

Australian UNIX systems User Group Newsletter

AUUGN

Volume 13, Number 3

June 1992

AUUGN

Volume 13 Number 3

June 1992

The AUUG Incorporated Newsletter

Volume 13 Number 3

June 1992

CONTENTS

AUUG General Information	3
Editorial	5
AUUG Institutional Members	7
AUUG Management Committee Election Results	9
AUUGN New Cover Design Competition	10
AUUG'92 Update	11
Open System Publications	13
SESSPOOLE	14
The WAUG Column	15
ACSnet Survey	16
MHS Release: AUUG Members Network Access	19
AUUG Book Club	20
AUUG Book Club- Order Form	23
Adelaide Summer'92 Technical Conference report	<i>Michael Wagner</i> 24
Melbourne Summer'92 Technical Conference report	<i>Michael Paddon</i> 26
Using a Multi-User UNIX Computer System as a File and Print Server for	30
MS-DOS based Personal Computer Local Area Networks	<i>Ross Parish</i> 30
Softway Engineering or Structured UNIX Kernel Hacking	<i>Peter Chubb</i> 45
Not Just Another Add-User Script:	52
Automating User Administration with Perl	<i>Janet Jackson</i> 52
Disaster Mitigation in a UNIX Environment	<i>Graham Jenkins</i> 62
From ;login - Volume 17, Number 2	66
An Update on UNIX-Related Standards Activities	66
Book Review: The Design and Implementation of the 4.3 BSD	75
UNIX Operating System Answer Book	75
Management Committee Minutes - 18th May 1992	77
AUUG Membership Categories	84
AUUG Forms	85

Copyright © 1992 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@muninari.oz.au

AUUG Business Manager

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

Phone: (02) 953 3542
Fax: (02) 953 3542
Email: eaf@softway.sw.oz.au

AUUG Executive

President **Pat Duffy**
pzd30@juts.ccc.amdahl.com
Amdahl Pacific Services Pty. Ltd.
1 Pacific Highway
North Sydney NSW 2000

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

Secretary **Rolf Jester**
rolf.jester@sno.mts.dec.com
Digital Equipment Corporation
(Australia) Pty. Ltd.
P.O. Box 384
Concord West NSW 2138

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

Committee Members **Andrew Gollan**
adjg@softway.sw.oz.au
Softway Pty. Ltd.
79 Myrtle Street
Chippendale NSW 2008

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

Peter Karr
Computer Magazine Publications
1/421 Cleveland Street
Redfern NSW 2016

Michael Tuke
mjt@anl.oz.au
ANL Ltd.
432 St. Kilda Road
Melbourne VIC 3004

Scott Merrilees
Sm@bhpesse.oz.au
BHP Information Technology
P.O. Box 216
Hamilton NSW 2303

AUUG General Information

Next AUUG Meeting

The AUUG'92 Conference and Exhibition will be held from the 8th to the 11th of September, 1992, at the World Congress Centre, Melbourne. See later in this issue for an update.

AUUG Newsletter

Editorial

Welcome to AUUGN Volume 13 Number 3.

In this issue we have the results of the AUUG Management Committee Election, and it has changed considerably. Note that the General Information on page three will change for later issues, as the new committee doesn't take effect until 1st July.

Also included in this issue are two more reports from the summer conferences and four papers from these conferences. An update on AUUG'92 is included and some book reviews organised by the Book Review Editor.

Despite the large number of papers available from the conference I am still interested in others, so don't hesitate to contact me.

Jagoda Crawford

AUUGN Correspondence

All correspondence regarding the AUUGN should be addressed to:-

AUUGN Editor
PO Box 366
Kensington, NSW, 2033
AUSTRALIA

E-mail: auugn@munnari.oz.au

Phone: +61 2 717 3885
Fax: +61 2 717 9273

AUUGN Book Reviews

The AUUGN Book Review Editor is Dave Newton. He has just changed jobs, so please contact me for more details.

A number of books are currently being reviewed. These reviews will be published in future issues.

Contributions

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

Volume 13 No 4	Friday 31st July
Volume 13 No 5	Friday 25th October
Volume 13 No 6	Friday 27th 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 left at the top and bottom so that the AUUGN footers can be pasted on to the page. Small page numbers printed in the footer area would help.

Advertising

Advertisements for the AUUG are welcome. They must be submitted on an A4 page. No partial page advertisements will be accepted. Advertising rates are \$300 for the first A4 page, \$250 for a second page, 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 editor 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

Acknowledgement

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

Disclaimer

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

AUUG Institutional Members as at 18/06/1992

A.J. Mills & Sons Pty Ltd	Data General Australia
A.N.U.	Deakin University
AAII	Defence Housing Authority
Adept Business Systems Pty Ltd	Defence Service Homes
Adept Software	Dept of Industrial Relations, Employment, Training & Further Education
Alcatel Australia	Dept. of Agricultural & Rural Affairs
Amdahl Pacific Services	Dept. of Conservation & Environment
Andersen Consulting	Dept. of Defence
ANI Manufacturing Group	Dept. of Foreign Affairs & Trade
ANSTO	Dept. of I.T.R.
ANZ Banking Group/ Global Technical Services	Dept. of Minerals & Energy (NSW)
Apple Computer Australia	Dept. of the Premier and Cabinet - SA
ApSCORE International Pty Ltd	Dept. of the Premier and Cabinet - VIC
Ausonics Pty Ltd	Dept. of the Treasury
Australian Airlines Limited	Dept. of Transport
Australian Bureau of Agricultural and Resource Economics	Dept. of Treasury & Finance
Australian Bureau of Statistics	DEVETIR
Australian Defence Industries Ltd	Digital Equipment Corp (Australia) Pty Ltd
Australian Eagle Insurance Co. Ltd	DMP Software Pty Ltd
Australian Electoral Commission	Duesburys Information Technology Pty Ltd
Australian Information Processing Centre Pty Ltd	EDS (Australia) Pty Ltd
Australian Museum	Electronics Research Labs
Australian National Parks & Wildlife Service	Emulex Australia Pty Ltd
Australian Taxation Office	ESRI Australia Pty Ltd
Australian Technology Resources (A.C.T.)	Expert Solutions Australia
Australian Wool Corporation	FGH Decision Support Systems Pty Ltd
Avid Systems Pty Ltd	Financial Network Services
Bain & Company	First State Computing
Ballarat Base Hospital	Fremantle Port Authority
BHP CPD Research & Technology Centre	Fujitsu Australia Ltd
BHP Information Technology	G. James Australia Pty Ltd
BHP Minerals	GCS Pty Ltd
BHP Research - Melbourne Laboratories	GEC Alsthom Australia
BICC Communications	Geelong and District Water Board
Bond University	Gemco
"Burdett, Buckeridge & Young Ltd."	Genasys II Pty Ltd
Bureau of Meteorology	General Automation Pty Ltd
Byrne & Davidson Holdings Pty Ltd	George Moss Ltd
C.I.S.R.A.	GeoVision Australia
Capricorn Coal Management Pty Ltd	GIO Australia
CITEC	Golden Circle Australia
Co-Cam Computer Group	Grand United Friendly Society
Codex Software Development Pty. Ltd.	Hamersley Iron
Cognos Pty Ltd	Harris & Sutherland Pty Ltd
Colonial Mutual	Hermes Precisa Australia Pty. Ltd.
Com Tech Communications	Highland Logic Pty Ltd
Commercial Dynamics	Honeywell Ltd
Communica Software Consultants	Honeywell Ltd
Computech Pty Ltd	I.B.A.
Computer Power Group	IBM Australia Ltd
Computer Sciences of Australia Pty Ltd	Iconix Pty Ltd
Computer Software Packages	Information Technology Consultants
Corinthian Engineering Pty Ltd	Insession Pty Ltd
CSIRO	Insurance & Superannuation Commission
Curtin University of Technology	Internode Systems Pty Ltd
Cyberscience Corporation Pty Ltd	Ipec Management Services
	IPS Radio & Space Services
	James Cook University of North Queensland

AUUG Institutional Members as at 18/06/1992

Labtam Australia Pty Ltd
Lancorp Pty. Ltd.
Land Information Centre
Leeds & Northrup Australia Pty. Limited
Macquarie University
Mayne Nickless Courier Systems
McDonnell Douglas Information Systems Pty Ltd
McIntosh Hamson Hoare Govett Ltd
Medical Benefits Funds of Australia Ltd.
Mentor Technologies Pty Ltd
Metal Trades Industry Association
Mincom Pty Ltd
Minenco Pty Ltd
Ministry of Consumer Affairs
Ministry of Housing & Construction (VIC)
Mitsui Computer Limited
Motorola Computer Systems
Multibase Pty Ltd
NEC Information Systems Australia Pty Ltd
NSW Agriculture
Nucleus Business Systems
Office of the Director of Public Prosecutions
Olivetti Australia Pty Ltd
OPSM
Oracle Systems Australia Pty Ltd
Ozware Developments Pty Ltd
Parliament House
Paxus
Philips PTS
Port of Melbourne Authority
Powerhouse Museum
Prentice Hall Australia
Prime Computer
Prospect Electricity
Public Works Department
Pulse Club Computers Pty Ltd
Pyramid Technology
Q.H. Tours Limited
Queensland Department of Mines
Queensland University of Technology
Redland Shire Council
RMIT
Royal Melbourne Institute of Technology
SBC Dominguez Barry
Sculptor 4GL+SQL
SEQEB Control Centre
Shire of Eltham
Silicon Graphics Computer Systems
Snowy Mountains Authority
Software Development International Pty Ltd
Softway Pty Ltd
Sony Australia Pty Ltd
South Australian Lands Dept.
Sphere Systems Pty Ltd
St Vincent's Private Hospital
Stallion Technologies Pty Ltd
Stamp Duties Office
Standards Australia
Steedman Science and Engineering
Sugar Research Institute
Swinburne Institute of Technology
Sydney Ports Authority
Systems Union Pty Ltd
Tasmania Bank
Tattersall Sweep Consultation
Telecom Australia
Telecom Australia Corporate Customer
Telecom Network Engineering Computer
Support Services
Telecom Payphone Services
Telectronics Pty Ltd
The Anti-Cancer Council of Victoria
The Far North Qld Electricity Board
The Fulcrum Consulting Group
The Opus Group
The Preston Group
The Roads and Traffic Authority
The Southport School
The University of Western Australia
TNT Australia Information Technology
Toshiba International Corporation Pty Ltd
Tower Computing Services
Tower Technology Pty Ltd
Tradelink Plumbing Supplies Centres
Triad Software Pty Ltd
TurboSoft Pty Ltd
TUSC Computer Systems
UCCQ
Unidata Australia
Unisys
University of Adelaide
University of Melbourne
University of New South Wales
University of Queensland
University of South Australia
University of Sydney
University of Tasmania
University of Technology
UNIX System Laboratories
Unixpac Pty Ltd
Vibro Acoustic Sciences Ltd.
Vicomp
VME Systems Pty Ltd
Wacher Pty Ltd
Walter & Eliza Hall Institute
Wang Australia Pty. Ltd.
Water Board
Westfield Limited
Wyse Technology Pty. Ltd.

AUUG Management Committee Election Results

Included below are the results of the AUUG Management Committee Election Results. A full Returning Officer's report was not available at the time of publication.

