POSIX just for C and Ada, and all other languages should simply figure out how to call system routines in those languages? Does this make those other languages second class citizens in a POSIX world?
3. If there are to be future language bindings, should the IEEE with its POSIX steering committee sponsor that work, or should the committees that define and standardize the

language (usually part of ANSI X3) define those bindings? There is some discussion of Modula 2's and COBOL's doing this, and the Fortran 90 project might be resurrected in X3J3 rather than IEEE POSIX. Is this good or bad?

4. Is the lack of interest in Fortran 90 simply a timing issue due to the lack of widespread access to Fortran 90 compilers, or is it due to a lack of interest in Fortran 90 itself? or is this just another victim of the economy?

There are undoubtedly a wide variety of reasons why there is insufficient support at this time for this work. There could be considerable debate over which reason was the most significant. Some would argue that the group should never have tried to start. However, it is clear that there is inadequate support to continue. I believe it is the responsible thing to disband this working group. If you don't agree, now is the time to speak up.

Unix Magic

Recently I was helping someone who was writing a C program that would allow non-privileged users to perform administrative functions such as controlling printer queues.

Things were going smoothly: the program simply `exec()`d the appropriate program after setting its effective UID.

However, a problem had arisen - for some reason, it wasn't possible to invoke some programs (such as `/etc/reboot`) - even as root. The message coming back was "exec format error".

I had a quick look at those programs - they were both (executable) shell scripts. I edited them by simply adding the familiar line:

```
#!/bin/sh
```

to the top of each file. Suddenly, the `exec()` of each of these files was working! My answer to why this fixed the problem was simple - "It's Magic".

This answer wasn't (quite) as flippant as it seems. UNIX doesn't support different file types at the operating system level (as other operating systems do).[†] Instead, each file is simply a sequence of bytes. It is up to the applications that use the file to impose a structure upon it.

However, if different applications will share the file, they need to agree on conventions on that file's format. For example, the operating system needs to be able to differentiate between executable (binary) files and shell scripts (and between shell scripts and ordinary ASCII files).

A file's type can be determined by looking for particular *magic numbers* within the file. If the file starts with a particular magic number, it is assumed to be a shell script. Similarly, there are other magic numbers associated with binary executables for particular CPUs.

In fact, the magic number associated with shell scripts is the number you get from reading the `#!` characters at the beginning of the file as a number. Shell scripts are an interesting case - if there are any characters following the `#!`, they are taken as the name of an executable program to start up. That program then interprets the rest of the script. (To prevent loops, if the interpreter itself starts with `#!`, the `#!` is ignored).

When we were trying to `exec()` a file before, the file didn't start with a magic number associated with either a shell script or a binary executable - `exec()` didn't know how to run it. Simply starting the file with `#!` gave `exec()` the information it needed.

Many magic numbers that are used in a similar fashion are stored in the `/etc/magic` file. This file has a simple format: we specify a byte offset, a type (such as `byte`, `short`, or `string`), a value to match, and a descriptive string.

This explains how `file(1)` works - it scans through `/etc/magic`, checking the file against each rule. If the file contains a `<type>` of the specified value at the specified byte offset, the descriptive string is printed. (`file(1)` also has some built-in heuristics).

We can also define our own rules in `/etc/magic`. As an example, suppose we wanted `file(1)` to recognise PostScript documents (which we will assume always start with `%!`).

We want a line in `/etc/magic` that looks for a string, `%!`, at byte offset 0 into the file, and prints the words "PostScript file" if a match is found.

To do this, we would put the following into `/etc/magic`:

```
0 string %! PostScript file
```

We can also allow more sophisticated file identification. As a contrived example, let us assume that we have an application (call it `f○○`) that outputs files that are later translated and sent to the printer.

[†] A slight simplification - let's ignore directories and special files.

These files always start with %%PRINT, followed by either L for landscape or P for portrait printing.

We could simply say:

```
0 string %%PRINTL foo output file (Landscape format)
0 string %%PRINTP foo output file (Portrait format)
```

Instead, we can first identify the file as a foo output file, and then determine its orientation:

```
0 string %%PRINT foo output file
>7 string L , landscape format
>7 string P , portrait format
```

The lines that start with > are continuation lines - if the first line matches, the continuation lines are scanned as well. This explains how a file can have several attributes (such as being a [dynamically linked] [stripped] [executable [for a particular CPU]]) - it has several magic numbers at different byte offsets into the file.

/etc/magic's format is really quite simple. Take a look at it sometime - you will be surprised at how many obscure file formats your UNIX knows about.

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

Unix Tricks — forcing NFS cache flushes

The Network File System (NFS) is probably the most widely used standard for remote file system access. If your workstation is part of a network, then there's a good chance you are sharing files using NFS between workstations and servers.