Total Votes Received:	109
President:	Philip McCrae
Vice President:	Glenn Huxtable
Secretary:	Peter Wishart
Treasurer:	Frank Crawford
General Committee (In alphabetical order):	
	Rolf Jester
	Chris Maltby
	John O'Brien
	Michael Paddon
	Greg Rose
Returning Officer:	Michael Tuke
Assistant Returning Officer:	Vacant

AUUGN New Cover Design Competition

We are looking for a new cover design for AUUGN to start with the first issue in 1993. To obtain member input we have decided to run a competition open to all AUUG members, with the entries to be judged by the AUUGN editor.

What is required to appear on the cover is:

1. The Name of the publication: AUUGN
2. The Name of the organisation: AUUG Inc.
3. The Volume and Issue numbers
4. Date, *i.e.* the Month and Year of the publication.
5. The registration number (ISSN 1035-7521).
6. Australia Post registration, currently, Registered by Australia Post, Publication Number NBG6524

The design should be easily reproducible and modifiable, (*i.e.* changes in date, *etc.*, also keep in mind the information on the spine). The current cover is black on coloured background (the colour changes monthly: white, red, green, yellow, orange and blue). Such a colour scheme is preferred, however there is limited opportunity for use of basic colours, keeping in mind that ease of reproduction is required and cost has to be kept down.

As with papers, I prefer documents to be e-mailed to me in postscript or troff. Hardcopy submissions should be on A4, addressed to:

Jagoda Crawford
AUUGN Editor
PO Box 366
Kensington, NSW, 2033
AUSTRALIA

E-mail: jc@atom.ansto.gov.au

Phone: +61 2 717 3885
Fax: +61 2 717 9273

The competition closes on Friday September 11th 1992. The entries will be displayed at the AUUG'92 for comments. The result will be announced shortly afterwards with the designer of the selected entry receiving a free year's subscription to AUUG.

AUUG '92 Update

It is time to mark in your diaries the week of 8 - 11 September for AUUG '92. This year, Australia's premiere conference and exhibition will be held in Melbourne at the World Congress Centre.

8 September is a full day dedicated to tutorials:

We are offering 2 full day sessions:

- SVR4 Internals
- BSD Internals

AM 1/2 day sessions:

- Insights into Guru-level C Programming
- Cruising the Network
- Intro to the UNIX Operating System

PM 1/2 day sessions:

- PERL
- The Easy Route to X-Windows
- Business Requirements and Open Systems

9 - 11 September offers 3 morning Keynote and Plenary sessions and 3 streams, Technical, Management, and Combined for the afternoon.

- Technical sessions are designed for the computer scientist or programmer wanting in depth knowledge. The how and why of programming details will be presented.
- Management sessions are designed to present the business case perspective on topics of general concern to managers in the computer industry.
- Combined stream, new to AUUG, is designed to provide enough technical and management information to assist MIS directors and systems administrators in making educated decisions.

Over 53 local and international speakers will provide their global and expert insights with respect to such topics as Rightsizing, Free Software, Fault Tolerance and the Future, Virtual Reality, Hardware Profiling of a UNIX Kernel, Log Structured File Systems, the New Security Paradigm, CICS for Open Systems, Open Systems, The Search for the Holy Grail, Xcelsis, A new Approach to Spreadsheets, Porting C++ to an Embedded Environment.... and many more!

The Conference Program/Registration brochures will be distributed in the first week of July. AUUG Members should look for their copy by the 15th of July and if you have not received it by then, please contact AUUG Secretariat at: 02 361-5994.

Fees for AUUG '92 are as follows:

<i>Tutorials:</i>	<i>On or before 28 Aug</i>	<i>After 28 Aug</i>
Full Day	\$280.00	\$330.00
Half Day	\$160.00	\$210.00
<i>Conference</i>	<i>On or before 28 Aug</i>	<i>After 28 Aug</i>
AUUG Members	\$300.00	\$350.00
Non Members	\$500.00	\$550.00
Day Fee Members	\$150.00	\$200.00
Day Fee Non Members	\$200.00	\$250.00
Full-time Student	\$100.00	\$150.00

Please note in an effort to reduce costs, luncheon is NOT included. If you desire to partake in a two course light luncheon, it is an additional \$25.00 per day. The cocktail party on Wednesday, 9 September and the Gala Dinner on Thursday, 10 September is included in the conference fee.

You should also note in the Conference Program/Registration brochure, that in the General Information section under travel, special arrangements with Ansett via Performax Travel have been negotiated to secure discount rates. In addition to the special accommodation rates other local hotels are sited for your convenience. A map on page 16 of the brochure will assist in locating the best site for you.

We look forward to seeing you in Melbourne for a very successful conference and exhibition!

- The Program Committee:
- Peter Karr - Chair
 - Liz Fraumann
 - Ian Hoyle
 - Robert Elz

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,
Uniform,
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
(02) 332 4066

Fax:

SESSPOOLE

SESSPOOLE is the South Eastern Suburbs Society for Programmers Or Other Local Enthusiasts. That's the South Eastern Suburbs of Melbourne, by the way.

SESSPOOLE is a group of programmers and friends who meet every six weeks or so for the purpose of discussing UNIX and open systems, drinking wines and ales (or fruit juices if alcohol is not their thing), and generally relaxing and socialising over dinner.

Anyone who subscribes to the aims of SESSPOOLE is welcome to attend SESSPOOLE meetings, even if they don't live or work in South Eastern Suburbs. The aims of SESSPOOLE are:

To promote knowledge and understanding of Open System; and to promote knowledge and understanding of Open Bottles.

SESSPOOLE is also the first Chapter of the AUUG to be formed, and its members were involved in the staging of the AUUG Summer '90, '91 and '92 Melbourne Meetings.

SESSPOOLE meetings are held in the Bistro of the Oakleigh Hotel, 1555 Dandenong Road, Oakleigh, starting at 6:30pm. Dates for the next few meetings are:

Wednesday, 8 July 1992
Thursday, 20 August 1992
Tuesday, 29 September 1992
Wednesday, 11 November 1992
Thursday, 17 December 1992
Tuesday, 2 February 1993
Wednesday, 17 March 1993
Thursday, 29 April 1993
Tuesday, 8 June 1993
Wednesday, 21 July 1993

Hope we'll see you there!

To find out more about SESSPOOLE and SESSPOOLE activities, contact either **Stephen Prince** (ph. (03) 608-0911, e-mail: sp@clcs.com.au) or **John Carey** (ph. (03) 587-1444, e-mail: john@labtam.oz.au), or look for announcements in the newsgroup aus.auug.

The WAUG Column

The speaker at our May meeting was Tim Hoffman from Sun Microsystems' Perth office. He spoke on "Multiprocessor Computing" - more specifically, on the multi-threaded architecture Sun are developing for their multiprocessor systems.

He started by briefly describing Solaris 2.0, which I previously thought was an operating system; it turns out to be a name for an operating system, called SunOS 5.0, combined with the OpenWindows graphical user interface and various applications that run on top of that. (Sun's current release is Solaris 1.0.1, which includes SunOS 4.1.2.)

Having sorted that out, Tim went on to the main purpose of his talk, a technical description of SunOS 5.0's multiprocessing architecture. The basic idea is that threads - flows of control within a process - are mapped onto lightweight processes, which provide "virtual CPUs". One or more threads may be mapped onto the one lightweight process. Lightweight processes are in turn mapped onto kernel threads, which run on the real CPUs. The programmer controls the number and combination of threads and lightweight processes controlled by a real process. The threads within a process share memory and other resources, such as open files. Synchronising access to these is up to the programmer; various mechanisms are provided.

Tim also described the hardware architecture of Sun's multiprocessor systems. This made some people start asking technical hardware questions, to which Tim responded that he was a software person who would rather be home hacking in Postscript. (Some people's idea of fun is very strange!)

Conditions were difficult at the beginning of Tim's talk because of loud music coming from the next room. When we complained, we were told that a private function were being entertained by a male stripper, who would soon be finished. The music actually continued for quite a while, during which Tim showed great restraint by removing only his jacket.

We had an excellent turnout at the meeting. Judging by the distribution of sexes, the strip-show next door had little to do with it.

Administrative note: WAUG now has a new meeting organiser, Mark Baker, and he is looking for speakers. So if you'd like to speak or have an idea for a speaker, please contact him at baker@telecomwa.oz.au or on (09) 420 6813.

If you're interested in joining WAUG (the Western Australian systems Group) or contributing to our newsletter YAUN (Yet Another Newsletter), our address is PO Box 877, WEST PERTH WA 6005.

Janet Jackson <janet@cs.uwa.edu.au>

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:

Frank Crawford (frank@atom.lhrl.oz) (02) 717 9404
or
Scott Merrilees (Sm@bhpes.oz) (049) 40 2132

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:

MHS RELEASE: AUUG MEMBERS NETWORK ACCESS

Under a special arrangement with Message Handling Systems - operators of TMX (The Message eXchange) - AUUG is providing members with a packaged offering for electronic mail and news access to the major networks.

The Message eXchange is a well established service that has been operating for over twelve months with a growing user community that includes individuals, large computer vendors and small software organisations.

AUUG members can dial-in to the TMX facilities to exchange messages with:

- other AUUG members
- the main Australian and overseas networks
- all or selected Australian and Usenet news groups
- the Clarinet professional news and information service.

The special price for registered AUUG members is a one-time \$395 (the normal TMX fee is \$495) which is inclusive of an initial 10 free connect hours and a special version of the MHSnet software. After the first ten hours, connection rates are \$8.00 per hour. (For volume users there is an optional prepayment of \$1200 for 300 hours at \$4.00 per connect hour). There is a minimum fee of one hour per month.

Connection to TMX may be via V22, V22bis, V32 and PEP modems. MS.DOS and Apple MacIntosh connections are also available.

The Clarinet service, for which fees are applicable, includes UPI wireservice, computer industry news, financial information and up-to-date major world news stories.

The TMX service is currently available in Sydney and the Melbourne service is expected to be operational by July. While other centres would need to dial-in to either of these systems, it is intended to expand the service to other states based on traffic demand.

AUUG is pleased to be able to introduce this service to its members as a means to increase communication effectiveness. For more information please contact Elaine Pensabene at Message Handling Systems Pty Ltd.

Phone: (02) 550 4448
Fax: (02) 519 2551
Email: elaine@mhs.oz.au

AUUG Book Club

Book Reviews

AUUG Inc and Prentice Hall Australia have formed the AUUG Book Club to give AUUG members a chance to obtain Prentice Hall books at a significant discount.

To obtain copies of the books reviewed here, fill in the order form that appears at the end of the book reviews. Don't forget to deduct 20% from the listed retail prices.

Review copies of these books were kindly provided by Prentice Hall.

If you would like to review books for further offers from the AUUG Book Club, please contact the AUUGN Book Review Editor (see page 5).

C STYLE: STANDARDS AND GUIDELINES

by David STRAKER
Prentice Hall, RRP \$42.95
ISBN: 0-13-116898-3, 1992

*Reviewed by
Peter Chubb
Softway Pty Ltd*

Who needs standards?

Wherever more than one tradition of programming is represented in a team, it's likely that there'll be more than one program layout style in use. People who have been weaned on one layout style may have difficulty in reading code laid out according to another. Anyone, no matter what layout they are used to, has difficulty when more than one layout is used in the same module — or even in the same function! A common house style, after it has been implemented, can have several advantages:

1. Highly-paid programmers spend less time reading code to understand it.
2. Less time is needed to decide how to lay out a program.
3. Depending on the standards, information can be embedded in comments according to the standards that can be extracted automatically to document the program. (Of course, this can be done without a standard, but then it is much harder to design automatic tools for extracting the information).

For software houses who are developing C programs, this book may give guidance as to how to decide on and implement an in-house set of standards.

C program layout is a hotly contested issue, with many, many variations. These variations are teased out by the author, and each one explained. Where there is a clear advantage for one form or another, this is shown — for example, putting variable declarations one to a line allows comments describing them to be placed on the same line, and reduces the likelihood of error when changes are made — and on the other hand, where there is no clear advantage, the pros and cons are discussed, more or less rationally.

In addition to pure layout issues, some deeper style issues are discussed in part four. The advantage or otherwise of deeply nested control structures, use of De Morgan's theorem to rearrange Boolean expressions, use of (ugh) goto and other much disputed 'features' of C are mentioned. Moreover, design issues like testability and maintainability are also discussed, even though this is stepping outside of the pure coding arena. The bulk of the book, however, concentrates on how one can place braces, parentheses, comments, declarations and control structures to optimise readability.

The final chapter give generally good advice on how to introduce the use of a C standard into one's organisation without more than a few fights. The problem is that people are generally passionately attached to the layout they grew up with. Even though in the long run a change to use the same layout as everyone else in the company will benefit them, (and everyone else more) they do not want to change.

Tools can help a lot in implementing a standard. All the tools mentioned are UNIX or freeware utilities — emacs, rcs, sccs, indent, grep, lex, yacc, etc — some of which are not available on other platforms. If one is not familiar with these

tools and what they do, one could feel a little left out of the picture, but I imagine that will not apply to any of the people reading this.

Appendices give a sample style standard and a glossary to help in reading the book. The only problem with the sample standard is that I disagree with most of it!

TRANSACTION PROCESSING SYSTEMS

by KRISHNAMURTHY, E. V. and
MURTHY, V.K.
Prentice Hall, RRP \$56.95
ISBN: 0-13-928128-2, 1991

*Reviewed by
Frank Crawford
Australian Supercomputing Technology*

With the increasing dependence on both databases and distributed computing (often together), there is a need to ensure that all the transactions involved behave predictably in all circumstances. This book is aimed at describing both the principles and practices of both centralised and distributed transaction processing systems. It is designed as a book for postgraduates in Computer Science and MIS professionals and attempts to cover current research activity, which suits both of the authors, E. V. Krishnamurthy is a Professor at ANU and V. K. Murthy works for Fujitsu Australia Software Technology Ltd.

The book starts with a discussion of what a transaction processing system involves, which covers a field much wider than a database, but includes distributed file systems and even traditional kernels. It also describes some of the more difficult conditions, such as system crashes or communication failure, which must be reliably handled. In many of these cases reliably means aborted, but at least it should be predictable.

The following chapter goes on with the definitions necessary for the more rigorous study in subsequent chapters. It also outlines a method of describing transactions which avoid more complex mathematical models. This is followed by two chapters describing the issues of transaction processing on a single or shared memory system, including concurrency control, locks, timestamps and certifiers.

Some of the more interesting chapters follow, these are distributed transaction processing, which cover such areas as distributed locking schemes, deadlock detection and two and three phase commits. This is an area of active research interest and many references are given to current research projects. Finally, examples of both experimental and commercial systems are given.

This book is written as a text and as such there are exercises and extensive references at the end of each chapter. Unfortunately, the writing style is also very much in a text book style, very dry and difficult to read. I found I often had to re-read sections to understand what was being described.

On the whole this book certainly seems to contain valuable information, however, it is really only suitable as a text book, as both the content and writing style are not aimed at a wider audience.

SYSTEMS APPLICATION ARCHITECTURE Common Communications Support: Distributed Applications

by James MARTIN with
Kathleen KAVANAGH CHAPMAN/Joe LEBEN
Prentice Hall, \$89.95
ISBN 0-13-785908-2, 1992.

*Reviewed by
Rod Berriman
Leeds & Northrup Australia Pty Ltd
<rodb@lna.oz.au>*

First a word of caution - to read the title of this book carefully. This is not an overview of SAA, neither is it a detailed reference on IBM Communications protocols. It is a descriptive work on the communications services and APIs supported within SAA. The yet to be published companion volume will cover the protocols in detail.

Nevertheless, this is a valuable work on the structure and intent of SAA. For those uninitiated into the world of IBM, the book's seven parts gradually shed light on the thinking behind the logo, and the word which most comes to mind is "proprietary".

Chapter 1 reveals that SAA is intended to be an "open architecture", in the sense that it enables applications portability across a range of IBM's proprietary operating systems - VM, MVS, OS/400 and OS/2. AIX is notably absent from this list. The author makes valid criticisms of the lack of open-ness, and even the lack of fixed standards defined. SAA is very much a moving target. Even the broad aim of portability has been weakened by IBM's concept of "co-operative processing". This means you may require both an OS/2 PC AND an IBM mainframe to run an SAA application. The definition of "portable" is vague too. An application should port to an AS/400 processor "simply by making a few minor changes and recompiling..."

light reading for what can be, at times, be a rather heavy subject.

The author does not dwell too long on criticism, and moves on quickly to describe the component parts of Common Communications Support (CSS). Part I covers the Distributed Environment, wherein the heterogeneous parts are linked by communications:- either SNA or OSI. IBM's commitment to OSI, however, is restricted to the interconnection of non-IBM processors. You'll be sorry (or relieved!) to hear that ethernet is not supported, only X.25 and token ring.

Part II "Transmission Objects", part III "Data Stream Architectures", and part IV "SNA Application Services" elaborate in considerable detail the way in which documents and databases are distributed in an SNA environment.

Part V "OSI Application Services" overviews ACSE, FTAM and X.400 electronic mail - the only OSI services included in CCS (so far).

Part VI "Network Infrastructure" gets into the nitty gritty of SNA, token ring, and the OSI equivalents. This section is good reading. The various generations of L.U. are discussed, as are the intricacies of IBM's network topology, especially with the newer APPN architecture (Advanced Peer to Peer Networking).

Part VII concludes with a discussion of APIs available for SNA and OSI.

This book has its irritations - such as the use of pro-forma "conclusions" to each chapter. "This chapter introduced Chapter Z describes...". It is by no means a reference work either, due to the absence of details. However it does provide a good overview of SAA communications. It is

AUUG BOOK CLUB & PRENTICE HALL AUSTRALIA

20% DISCOUNT TO AUUG MEMBERS

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

- Straker/C Style**
RRP \$42.95* ISBN: 0131168983 Paper 1992
- Martin/Systems Application Architecture**
RRP \$89.95* ISBN: 0137859082 Cloth 1992
- Krishnamurthy/Transaction Processing Systems**
RRP \$56.95* ISBN: 0139281282 Paper 1991

*Deduct 20% from listed retail price

Name: _____ Organisation: _____

Address: _____

(Street address only)

Telephone: _____

Please send my book/s on 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

ORGANISER'S REPORT - AUUG SUMMER CONFERENCE - ADELAIDE 1992.

Attendance of 25 people, comprising

Admin:	2	Registration, Program Chair
Speakers:	5	
Invited:	6	
Existing Members:	8	
New Members:	3	(including 1 institutional)
Non-Members:	1	

After all the effort from various people in organising the conference, it was a good feeling to get underway with no major hitches. All the interstate speakers arrived, fortunately the airline refuellers waited until Friday to cause disruption. Glenn Huxtable as Summer Conference Co-ordinator came along to fly the AUUG banner (literally), and all the various projection equipment was still in working order. Many thanks to Santos for providing such a well-equipped venue.

After the formalities of registration and coffee, Ron Hoskin as Program Chair welcomed everyone, especially the new AUUG members, and introduced the first speaker, Ken McDonell. Ken's talk on tools for Software Quality Assurance was thought-provoking and got the day off to a lively start. Following morning tea was the first of the video presentations "Idea Power" - presented by none other than Vincent Price, superimposed on a background of haunted house with obligatory thunder and lightning.

It was a light-hearted introduction to the concepts of convergent and divergent thinking and their use in problem solving. Ken McDonell and Vincent Price were a hard act to follow, so as conference organiser I drew the short straw. Michael Selig followed next, presenting Functional Software's approach to UNIX System Management. Michael was able to illustrate the Object-Oriented concepts with some useful examples.

Lunch was an interesting variety of sandwiches and cakes from our local favourite supplier, and seemed to be enjoyed by all. The almond paste, banana & date sandwiches drew the most comments. To keep everyone awake after lunch, another video was scheduled - "Whose Idea was it Anyway". This is a very recent Australian production from the EDP Auditors Association on some of the pitfalls of introducing a new, all-singing, all-dancing, totally-integrated and paperless computer system into a company. The quality (and accuracy) of the production was excellent and gave everyone a good laugh.

Chris Clarkson followed with an overview of SNMP and a brief history of the Internet, which was particularly relevant given the number of commercial users in the audience interested in getting a connection. Ken McDonell was curious to know if anything other than toasters, Coke Machines or CD players have had SNMP agents implemented.

Hugh Irvine was the final speaker. His talk on a Wide-Area Network implemented using ISDN was informative and full of useful hints for taking advantage of this new technology. This brought the first AUUG Summer Conference in Adelaide to a close.

Thanks to all those who attended and assisted in making the event a success, particularly,

WAUG - for lending use their theme and flyers
The interstate speakers - for their time and effort
AUUG Secretariat - help and assistance
Di Williamson - organisation and preparation.

MICHAEL WAGNER
Conference Organiser

Melbourne Summer Conference

A Review

Michael Paddon

June 17, 1992

The Melbourne round of the 1992 AUUG Summer Conference Series was held at BHP Research Labs on the 28th of February. Attendance was excellent, stretching the conference facilities to near capacity.

There was no real "theme" to emerge from the papers presented; quite the opposite, there was a rather broad spectrum of diversity. This serves to underline the fact that Melbourne's Unix community is growing ever larger in size, scope and breadth of interest.

Papers

In reviewing the papers, I'll stick to the sequence in which they appeared on the official programme. The inevitable last minute problems and reshuffles meant, of course, that the marching order was somewhat different.

Engineering a Connection to AARNet **Peter Elford, Australian Academic Research Network**

Peter Elford's paper on AARNet covered far more ground than the title implied. He presented a large amount of detailed information on how the Australian backbone hangs together and how various network services are delivered to the user. This talk also provided an insight in the policies and future directions of AARNet. Last, but not least, Peter addressed the issues of becoming an AARNet affiliate: how to go about it, how much it costs and what to expect in return.

ISDN - Never Mind the Quality, Feel the Bandwidth **Scott Colwell, Labtam**

Scott Colwell recounted some real life experiences in trying to use tomorrow's physical network layer today. He exploded the myth of a unified ISDN standard and pointed out the problems of the resulting incompatibilities. Standards were not the only issue examined; Scott also pointed out the shortcomings of Telecom's existing services, both technically and from a business perspective. The audience was left with

a solid overview of how the basic components of an ISDN service fit together and a better understanding of strengths and weaknesses of the technology.

Computer Crime

Det. Sgt. Ken Day, Australian Federal Police

Ken Day's presentation of computer crime from an enforcement perspective was one of the best received papers of the day. After giving a quick overview of the relevant Commonwealth legislation, he proceeded with a question and answer format which drew forth some lively questioning from the audience. It seemed that everyone had a story to tell or an opinion to air. He drew to a close by inviting people with further queries or problems with system crackers to contact him.

Given the excellent response to this paper, it might be appropriate for AUUG to set up an open forum in which members can debate this issue and exchange information.