NFS performance is improved by NFS clients caching recent reads and writes including data blocks and inodes (information about files). Sometimes the caching can get in the way, but there are ways to force NFS to flush its cache and reload up-to-date data from the server.

One trivial example that I come across often is when I try to access a file on an NFS client and fail due to the wrong file permission:

```
client> cat /usr/local/include/tcl.h
cat: /usr/local/include/tcl.h: Permission denied

client> ls -l /usr/local/include
total 57
-rw----- 1 root      2577 Aug 13  1991 expect.h
-rw----- 1 root     12493 Jun  2  18:50 tcl.h
-rw----- 1 root      4968 Jun  2  18:50 tclHash.h
-rw----- 1 root     28556 Jun  2  18:49 tk.h
```

I can go to the NFS server (where these files really reside) and change the file modes:

```
server> chmod 644 /usr/local/include/*.h
```

If the NFS client is a busy machine, then the old inode permissions in its NFS cache might get flushed by other NFS activity. However, on a quiet or personal workstation, the client may still have the old data:

```
client> ls -l /usr/local/include
total 57
-rw----- 1 root      2577 Aug 13  1991 expect.h
-rw----- 1 root     12493 Jun  2  18:50 tcl.h
-rw----- 1 root      4968 Jun  2  18:50 tclHash.h
-rw----- 1 root     28556 Jun  2  18:49 tk.h
```

I can wait until NFS decides that its cache is too old and invalidates the data, or I can force it to flush the cache by replacing it with other data, by listing any largish directory:

```
client> ls -l /usr/bin > /dev/null

client> ls -l /usr/local/include
total 57
-rw-r--r-- 1 root      2577 Aug 13  1991 expect.h
-rw-r--r-- 1 root     12493 Jun  2  18:50 tcl.h
-rw-r--r-- 1 root      4968 Jun  2  18:50 tclHash.h
-rw-r--r-- 1 root     28556 Jun  2  18:49 tk.h
```

and now, I can read the file. This works because on my system /usr/bin is also NFS mounted and is large enough that all the reads required to perform the `ls` force the NFS client to eventually discard all the old data in its cache.

Glenn Huxtable <glenn@cs.uwa.edu.au>
From WAUG, the WA Chapter of AUUG

Unix Traps — endlessly recursive directories

Here's a little test I just did.

```
% ls -l
total 1
drwx--s--x  3 janet          512 Jul 25 16:34 test
% ls -l test
total 1
drwx--s--x  2 janet          512 Jul 25 16:34 junk
% cp -r test test/junk
cp: test/junk/test/junk/test/junk/test/junk/test/junk/test/junk/test/
junk/test/junk/test/junk/test/junk/test/junk/test/junk/test/junk/test/
junk/test/junk/test/junk/test/junk/test/junk/test/junk/test/junk/tes
t/junk/test/junk/test/junk/test/junk/test/junk/test/junk/test/junk/te
st/junk/test/junk/test/junk/test/junk: Too many open files
```

Copy a directory on to one of its own subdirectories, and `cp(1)` will dutifully recurse until something snaps (in this case, the file descriptor table). `ls -R` will show that all those directories have indeed been created.

Now, if you accidentally did this, you'd probably realise what had happened. However, the other day, I helped a user who had managed to do it from Sun's File Manager, which is one of those horrid Macintosh-style interfaces to the Unix filesystem. "I just did this and that and then pressed 'Cut'," he protested. "I have logged out and back in again, but still it is copying and copying." Well, it wasn't. It had stopped, as above. However, one of his shell startup files used `find` to print the locations of core files, and — you guessed it — there was a core file in the directory he'd copied, so when he logged in, `find` was reporting all n copies of it, for some ridiculous value of n . It took me a good fifteen minutes to untangle first my, then his confusion.

There are two morals to this story. The first is that graphical filesystem interfaces can be a pain.

The second is related to the `find` in the user's shell startup file. This had been put there by some well-meaning systems administrator. Unfortunately, the user had no idea what the list of core file pathnames meant, and if there were no core files, it simply took a long time to log in (something he had never questioned). So the second moral is for systems administrators: don't be tempted to inflict all your own neat shell startup commands on your users.

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

AUUG MANAGEMENT COMMITTEE

SUMMARY OF MINUTES OF MEETING 2nd July 1993

Present: Phil McCrea, Glenn Huxtable, Frank Crawford, Peter Wishart, Michael Paddon, Greg Birnie.
Guests: Liz Fraumann, Wael Foda, Lachie Hill, Simon Kravis, Paul Spencer.
Apologies: Chris Maltby.

1. Election Results

PM welcomed Greg Birnie as a newly elected member of the committee.