Disaster Mitigation in a Unix Environment

Graham Jenkins, IPEC

Graham Jenkins presented a amusing and eloquent analysis of the disasters that can befall an organisation using a Unix system for mission critical purposes. Interestingly, none of the failure agents listed (ranging from rogue cleaners to hardware to the general manager) were specific to Unix, although some of the avoidance and recovery measures were slanted towards our favourite operating system. A timely reminder that as we use Unix ever increasingly in commercial forums, we must also factor the overall environment into system design.

The Epoch System

Craig Bishop, Geelong and District Water Board

Craig Bishop's paper dealt with the problem of having to provide on-line access of many gigabytes of data to a large network of workstations and PC's. More specifically, he described the solution chosen by the Geelong and District Water Board – the EPOCH InfiniteStorage Management System. This is a dedicated NFS server with a high capacity optical backing store and magnetic disk caching. Craig explained the details of configuring and administering such a system; when to use WORMS versus erasable optical media, how files are tracked, how archiving and backups are done and how to tune for performance.

SNMP – Network Management

Peter Elford, Australian Academic Research Network

Peter Elford took the podium again to wax lyrical about network management. He discussed a variety of generic issues (including configuration, performance, fault handling, security and accounting) and went on to point out that no system can address all these requirements. He examined the problem from the perspective of using SNMP and/or CMIS/CMIP to provide fundamental support for tailorable network management. In passing, Peter mentioned some of the tools that are commercially available to the network administrator and pointed out that flashiness seemed inversely proportional to functionality. He also recounted numerous personal experiences from his work with AAR-Net, which lent a fascinating insight into the problems of running a truly large network.

Experiments in Network Distributed Image Synthesis

Steven Hayes, CITRI

Steven Hayes spoke more about distributed computing than image synthesis. The expense of generating photo-realistic images was used to demonstrate the need for mature distributed processing systems for easily partitioned problems. He gave a detailed analysis of half a dozen existing languages and environments in the context of his requirements for high-end graphics. Steven's paper was an ideal primer to the state of the art in heterogeneous distributed computing systems.

Remote Terminal Emulators for SQA

Ken McDonell, Pyramid Technology Corporation

Ken McDonell took a fresh look at every programmer's pet hate – software quality assurance. He proposed the use of a remote terminal emulator (a common tool in the benchmark world), along with standard Unix tools, to assist the testing of systems with complex user interfaces in a realistic fashion. Ken iterated the different classes of test that could be performed with a RTE, and then he discussed how an such a RTE could be designed to take maximum advantage of the Unix environment. He concluded by emphasising the fundamental advantages of a tools-based approach.

Shake – A Better Makefile Generator

Michael Paddon, Iconix Pty Ltd

I put in my own two cents worth, describing one of our favourite tools at Iconix. The shake system is little more than an m4 front-end to a collection of shell scripts that perform most of the footwork involved in writing a makefile. There were three morals. Firstly, the Unix tool

philosophy is affirmed yet again. Secondly, m4 is very powerful, very ugly and almost impossible to use. Last, and least, it's amazing what lengths a programmer will go to, not to mention the amount of code he will write, just to avoid a simple task like writing a makefile.

Panel

The last session of the day was the traditional panel. All of the contributors were seated in front of the audience and the floor was opened (try to visualise small generic furry mammals and rotating knives). In fact, this session was mostly consumed by questions relating directly to the papers presented, few new topics were broached.

The Aftermath

At the termination of official proceedings, the conference adjourned (with remarkable conservation of numbers) to the Notting Hill Hotel. A detailed post-mortem of the event was initiated in one corner, whilst others investigated the beer and wine supplies with vigour. A small, but hardy, group survived and was reported to be headed towards Chapel Street for pizza. They were never seen again...

Thanks

As I said in the conference programme, it takes a lot of hard work and guts to present a paper to a large and technically demanding audience. My thanks to all the contributors for making it all happen. Hopefully your efforts will inspire others to sacrifice some time a sweat next year.

I'd also like to publically thank Ian Hoyle, who co-organised the conference with me, and who certainly put in more than his share.

BHP strongly supported this conference by providing the venue, catering, printing and a plethora of minor resources. Thank you for contributing, in no small way, to the success of the day.

Finally, I'd like to thank the AUUG committee for supporting and encouraging the summer conference series.

"USING A MULTI-USER UNIX COMPUTER SYSTEM AS A FILE AND PRINT SERVER FOR MS-DOS BASED PERSONAL COMPUTER LOCAL AREA NETWORKS"

Ross Parish (TasAudit)

INTRODUCTION

Personal Computers running the MS-DOS operating system became popular in the early to mid 1980's. Most of these computers were used for single applications, perhaps being shared by a group of users. Some were connected to mainframe based systems via serial lines and were able to act as a terminal on that mainframe using appropriate emulation software, or transfer files to and from the host.

Later on, PC's were being connected together in the form of a Local Area Network, either peer to peer connections which allowed the sharing of data and resources such as printers, or with a dedicated server.

Access to multi-user systems was achieved either through direct connect lines from each PC or through a bridge or gateway from the PC server.

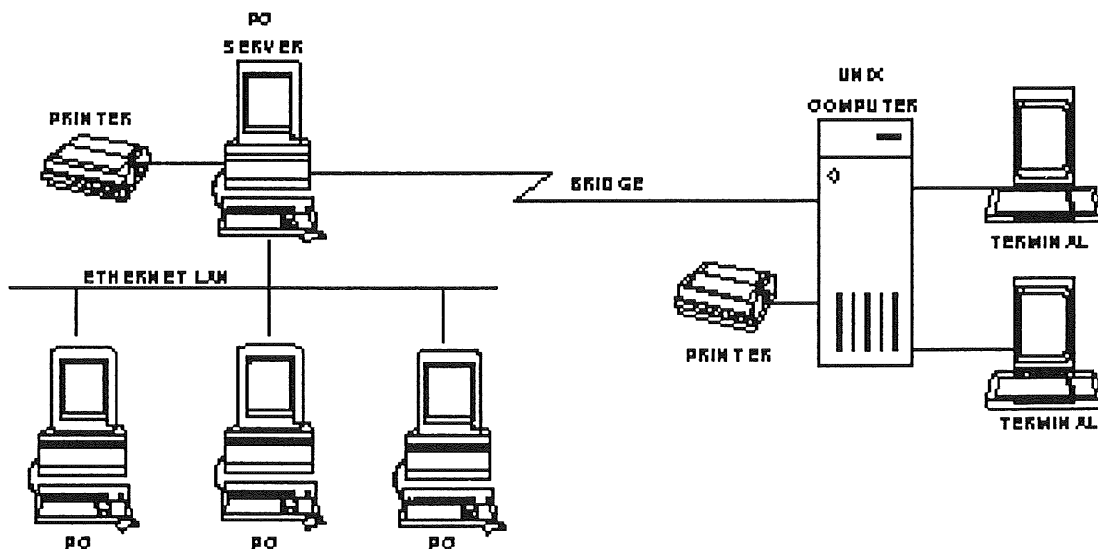


Figure 1 - PC Based Local Area Network with Bridge to Multi-user System

Unix is rapidly gaining in popularity in the business world. All businesses who are considering Unix have some kind of PC's running MS-DOS. Often large investments have been made on MS-DOS based equipment. Emphasis is now being placed on integrating.

As Unix becomes more common in business and as information is used and stored on both kinds of systems, many organisations have decided to link Unix and DOS.

Lans and multi-tasking systems are not mutually exclusive technologies, they can co-exist successfully.

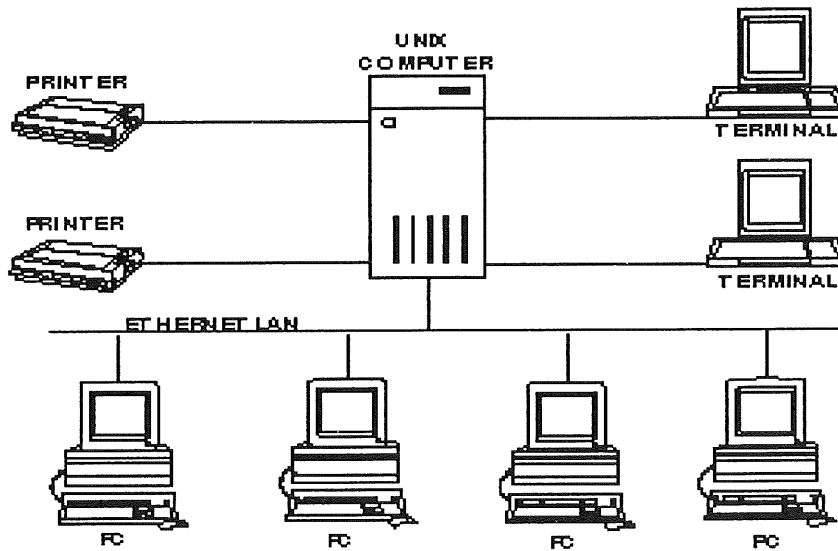


Figure 2 - Using the Multi-user Unix host as the PC Lan Server

WHAT ARE WE TRYING TO ACHIEVE?

The ultimate goal is real-time sharing of resources and information by a number of users. Both multi-user and lan situations have advantages and disadvantages. The aim is to provide systems without having to sacrifice the functionality of either.

DOS provides users with a variety of applications, user interfaces (mice, bit mapped graphics and windows). Sharing has only been available through LANS which are difficult to install, configure, tune and secure as the number of users increase. DOS only LANS are pressed to meet the price/performance of Unix systems. DOS based servers are single tasking, you often need to install additional ones for gateways, print spooling etc., whereas a single Unix machine can perform all these functions.

DOS provides users with packages such as word processing and spreadsheets, with a wider variety of options available and these packages are more polished and user friendly than their Unix counterparts. Organisations have often made huge investments in copies of Microsoft Word and Lotus 1-2-3 and dozens of other programs and utilities and in the experience gained through their use.

Unix has its advantages with data base systems and accounting and engineering applications. Unix provides multi-user, multi-tasking, record and file locking and data security.

A network that contains both multi-user Unix systems and DOS lans is easier to administer and keep secure (contains menu driven system administration, for installation, adding users and backup and allows the three Unix security levels to be applied to the DOS environment).

By integrating Lans and multi-user Unix systems you have everything to gain and nothing to lose.

There are two Options:

MS-DOS under Unix - i.e. DOS emulators. Products like VP/ix and Merge 386 give users the flexibility of DOS, Unix and Lans. These products will be discussed briefly later in the paper.

MS-DOS with Unix - i.e. Unix systems acting as servers. This is the main focus of this paper.

Objectives:

File sharing - both Unix and DOS users can have access to the same data files.

Easier access to data on other systems - through products such as NFS, DOS users can access any information that the host system has access to.

Enterprise wide electronic mail.

Cost savings - there is no need to purchase additional hardware to fill the role of a PC server. The cost of the network software is similar on PC and Multi-user platforms.

Better security - the multi-user system provides additional security over the PC Lan products.

Data management - as all corporate data is stored centrally, it is easier to classify, monitor and backup.

External communications - gateways to other systems are easier to achieve through a multi-user host which generally has these facilities built in.

Sharing of resources (e.g. Printers, High capacity disk drives).

HOW IS THIS ACHIEVED?

The attraction to achieve this sort of connection is not that it's Unix, but its the connection itself (access to electronic mail, large hard disks and fast printers) and it opens the option of DOS users running Unix applications.

The Unix based server controls workgroup functions such as supporting gateways remote communications, print spooling and supporting the Lan while still acting as the host of its own multi-user system.

The advantages of this approach include cost savings from using dumb terminals where local processing and storage is not required, but the flexibility to use PC's and PC users can share files and applications with terminal users.

The Unix system provides log-in security features for both Unix and DOS users and contains standard system administration software for installing and configuring the system.

You generally don't need to install an additional network adaptor in the Unix host as most have network connection (ethernet) built in.

Each PC does however require its own ethernet adaptor. Some products have a limited number of ethernet adaptors that they support. Choose a well known "non-proprietary" one, you never know when you may wish to change network software. A 16 bit adaptor can be purchased for around \$300 ex tax. The current trend towards laptop or notebook computers is also catered for with external adaptors. These generally attach to the PC's parallel port and although the performance is not quite up to that of a card type adaptor, it is certainly adequate for most applications. Some notebook computers now come with an adaptor slot to which an fully function external adaptor can be attached. These cost a little more but give you the added flexibility.

Once the decision has been made and an adaptor is obtained is where the fun starts. The installation of an ethernet adaptor in a PC is not quite as easy as it would first appear.

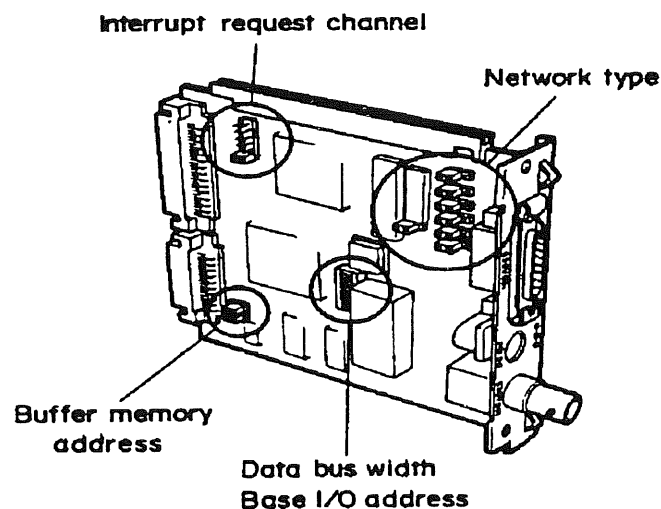


Figure 3 - Ethernet Adaptor

There are generally three parameters that need to be configured on an ethernet card, the shared memory address - a 16k segment of DOS memory between 640k and 1Mb which controls the network packets and is used for buffering, the I/O address and the IRQ channel (interrupt). Often jumpers on the ethernet cards themselves need to be changed - refer figure 3.

For the first time user, this can be a daunting task. You may be lucky and purchase an ethernet adaptor that is configurable by the command line.

Getting these right is often the most difficult aspect of connecting the PC to the network, not the network software itself. i.e. overcoming conflicts with other programs, video adaptors, serial ports and other add-in cards. For example, if a PC has a second serial port (COM2:) this uses IRQ3 (the default or factory setting for most network cards). You have to either disable COM2: or go through the painstaking process of choosing another combination of parameters.

The problem is compounded in organisations where several different PC types exist - they may have different video adaptors which use different areas of high memory - and some products have a finite number of combinations that you can choose.

There are now products on the market, such as Manifest from Quarterdeck which allow you to determine what is using those areas of the PC's internals and to assist in resolving conflicts.

You have to determine whether your Unix supplier is a total solutions vendor (will probably cost you more) or source components individually. It is a good idea to purchase one or two cards from the Unix vendor initially (you can at least hold them responsible for getting the thing together in the first place) and then experiment with the 3rd party components once you have gained the experience.

WHAT AM I LOOKING FOR FROM THE PRODUCT?

Easy to install - these days most Unix systems come a system administration menu which includes an option for the installation of software packages. The packages are written with an install process which interacts with this 'sysadm' utility. This allows the person installing to answer a few questions and the rest of the installation is largely automatic. The PC installation should also have an install program that leads the user through the various options. The PC install is the usually the most complicated due to different ethernet adaptors and PC's (see above).

Performance - including little overhead on the "host", amount of memory it uses in the PC. Unless the product has little affect on host users and applications then the purpose of the integration is defeated.

Ease of management - the provision of easy to follow system administration functions is an essential component.

Security - the need to be able to identify users and apply (at least) the Unix security levels to the DOS files. Also the ability to tailor individuals particular security requirements. e.g. set an individual Unix "umask" for each user.

Flexibility - does it give access to different areas of the Unix file system? Can users easily load and unload the network software as required?

Integration - how do I recognise that the files I have created in DOS are owned by me when I am logged into Unix? The ability to exchange files between the two environments and the extent to which this can be done transparently.

Interaction between DOS and Unix versions of products. e.g. Office Automation software, electronic mail.

Terminal emulation - I need to be able to act as a terminal on the host to run these applications? How easy can this be done? What emulations are supported? Can I tailor the options? Key mapping is an important consideration here.

Remote procedural calls - I need to be able to run a "one-off" Unix command without having to log on to the host. The ability to run a single (non-interactive) command on the Unix host from the PC such as check a mail queue, check the status of a print-out, or whether Bill Smith is logged on.

Closely resembles the DOS environment. What we don't want is it does X but it doesn't support Y. This is particularly the case for where you have an existing PC Lan software and are integrating a Unix server into the Lan.

WHAT PRODUCTS ARE AVAILABLE?

PC Interface (PCI)

From Locus Computing (now bundled as part of SCO Unix V.4 and Open Desktop).

The Unix segment is easy to install, there are few parameters (you can buy it in 1-16 and 17-32 user versions), the install process does rebuild the Unix kernel (doesn't everything), but the Unix processes are not particularly heavy on host resources. There are two parent processes that are started as part of the Unix system startup (one to serve DOS file and print serving and one for terminal emulation) and a child process is spawned each time a PC runs the PCIINIT program or the terminal emulator. There has to be an entry in the /etc/hosts file for each PC on the network in the Internet format of xxx.xxx.xxx.xxx.

The PC configuration involves only two device drivers to be placed in "config.sys" - there is an install program, but it does require some knowledge of the concepts surrounding Ethernet adaptors. The number of ethernet adaptors supported is limited. The actual connection to the host is achieved by a PCIINIT program which must have as one of its parameters an internet address corresponding to an entry in the /etc/hosts file.

PCI requires you to purchase a separate PC version for each PC. i.e. if you only have 5 PC's, you have to purchase the 1-16 user host version, but you only have to purchase 5 PC versions. One of the device drivers has a unique id, which means that you cannot boot two PC's from the same disk. The host version does come with one PC version.

Maps the whole Unix file system as the next available DOS disk on the PC, or to a designated drive letter as an option.

Has its own terminal emulation product (called em), which is only vt100 based. However, many third party emulation products now have an option to work with PCI networks. The PCI terminal emulation does not support TCP/IP, it requires the installation of its own terminal devices (/dev/ptyp0 etc.).

Will work over RS232 serial communications (however it is slow). Is useful if you have a remote site and wish to transfer the odd file or act as a terminal. Don't try running Microsoft WORD off the host over a modem at 1200bps!

Has a Unix to DOS and DOS to Unix file conversion facility (UNIX2DOS and DOS2UNIX).

Has a facility to run non-interactive unix commands from DOS (called "ON"). It also has separate re-direction and "pipe" functions so that you can combine the DOS and Unix syntax if necessary. ON.EXE can also be copied to be the name of the Unix command, e.g. GREP.EXE so that the DOS user doesn't have to understand the syntax of the ON and the Unix commands - i.e. they think that they are DOS commands. A DOS environment variable UPATH also allowed system managers to set up so that full path names of the Unix commands were not required, and an EXPORT variable allowed you to set the Unix environment (e.g. terminal type) needed to run some Unix commands. One Unix command that was not available using ON was 'newgrp', which would have been useful in overcoming one of the limitations of the Unix security system.

The environment was simplified due to the fact that it used Unix. i.e. security was based on the Unix system, the /etc/passwd and /etc/shadow files were used for access and the Unix printer spoolers were accessed directly.

Printing was facilitated by three logical parallel ports (LPTn:) which could be directed to any of the Unix printer names.

PCI is a good intermediate choice - where serial connections are not enough and you don't require a full scale network operating system. It is cheap (relatively) and has an attractive simplicity. It is most suited for organisations with a smaller number of PC's.

However, PCI is a big loser in respect of the fact that it has not kept up with the latest versions of DOS based network (and in some cases non-network) software. e.g. Word 5 would run but you couldn't save, Lotus could not be installed because the number of licensed users was stored as a hidden file, which were not supported.

If Locus were to overcome these deficiencies (they would have to in order to stay in business) then PCI would meet the majority of organisations needs.

PC NFS

This is the most obvious product available, due to the success of NFS in the Unix world, and for Sun users perhaps the only one available until recently.

There is no host product required just alteration of the /etc/hosts and /etc/exports files to account for the PC's running PC NFS.

There are however many device drivers that need to be installed on the PC and this can cause problems with memory availability for PC applications.

It allows users to 'map' to several Unix sub-directories as different logical DOS drives and thus uses the concepts of NFS.

There are some doubts about security and even Sun documents these concerns in the manual. For example, once the PC's are booted up, there is no requirement to log in, but on execution of the appropriate "NET USE" commands, the file systems are accessible. You do not have access to files which are secure, but public files (Unix mode rwxrwxrwx for example, i.e. able to be accessed by 'other') are available.

Three printers are accessible from three logical ports (LPT1:-LPT3:) also using the NET USE command.

Most of PC NFS apart from the file system mapping draws from PC Interface, it even uses the same commands (although some have different names and have been modified slightly e.g. rsh, telnet) and there are several copyright Locus Computing messages all through. Telnet is the vt100 emulator that draws from PCI's "em", but it does actually use TCP/IP and the standard unix devices.

There is a better user interface than PCI, with menu driven connect, map etc. features.

It is therefore hamstrung by some of PCI's limitations, however the data save problems do not exist.

It is relatively expensive, you have to buy a copy of it for each PC and there is a problem with upgrading as you have to upgrade each copy.

The latest version has remote printing, also has a revised version of telnet that provides additional emulations e.g. vt220. This "advanced telnet" product is however an additional cost of approximately \$250.

An add-in product called 'lifeline' can be purchased which provides better security, allows back-ups of PC's to the server devices and has mail facilities.

PORTABLE NETWARE (PNW)

Portable Netware was originally dubbed "Netware for Unix", however due to the fact that Novell was targeting markets other than Unix (e.g. VMS) its name became Portable Netware. It is however referred to as Open Netware by some vendors. It is available from a number of vendors, the most recent being Sun.

Is an implementation of Netware 386 running under Unix. Novell re-wrote netware in 'C' so that it could be ported to other platforms, including Unix.

Novell have co-operated with a number of Unix vendors in the production of Portable Netware.

PNW contains most of the features that exist in Netware 386 - some have not been implemented (simply because they are not necessary) and others have Unix equivalences. Those that don't exist on "fconsole", for example, are file lock activity, LAN Driver Info, Purge and Statistics. One excellent Netware feature that is retained in PNW is that to connect to the network, no device drivers are required in "config.sys". Two DOS TSR (terminate-and-stay-resident) programs are executed. This provides the PC user with the flexibility of not having to be in the network all the time (and additional memory as a result). The two programs also take up little memory and later versions allow these programs to be loaded into high, extended or expanded memory.