1.1. Vacant Positions

The last election results had left two committee positions and the Returning Officer and Asst. Returning Officer positions vacant. The committee discussed several candidates for these positions.

It was decided that Chris Schoettle and Stephen Boucher should be co-opted into the two vacant position on the AUUG Management Committee.

Michael Tuke was co-opted into the vacant position of Returning Officer. It was agreed to seek advice from Michael Tuke over who should be co-opted to the position of Asst. Returning Officer.

2. AUUG94 Concept

The theme for AUUG94 need to be decided before AUUG93 so advertising material and preparations can begin at AUUG93. There were a number of presentations on the theme and concept for AUUG94:

- (1) a concept based around keys. "The Key to Open Systems".
- (2) a concept based around a southern cross motif. "Building the Enterprise".
- (3) a concept based around a crystal ball. "Open Systems: The Future is Clear".

It was agreed that the crystal ball concept was the best and should be selected. There was some discussion about the appropriateness of the words "The Future is Clear". There may need to be some work to come up with a different wording.

3. AUUG93 Update

3.1. Budget

FC reported that there will be a significant expenditure this month for AUUG93 expenses. This was the worst month for cash flow. The finances were tight this month but FC is confident that we can meet the demand.

EF reported that in general we are slightly under budget for AUUG93. Most items were coming in on budget or just below.

3.2. Student Volunteers

Students are being organised to provide a locator service to help people find things at the conference. Students would be rostered on and be identified in "STAFF" T-Shirts.

3.3. Show Daily

There would be a "show daily" for the conference. It was called "AND ..." (for AUUG News Daily). It would be around 8 pages each day and distributed at the start of each day. We are hoping to get copies delivered to the doors of delegates staying in the conference hotels early each morning. Of the 8 pages, 5 pages would be prepared before the conference and 3 pages would be topical news about the conference done overnight. We would be printing around 10,000 copies.

3.4. AUUG93 Launch

PM and EF attended the AUUG93 launch. It received good press coverage (Australian, UNIX Update front cover ...). It will probably make it into monthlies and weeklies a little later.

3.5. Mailing Lists

We should have a 30,000 mailout. The ACS mailout will be sent out separately to the others under a cover letter from the ACS.

3.6. Registrations and Program

We have received 54 early bird registrations. This makes the exercise of early bird registrations worthwhile.

It is planned to have a small 2/3 A4 bi-fold conference program for delegates. This will be easier to carry around for reference during sessions and differentiate it from the exhibition guide.

There have been 7 BOFs scheduled so far: Plan 9, SAGE, SCO UNIX, Codex, NT & UNIX (by CMP and Sun), and Sun User Group. There are no hospitality suites registered yet.

3.7. Student Grants

ACS has offered a prize for the best student paper \$500. To be presented by Brendan Hanley from ACS. The Research Foundation for IT has offered to pay for 5 students registration at AUUG93. There was discussion about making this available to students around Australia and the feasibility of that. It was decided to post an offer to aus.auug and select 5 students.

4. Weekly Column in The Australian

PM reported that the Australian has asked AUUG to provide a weekly column. They require a 700 word article each week. There can be no corporate messages or product pitches. They would like to see business issues, and technical issues. They suggested the first column be about cost issues, covering running and setup. The column will have an AUUG byline and logo. This is a very important piece of profile for AUUG. MP will coordinate articles to ensure consistency.

5. Membership

WF presented the membership report from the Secretariat. We have about 10% new membership. The "un-financial members" numbers shown are inflated since the memberships renewals only went out a few weeks ago and the membership term had only expired 2 days ago. It was noted that membership growth seemed strongest in areas with active chapters.

6. President Report

PM reported that he had asked Dame Leonie Kramer to be the dignitary to open AUUG93. He has not yet had an official response.

7. Treasurers Report

FC reported that we made a \$26,300 loss last financial year. We had projected a \$25,000 profit. The difference matches fairly closely to the \$57,000 drop in income from AUUG92. Our plans for 1993/94 and AUUG93, with the revised profit sharing arrangements, should address this problem.

8. Secretary's Report

Letters have been sent to all new chapters welcoming them to AUUG. A formal petition has now been received from Darwin (AUUGNt), formalising their chapter status.

9. Profit Sharing for Summer Conferences

There was discussion about the disparity of current arrangements for profit sharing with summer conferences. It was noted that some chapters return all funds, some keep their funds. In order to provide uniformity and to fund AUUG national support for the conferences it was agreed that some policy was needed. This policy would apply to future conferences. This should be discussed at the chapter council meeting.

Motion: That the following arrangement be put to the chapter council for discussion: the chapters receive a 75% share of profits from the summer conference and AUUG to receive 25%. This arrangement excludes the one summer conference each year which receives additional support, that conference will have separate arrangements. Moved: GH/MP. CARRIED.