Other netware equivalences (printcon, pconsole etc) are fully available as are system login scripts, send message etc. Some commands such as userlist also run under Unix.

As Portable Netware is fully compatible with other versions of Netware, a PNW server can be added to an existing Netware LAN. This opens Netware users up to a range of options. There are some problems if there are Netware 2.11 Servers on the LAN.

Portable Netware involves mapping logical DOS drives to directories under the Unix file system including NFS volumes if required and connecting logical PC Printer ports (LPT1:, LPT2: and LPT3:) to the Unix print spoolers.

PNW also may come with a number of add-in or enhancement products (these are not developed by Novell, but are third party packages and often are marketed by a specific vendor). Examples of these products are gateways including a IPX/SPX to TCP/IP gateway, which allows the PC to act as a terminal on (i.e. telnet) or to transfer files to or from (i.e. ftp), any host on the network, including non-Unix hosts that support TCP/IP, a MHS to X.400 gateway product, Arcserve - a program that allows PC's on the network to be backed up to host devices (disks,tapes) in unattended mode and Monitrix - a network management tool which can display network activity in a graphical mode, or export traffic and error statistics to PC products such as dBase and Lotus 1-2-3.

PNW does have complicated configuration requirements and although the Unix end is driven by the standard 'install package' utility it is rather long winded.

PNW requires the system administrator to set up the Netware users and group security as well as the Unix security. This feature can be both an advantage and a disadvantage. If you have PC only users it can be done without setting them up as a Unix user, however, if you are setting up an environment where the DOS and Unix users are sharing data, it is better to limit the extent of the netware security setup and let Unix handle the security. PNW handles this feature through a facility called 'Hybrid Users Configuration' - it allows the system administrator to map the Portable Netware user to the corresponding Unix user. All files created under Netware then become owned by the Unix userid rather than netware.

The way to set up a PNW system will depend on whether you are adding to an existing Netware site (in which case the familiarity with Netware will allow this environment to be managed more easily) or whether you are adding Netware functionality to an existing Unix system (where the system manager will probably prefer to limit Netware functionality and use the Unix features to control the system).

Printers under PNW use the Unix print spoolers. Print-outs are passed from the Netware queues to the Unix 'lp' spooler system. This makes the netware spooling somewhat redundant due to the fact that although Netware thinks it has printed, it may still be sitting around in a Unix queue (or have disappeared into thin air!). This also has other advantages - you get the choice of using Netware facilities for configuring print jobs or using the Unix interface programs. Remote printing is also supported under PNW (i.e. ability to print to a printer attached to a client PC rather than the server).

PNW also comes with a feature called Novell Virtual Terminal (NVT). This allows PC users to act as a terminal to access the Unix host. It is bundled, having been developed by Novell, and although the TCP/IP gateway allows access to multiple hosts, it has to be purchased separately, whereas NVT is bundled and should meet most users requirements. NVT can be used along with any terminal emulation product that supports the 'interrupt 14' type network interface or you can purchase a NVT emulator product at the same time you purchase PNW. The latest version of PNW supports multiple Unix sessions under NVT.

One of the aspects lacking with PNW was the absence of a 'Remote Procedure Call' product such as PCI's 'ON' utility. This has now been addressed and some vendors are shipping a facility called 'Upoint' which is a point and click facility to allow DOS users to perform Unix type functions on their files and to run non-interactive Unix commands.

What's wrong with it?

Inconsistent workstation ID - each PC has a different ID depending on which one was connected following the network startup. This is a minor problem, however for management purposes it would be an advantage to know which users is using which PC.

There is quite a bit of additional management required. i.e. to backup the binderies (the data base of permissions) and shadow files (special directory entries which keep the Unix and pseudo DOS file systems in sync.)

LAN MANAGER FOR UNIX (LM/X)

LM/X has much the same features as Portable Netware.

It does however require lots of device drivers (in "config.sys") so DOS memory is down to around 515k.

It is a Unix based implementation of Lan Manager and hence maintains the standard LM security. There is a 'hybrid' user facility.

LM/X comes with TCP/IP for telnet terminal emulation (vt100 only) and ftp file transfer. You can however use "off the shelf" packages that support TCP/IP. Once you're in a terminal emulation session you can't hot key back to DOS. This loses some of the flexibility that is available in some of the other products.

LM/X allows you to map logical DOS drives to any Unix directory, including NFS volumes.

The Unix based system management menu facility is excellent.

LM/X does not allow user login scripts (you have to put everything in DOS batch files). The next version is supposedly going to have this facility available.

There is a RPC facility called Client U-Commands, which allow Unix commands to be executed without having to establish a terminal emulation session.

Printing is available to Unix host printers or to local printers. This latter facility works extremely well.

There are no network monitoring programs or backup options bundled.

The interaction of Unix and DOS versions of applications is supported through API's.

There is a menu driven Unix install which is fairly straight forward.

Comparison LM/X & PNW:

The purpose of this paper is to give an overview of the concepts involved and to list what options are available - not to draw comparison between whether X is better than Y. However, the PNW vs LM/X question is one that is often raised. I am not going to pass judgement, except to quote the following facts:

53% of PC LANs use the IPX protocol (Netware), 12% use NETBEUI (Lan Manager) and 10% TCP/IP.

PNW is compiled to run on a particular platform. But PNW vendors add communications gateways and X.400 services (these are not certified by Novell).

LM/X is a rewrite, not a port - individual vendors versions of LM/X won't inter-operate.

OTHERS

There are other options available, too numerous to evaluate in detail. The four products listed above are the mainstream products and fall under the ambit of a total solution. Other products may provide only limited functionality, but may meet a particular sites requirements.

TotalNet - a flexible and integrated solution

PCSA - This is Digital's proprietary version that runs under VMS (& Ultrix?).

PrimeLink - can run under Unix or use a PRIMOS host as the server.

PC/TCP - a popular and cheap alternative.

WIN/TCP

Ice.TCP

(The June 1991 issue of Unix World US compares some of these latter products in terms of functionality - it also has an article on Portable Netware).

WHAT ARE THE ADVANTAGES?

Controlling End User Computing - at last the DP manager has a way of ensuring that all the PC's are accounted for, data can be shared and distributed across the network (rather than using "frisbee net").

"Corporate Data" - it allows for single copies of data bases etc. to exist and to be maintained as a 'master record' rather than individual users versions on individual PC's.

Systems Integration - through products such as the gateways, there is an ability to integrate not only the DOS and Unix environments, but other environments as well.

Resource sharing - managers can maximise the benefits obtained from purchasing that super fast laser printer, by making it available to everybody on the network.

Less redundancy - of data and resources (i.e. disk storage, printers).

Central backup - makes it much easier for recovery of lost data.

Flexibility - the options are endless. If your user only requires to access Unix applications, give them a dumb terminal. If they are a power user, give them a PC running Windows and through these integration products they are able to organise their desktop to switch between tasks and maximise their productivity.

Better Security - users can be restricted to the data and the applications that they need, sensitive data can be stored in a secure environment, rather than on the disk of the individual PC and the Unix operating system adds an additional dimension of security over the Network and DOS security.

DISADVANTAGES

Disaster/Recovery

If the server goes down, your network is out of action. This is especially the case where the PC's do not have hard disks and therefore rely on the server for software as well. PC's with their own disks can however load software and continue to work in the local environment whilst the network is unavailable.

Performance

Although with the newer and more powerful RISC platforms this is becoming less of an issue, it is still unlikely that performance will match that of a dedicated PC server. However, as the vendors of these products all point out - if you want dedicated server performance then don't buy the product - the product is aimed at a specific market - those organisations who wish to integrate their Unix computers with existing PC servers (in the case of PNW and LM/X) or don't have the requirement for full blown PC network features but wish for their PC's to have access to a wider range of Unix based resources (high speed printers, larger disks, applications and wider communications).

A big minus for the network connection is that its always there. (i.e. uses valuable memory - users cannot strip out device drivers to give that memory back when they're not using the Unix resources). For PC users this can quickly throw cold water on the advantages of the network. The 'RAM cram' problem can however be overcome with software such as loadhi, xma and emsnet and Windows.

UTOPIA - WHAT IS NEEDED?

The products explored do go a long way in maximising the benefits available in both environments, however, do they go far enough?

Better Seamless Integration

Although the products do protect the user from the Unix operating system and merely extend their existing environment (i.e. DOS), one feature that is lacking from the products is the ability to launch a Unix applications without having to go through the processing of loading terminal emulation software and logging on to the host (and then perhaps being presented with a further menu of choices).

What is required is a process similar to the Unix 'rlogin' process where a user is identified on two unix hosts and the rlogin command issued from one host causes the user to be automatically logged on to the other. This could be achieved by an extension to the "hybrid" user concept.

The difficulty for vendors such as Novell and Microsoft is that they want their products to be merely implementations of their DOS (or OS/2) based products running under Unix and the more functionality they add negates this objective.

Better use of the "Hybrid" user concept

Most of the products have a configuration parameter that sets the default Unix 'umask' for all DOS network users. The hybrid user concept should be taken a step further to allow more tailorability of individual users.

Remote Procedure Calls

This is considered to be an essential element of any package - it can protect many users from the Unix system and emulation products.

Network Monitoring Tools

Although some vendors have added this feature (which has often been developed by a third party) it should be part of every networking product.

THE FUTURE

X Windows

There are already a number of products available that allow your PC to act as an X Windows terminal or server (e.g. X Vision*, X11/AT*, HCL-Exceed, PC Xsight, PC XView, Desqview/X).

* - run under MS-Windows

Client Server

Being able to run the PC as a front end to an application running on the host supported by the Network Software's communications protocols. There has already been some work done in this area.

OSI and distributed applications

Commentators say that OSI will be too large for DOS, as the RAM cram is already a problem. However most vendors have plans to implement their products using OSI protocols.

Distributed applications - only email is presently available which is most popular on TCP/IP and Novell. Other applications will emerge.

Multi Tasking

Products such as MS-Windows, Desqview or JSB Multiview Desktop allow the PC user to switch between DOS and Unix applications. PC-Connect and TransActor allow multiple Unix sessions under Windows.

SUMMARY - CHOOSING A PRODUCT TO SUIT YOUR NEEDS

There are plenty of options available. The paper has listed what to look for in a product and what each product has to offer - choose one that best suits your requirements.

APPENDIX:

DOS EMULATORS

PCI has a sister product called "Merge 386" which is a Unix based software package that allows some non-graphical MS-DOS software packages to be run from a "dumb" ASCII terminal (Wyse 60 for example). This is useful if you have some users with these terminals and they have a (limited) requirement to access files used by PC users. The package works quite well, can access pseudo expanded memory and is individually tailorable, by having users autoexec.bat and config.sys files in their "home" directories. The main problem with Merge is the number of products that will run with it are minimal, the number of terminals supported are small and some ill-behaved DOS applications cause the Unix system performance to be degraded (e.g. MS-WORD version 4 has a keyboard driver that continually polls the CPU, which has the effect of causing CPU time to almost equal elapsed time).

Merge 386 is bundled with some versions of Unix V.4.

You are probably better off buying the Unix version of the application that you wish to run, if it is available.

There is also a similar product called VP/ix which is the sister product to Interactive Unix and IBM has PC Simulator 6000 for its RISC 6000 range. Others such as XDOS (which takes a DOS based application and re-compiles it) have appeared in recent years. Be warned, there are more limitations/restrictions with these products. For example ill behaved DOS programs which take advantage of the hardware features of PC's - cannot be expected to work under Unix.