10. AUUG93 Committee Registration/Travel Expenses

There was discussion about the policy for paying committee members fees/travel to attend conferences. It was agreed to re-affirm the existing policy.

Motion: That committee members are re-imbursed for travel to committee meetings and a committee member's travel to the winter conference to attend meetings is covered, but only if the member is not attending the

11. AUUGN

The back cover of AUUGN has been sold to a client through JSP. We need more contacts for advertising in AUUGN. We need to make sure all chapters provide newsletter input for AUUGN. We should also be archiving the chapter newsletters.

12. Chapters

Chapters need to get timely updates of membership data from the Secretariat. At least every six months, preferably monthly. It was time to review the membership database. PW to liaise with Secretariat to work out how to keep chapters informed.

It was agreed that the chapter council meeting would be held on the Saturday after the next committee meeting in August. Committee meeting - Fri Aug 27th @ Softway. Chapter Council Sat Aug 28th, venue TBA.

There is now a petition to form a NSW chapter. There is an interim committee which will meet soon.

13. Summer Conferences

Kirk McKusick was keen to present a paper and tutorial at the conferences. AUUG would pay air-fares to Aust. and each conference and also accommodation at each conference. Kirk wanted to make money from the tutorials to support himself. This issue should be discussed at the chapter council. GH will continue to liaison with Kirk about arrangements.

14. Other Business

It was agreed that we should seek relationships with other user groups in the Asia/Pacific region.

15. Next Meeting

The next meeting will be on Fri 27th August 1993 at Softway. The chapter council meeting will be the next day (Sat 28th August) at a venue to be advised.

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.

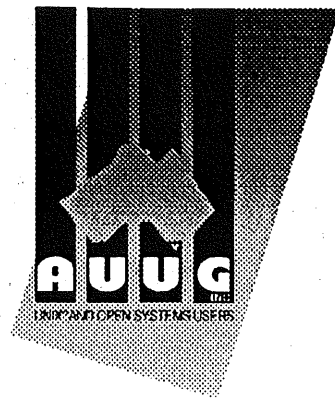
Check that your membership isn't about to expire and always keep your address up-to-date. Ask your colleagues if they received this issue of AUUGN, tell them that if not, it probably means that their membership has lapsed, or perhaps, they were never a member at all! Feel free to copy the membership 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

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

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

TOTAL REMITTED: AUD\$ _____ (Cheque, or money order, or credit card)

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: _____

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

Over for Institutional Membership

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

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.

AUUG Incorporated

Application for Newsletter Subscription

AUUG Inc.

Non members who wish to apply for a subscription to the Australian UNIX systems User Group Newsletter, or members who desire additional subscriptions, should complete this form and return it to:

AUUG Membership Secretary
 PO Box 366
 Kensington NSW 2033
 Australia

- Please don't send purchase orders — perhaps your purchasing department will consider this form to be an invoice.
- Foreign applicants please send a bank draft drawn on an Australian bank, or credit card authorisation, and remember to select either surface or air mail.
- Use multiple copies of this form if copies of AUUGN are to be dispatched to differing addresses.

This form is valid only until 31st May, 1994

Please *enter / renew* my subscription for the Australian UNIX systems User Group Newsletter, as follows:

Name: Phone: (bh)
 Address: (ah)

 Net Address:

 Write "Unchanged" if address has
 not altered and this is a renewal.

For each copy requested, I enclose:

- Subscription to AUUGN \$ 90.00
- International Surface Mail \$ 20.00
- International Air Mail \$ 60.00

Copies requested (to above address) _____

Total remitted AUD\$ _____
 (cheque, money order, credit card)

Tick this box if you wish your name & address withheld from mailing lists made available to vendors.

Please charge \$ _____ to my Bankcard Visa Mastercard.

Account number: _____ . Expiry date: ____ / ____ .

Name on card: _____ Signed: _____

Office use only:

Chq: bank _____ bsb _____ - a/c _____ # _____

Date: ____ / ____ / ____ \$ _____ CC type ____ V# _____

Who: _____ Subscr# _____

AUUG

Notification of Change of Address AUUG Inc.

If you have changed your mailing address, please complete this form, and return it to:

AUUG Membership Secretary Fax: (02) 332 4066
PO Box 366
Kensington NSW 2033
Australia

Please allow at least 4 weeks for the change of address to take effect.

Old address (or attach a mailing label)

Name: Phone: (bh)

Address: (ah)

..... Net Address:

.....

.....

.....

New address (leave unaltered details blank)

Name: Phone: (bh)

Address: (ah)

..... Net Address:

.....

.....

.....

Office use only:

Date: ___/___/___

Who: _____

Memb# _____