--

Softway Engineering or Structured UNIX Kernel Hacking

Peter Chubb

June 5, 1992

Abstract

Softway is in the process of changing the way in which we produce UNIX software. In the past, we've relied on individual expertise and commitment rather than on particular methodologies. As we've grown, and the size of the projects we undertake has grown, this is no longer adequate.

Beginning gently, project by project, we've been introducing a company-wide Software Quality program, with the eventual aim of complying with AS3563. Initial experiences have been good: we always knew we were producing good software, but could never prove it. Now we can.

1 Introduction

None of the design and software engineering methodologies that I was taught about at University seem quite to fit UNIX kernel development work.

When developing UNIX kernel enhancements, the key features are:

- A large body of existing code that mostly works.
- Few comments, and some of them misleading, in the code itself.
- No, or very little, design information about the code.
- A staff that needs specialised knowledge of a wide variety of fields that impinge upon the kernel's code, including hardware design, database systems, security, real-time expertise, schedulers, etc.

The kinds of people that make changes to the UNIX kernel are also somewhat different from the normal software engineer (if there is such a thing). The stereotype of a bearded gentleman, rather poorly dressed, who is impatient of formality, and sees it mostly as getting in the way is not too far from reality.

As such, introducing Quality techniques for UNIX kernel development poses a few special problems. In the remainder of this document I give a (necessarily brief) summary of some of the ways we at Softway are trying to improve the quality and the perceived quality of our software.

2 Traditional Softway Software Development

Software engineering is the process by which quality software is developed to a budget. As such, it involves being able to

- extract from customers a good idea of what they really want.
- estimate the time and effort involved in producing a given piece of software.
- produce bug-free (or almost bug-free) software with the bare minimum of resources.
- measure the process so as to do better next time.

At Softway this has traditionally been done by relying on individual expertise. Softway has always employed some very highly skilled individuals; people who are able to look at an incomplete requirements specification and say, ‘Oh, that’ll take fifteen person-months’, and others who can code directly from a loose requirements specification and produce code that is readable, well-commented, and close to bug free.

As we’ve grown, we have been tackling bigger projects — ones that are closer to ten person-years than one — and we’ve had to employ more staff to cope. This has meant that the proportion of people in the organisation who can estimate reliably, code and design on the fly, and test adequately has fallen. Moreover, as the kinds of jobs tackled have changed, the old wisdom has become inadequate.

Consequently, we’ve had to introduce a set of more formal techniques to ensure that the products we deliver are of the highest possible quality given the budgetary constraints we operate under.

We’ve now run four projects under the new regime — a total of 30 man years, or thereabouts — and so are in a good position to comment on ways of migrating to a more formal approach to software development.

3 Traditional Softway Methodologies

Traditionally, when a job arrived (usually because someone at Softway heard about a job that *might* be coming up) someone would be assigned to find out what the customer wanted — usually the same person that found out about the job in the first place.

Having found out as much as possible, that person would write up a requirements specification and project plan as a basis to quote from. If the quote was accepted by the customer, depending on the size of the project, one or more extra people would be assigned to it, and they would spend some time discussing ways and means of meeting the requirements, and would then go ahead and implement software in prototype form.

The prototype would be evaluated informally, and either thrown away or beefed up to match the requirements.

This classifies as a dynamic programming environment, according to Cem Kaner:

A dynamic environment is characterised by a budget that is too small, a staff that is too small, a deadline that is too soon and can’t be postponed for as long as it should and by a shared vision and commitment among the developers.

In a dynamic environment, the quality of the product lies in the hands of the individuals designing, programming, testing and documenting it; each individual counts. Standards, specifications, committees, and change control will not assure quality, nor are they relied on to play that role. The development team does make some agreements, but it is the commitment of the team's members to excellence, their mastery of the tools of their craft, and their ability to work together that makes the product, not the rules.

The development team has a vision of what they want to create, and a commitment to getting as close to the dream as they can. They recognise that they'll have to flesh out details by trial and error. By the time they've worked out the final details for one aspect of the product, they have a working version that one or two key people understand fully. That version is the "specification".

I've quoted these paragraphs because they show the good points about this work style — the key features for a successful outcome are:

- A small closely knit project team, so that communication is easy and informal.
- A team consisting almost entirely of very good committed programmers, so that very few bugs are introduced during coding.
- Well understood problems, such as porting UNIX or writing a device driver.

At that time, Softway was involved in research work. At one time, 30% of our staff were working on one product, which, at the time, brought in no income, and didn't really have any deadlines. It did have a team of people passionately devoted to making it the best technical product possible.

It was a UNIX kernel product. It had no fixed project plan, and therefore could never finish — it never reached a milestone, because there were none.

However, it was making progress. But the progress could not be measured.

The problem arose when we wanted to sell it to somebody...

3.1 Problems with Laissez-Fair Development

There are several major problems with this development style:

- **Scope** There is no way to limit the work that has to be done. If an extra feature pleases the project team, it is inserted.
- **Traceability** There is no way to say "*This is to match that requirement,*" so work can be wasted in doing unnecessary things, and likewise, some requirements can be left out.
- **Measurability.** There is no way to tell how good the delivered code and documentation really is — so there is no way to tell if new development techniques are having any real effect. Moreover, there is no way to tell our customers how good we are.
- **Quality.** Everyone has off days — days when the error rate quintuples, and stupidity reigns supreme. In a laissez-faire development environment, there's no mechanism for catching major blunders like these (although errors which prevent the product from working will generally be caught, later rather than sooner).

- **Testing.** Without firm requirements, there's no way to do any serious integration or acceptance testing. Moreover, programmers are probably the worst people to design tests for their own code, because they are very bound up in it. They (we) find it difficult to step back and look at the entire problem. For that matter, everyone I know finds designing test strategies and test cases very tedious.

There are also a number of advantages:

- **Paperwork.** There isn't any. People working on a project are free to spend 100% of their time thinking, designing, programming and testing. (Of course, this doesn't mean that they do!)
- **Flexibility.** The people working on the project can always slot some more work in to meet a new requirement, or delete one that has since become unneeded.
- **Freedom** For any given project the most appropriate design techniques can always be chosen. People working in this 'methodology' tend to design on-the-fly, so everyone gets to do creative work.

4 The New Methodologies

How can the advantages be kept, while eliminating the major disadvantages?

The approach taken was to take the AS3563 standard and move gradually toward development and quality assurance methodologies that would satisfy it, while attempting to retain the best points of our traditional approach.

4.1 AS3563

The standard provides a framework for developing quality software. The meaning of 'Quality' here, is "Meets Requirements".

The standard does *not* mandate particular design, analysis, project planning or quality assurance methodologies. What it *does* specify is that for any given project, the methodologies used shall be written down in the documentation for that project.

Rather than start by writing the Quality Manual that should lay down the law about all the officially approved standards for developing software at Softway, we decided to start off trying to work out what standards are appropriate for the kinds of work we do. As we're all practical types, we started off trying to use the IEEE standards for project plans (IEEE 1058) design descriptions (IEEE 1016) and requirements specifications (IEEE 830).

Use of the project plan and requirements specification standards were both reasonably successful after we worked out how to use them. The design standard was not, for reasons outlined in the next section.

4.2 Design Techniques

A design tool has inbuilt mechanisms for cross checking design against requirements. It allows one to structure one's material such that all the information necessary to think about a design at a particular level is in a compact form – at most one or two pages. It provides a framework

to work in that aids creative thought. The one feature common to all good design tools is their power of expressing and manipulating complex relationships in a small space.

The IEEE standard for software design descriptions is just that — a documentation standard. Unfortunately, we didn't realise this, and tried to use it as a design tool.

A documentation standard says what sections have to be present for a complete description of a design to various groups: to the quality assurance people (we didn't have any!) to coders, and to designers of flow on documents (design flows on to code and test design). In order to fill in the sections, *the design already has to be finished*.

We lost a lot of time because we didn't realise this, and tried to use the document standard as a design methodology — a way to think about things in order to do a design — rather than as a documentation standard.

4.2.1 Our design methodology

Most usual design methodologies in the literature are not suitable for the kinds of work we do. This work usually involves changing the UNIX kernel — a large program with well defined interfaces — and usually under the constraint that the changes be as few as possible.

To this end, the 'hooks' style of programming, and rapid-prototyping seem to be the most appropriate way to satisfy everyone. *Hooks* are function calls inserted into existing kernel code to functions that will modify the behaviour of the existing code.

The prototype, usually on an Intel 80386 machine, has the functionality (but not the performance) of the 'real thing'. Code can be prototyped quickly in the Intel kernel (using suck-it-and-see methods), and when it behaves properly, ported to the mainframes we do most of our work on.

In the excitement of getting a prototype working, quality requirements can be forgotten — anything goes if it works — but it is very important that the real thing be as close to bug free as practicable.

To this end, we introduced *reviews*.

4.3 Reviews

The most useful quality assurance method we've found is 'reviews'.

A review is an opportunity for many people, not just the author, to scrutinise a document (a document can be a piece of code, a design, a requirements analysis, a test harness — almost anything) and criticise it.

Before a review, the material to be reviewed — code, design, test-specification or whatever — is distributed to a select group of people, including:

1. the author
2. a moderator/chairman
3. some reviewers.

The reviewers are chosen based on who's available, and on who has the background to be able to review the documents effectively.

These people are given, along with the document, a checklist of things to look for. (In early attempts at reviewing, this checklist was not developed, and people mostly found spelling and grammatical errors, not real problems). They look for problems in the following main areas:

1. Conflicts with the input documents (for code, that would be the design document; for a prototype or a design, the requirements specification, etc).
2. Conflicts with the standards being used for the project (The Softway coding style guidelines, the IEEE document formats, or whatever).
3. Feasibility — whether the design, code or whatever will work properly.
4. Robustness — whether the design, code or whatever will continue to work if various changes in its environment occur (like porting from a single to a multiprocessor, for example).

In addition, specific documents have their own checklists; and on a project by project basis extra items can be added to the checklists for particular project phases.

We are still learning about how to make reviews effective.

4.4 Bug Reporting

Softway personnel are notoriously bad at reporting bugs. When they find a bug, they tend to fix it without telling anybody.

We wanted to keep track of how many bugs were found in various projects, so we¹ invented an incentive scheme.

The first person to fill in a bug report form for a new bug received a chocolate bug when the bug was confirmed by another member of the team.

This had two effects:

1. People were a lot keener to find *and report* bugs.
2. A friendly rivalry grew up around the number of bug carcasses attached to one's X-terminal, and hence the number of bugs found.

The first effect was the important one, however — with the reward only given for a completed form, some analysis could be done on the kinds of bugs being found.

We found in our first project using the technique that of the bugs we found, 45% were found in reviews, and the remainder in testing.

4.5 Initial Reactions

Initially, people didn't like all these standards, reviews and so forth. They were seen as subtracting time from the more important business of getting the next deliverable ready. However, as we get better at reviewing material, and better at using the standards, opinion is coming round to agree that reviews are valuable.

However, we have changed the way we review material. For a start, all material to be reviewed must be desk-checked by someone else first — this stops us wasting time on stupid errors. Also, code must pass `lint` with no error output before being reviewed.

¹Lucy Chubb and Paul Brebner came up with the idea in the first place.

5 What comes next?

Our aim is to continue to improve as much as possible to produce quality kernel and systems-level UNIX software. We want AS3563 compliance, but we also wish to maintain some of the old Softway spirit. We're still working out how to do this effectively.

References

- [1] *Software Quality Management System — Part 1: Requirements*. AS3563.1-1991.
- [2] *Software Quality Management System — Part 2: Implementation*. AS3563.2-1991.
- [3] *IEEE/ANSI Std 1058-1987 Standard for Software Project Management Plans*.
- [4] *IEEE/ANSI Std 1016-1987 Software Design Descriptions*.
- [5] *IEEE/ANSI Std 830-1984 Software Requirements Specifications*.
- [6] Cem Kaner. *Testing Computer Software*. TAB Books Inc., 1988.

Not just another add-user script: Automating user administration with perl

Janet Jackson
<janet@cs.uwa.edu.au>

Department of Computer Science
The University of Western Australia

1. Introduction

I have developed tools to automate the following tasks.

- Keeping track of temporary users.
- Adding a user.
- Adding large groups of users, such as classes of students, in bulk.
- Removing a user.

I wrote the tools in perl (see [1]), which I found to be good at the tasks involved, such as dealing with site-specific peculiarities.

My environment

As I write this, at UWA Computer Science we have a network of 28 Sun systems, running SunOS,¹ a Berkeley-derived UNIX² operating system.

The bulk of our users are the department's staff, research students and undergraduate students. The systems are also used by visiting researchers, vacation student employees and a few other people. Currently, two of our Sun servers are undergraduate student systems: an undergraduate's account is local to one of these servers. Other users' accounts are available on the entire network, via Sun's Network Information Service (NIS)³ (see [2]).

I will discuss my tools in the order in which I developed them.

2. Keeping track of temporary users

Motivation

Many of our users are with us only temporarily - for example, postgraduate students, visiting researchers, and staff on short contracts. Because I add most of the accounts I usually know when these people arrive, but I'm not often told when they leave. This meant we had unused accounts lying around taking up disk space for six months until I got around to thinking, "Hmm, I haven't seen Mary for a while - perhaps she's finished her PhD at last".

My first thought was to have a regular round-up of accounts: either ask our administrative staff to check on who's still here, or force the temporary users to fill in a renewal form each year. However, either of these would cause a lot of anguish, so I decided to let `cron(8)` take care of it.

¹ Sun and SunOS are trademarks of Sun Microsystems, Inc.

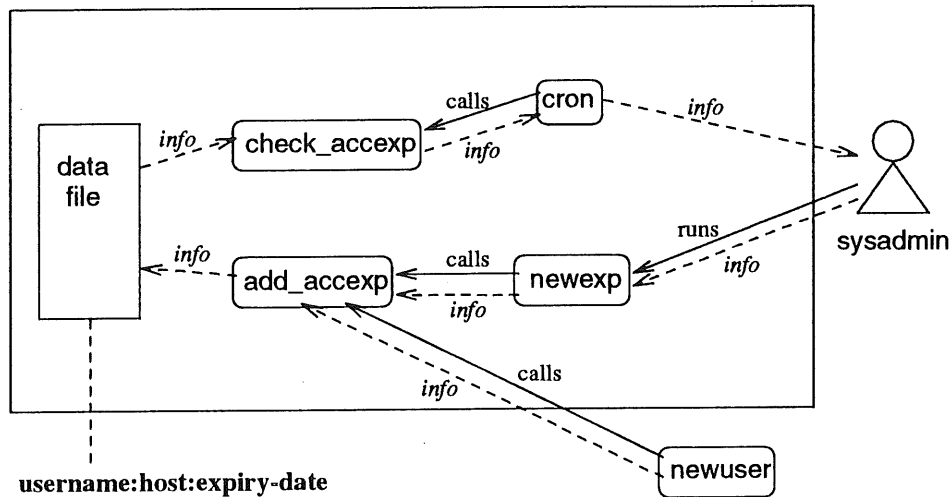
² UNIX is a registered trademark of UNIX System Laboratories, Inc.

³ Formerly called Yellow Pages (YP).

The system

I call the system "the account-expiry system", but it does not expire - that is, disable - accounts. In our environment, expiring accounts would be considered vicious and politically unwise. However, it would be easy to modify my system to do so.

What does happen is that `cron` mails me when a person is due to depart, to remind me to check whether they're still around. To keep track of people's supposed departure dates, I use a simple flat-file database.



The database is a single file, with a line for each user. There are three colon-separated fields: username, hostname and expiry date. Currently, most of our users have network-wide NIS accounts, and their hostname field is empty. The file is simple for programs to process, and is also easy to edit by hand - to change someone's expiry date, for example.

After designing the file, the next thing I did was to write a perl script, `check_accexp`, that scans the file for expired accounts. If it finds any it mails a report to the appropriate address - currently a mail alias "accountadmin" that points to me. The database is kept on our main fileserver, whose root `crontab(5)` file contains an entry to run `check_accexp` every weekday morning.

`add_accexp` is the tool that adds a new entry to the database. It's another perl script. It's easy to add a new entry with an editor; however, I wanted a software tool to automatically check the validity of the information being entered. `add_accexp` is used as follows:

```
add_accexp username [-d expiry-date] [-h hostname]
```

(If no expiry date is given, the expiry date in the database is set to empty, which means the account never expires. As mentioned above, if no hostname is given, the account is network-wide.)

`add_accexp` checks that the host and username exist, that the user does not already appear in the database, and that the expiry date is valid and has not already passed.

`add_accexp` is not particularly user-friendly; I wrote it with the intention of calling it from another program, rather than using it by itself. The other program was eventually going to be my automated user-adding tool, but that was still a dream. In the interim, I wanted an easily-used front-end for `add_accexp`.

`newexp` is another perl script. You simply type `newexp`; it prints helpful information and asks you questions. From your responses it constructs an `add_accexp` command, calls it, and if it succeeds, tells you so. (If it fails, `add_accexp` itself tells you why.)

Why perl (and what is perl, anyway)?

perl (Practical Extraction and Report Language) is a powerful scripting language written by Larry Wall. It is available free, under the GNU Public Licence.

Like shells, it is an interpreted language (but unlike shells, it checks the syntax of the script before running it).

It has the system-interaction capabilities of shells, better pattern-matching than `sed(1)`, and a superset of the capabilities of `awk(1)`, including associative arrays. It is structured rather like C, and implements many system and library calls. It has subroutines and libraries. It also incorporates string and list processing ideas from various languages, such as Lisp and Basic.

Unlike `awk` and `sed`, its flow of control is not driven by the input data; however, it does have their notion of a pattern space.

`perl` is a useful language for tasks that are difficult in shell. It saves you having to go down to a lower level and code them in C. This means easier, faster coding. I designed and wrote the entire account-expiry system in an afternoon.

`perl` was designed for manipulating text. Its constructs make processing a flat-file database, such as the account-expiry file, very easy. It is a higher-level language than C - you can work with text without worrying about how it is stored - but like C, it is a structured language, and because it was not intended to be a user interface, it does not have the syntactic peculiarities of a shell.

`perl` implements a large number of system and library functions. This is useful for systems administration tasks. For example, `add_accexp` uses the `perl` function "getpwnam"⁴ to check that the username exists. This means it doesn't have to know whether it's running in a NIS, non-NIS, or mixed environment. Also, because no forks are needed, calling the function is more efficient than grepping the password file for a local account and/or using `ypmatch(1)` to check for a NIS account.

3. Adding users

What I already had

I had an add-user shell script that had suffered at the hands of many programmers. It added an account, but left a lot to be done by hand. For example, it couldn't add the account to mail aliases or set up a disk quota for it. Instead, it echoed reminders to tell the systems administrator to do so! This was a nuisance. We use disk quotas on all our home directory filesystems, and every user is in one or more mail aliases, depending on their job (for staff) or course (for students).

To make matters worse, I had two versions of the script - one for NIS accounts and one for third year student accounts, which are local to a certain machine. This made maintenance difficult.

I had a shell script that added bulk students by calling the add-user script. This worked quite well, but its input was in a very different format to the university's class lists. I had an ugly, unmaintainable collection of last-minute shell and `perl` hacks which partly automated the conversion of the class lists. These worked under most conditions but required a good deal of human intervention. And because the add-user script didn't do mail aliases, I had to edit the mail alias file by hand to put the students in course mailing lists.

I do not enjoy doing tedious tasks that computers can do better. So I decided to automate the whole messy business.

What I wanted

I wanted a program that would add a new account automatically, taking care of local requirements. In our setup, there are many things which have defaults that depend on the kind of account, usually on its primary group. These include:

- Whether it is a NIS or local account.
- The startup files (`.cshrc` etc).
- The UID.

⁴ This calls the C library function `getpwnam(3)`.

- Mail aliases.
- The disk quota.
- The default shell.
- The expiry date.
- The home directory path.
- The `.forward` file. (See the manual page `forward(5)`.)

I wanted to be able to change the defaults easily when local requirements changed.

I wanted a single tool that could add either a NIS account or an account local to any machine. I wanted it to be non-interactive - no menus or questions. This was because I prefer that style, and because I intended to call the program from another that would add a large batch of users. A menu or question-answer front-end could be written later if necessary, perhaps to suit the preferences of other systems administrators.

Because it would be used infrequently, the tool had to provide adequate help and error messages. It had to have good error handling, particularly as I intended to call it from a batch-adding tool that would probably be left running while I went to lunch.

Existing software?

There are a few commercially available systems administration packages for Suns - less for heterogeneous networks. (Our network is currently all-Sun but could become heterogeneous.) Commercial packages tend to be menu-based and not have a batch-adding facility. They also tend to be large, expensive, and attempt to cover all areas of system administration, whereas I wanted something small, cheap, and good at just one thing.

There are various add-user programs out there, some for sale and some free. Everything I saw was rather minimal compared to what I wanted to do. Anything I bought or FTPed would have required radical modifications, so I wrote the tools myself in perl. It took me a couple of weeks,⁵ including the tool for removing a user and the manual pages.

Adding a new user

First, I wrote a perl program called `newuser`. It is typically used like this:

```
newuser -sn Jackson -gn "Janet R" -rm 5.50 -wp 2231 -ya programmers janet staff
```

This adds a user "janet", in primary group "staff" and NIS (YP) mail alias "programmers". The `gcos` field of her password entry will contain her name (Janet R JACKSON), room number (5.50) and work phone number (2231).

`newuser` does the following:

- Adds the user to the local or NIS password database. If run on the NIS master server, it defaults to NIS, otherwise to local.
- Adds the user to the local and/or NIS group databases, if you give extra groups on the command line.
- Adds the user to the local and/or NIS mail alias databases. For most primary groups it puts the user in a default mail alias (eg research students go in the "postgrads" alias). Extra aliases can be given on the command line.
- Gives the user a quota on their home filesystem, if that filesystem has quotas. There are group-dependent default quotas which can be overridden on the command line.
- Adds the user to the account-expiry system described above.
- Adds the user to the campus whois database (for staff, research students and certain other groups). This is done by mailing an update to the machine where the database is kept.

⁵ Person-weeks, not calendar weeks. I didn't do it all in one stretch.

- Sets up the user's home directory, copying in the standard startup files (.cshrc etc) for their primary group.
- For staff and students, mails an introductory message to the user. This explains how to get help, lists names of printers and mail aliases, explains the tea fund software - that sort of thing. Different groups get different messages.

If you run `newuser` with no arguments, you get a usage message. If you give it "-?" or "-help", you get the following help message:

```
Usage: /etc/netadm/bin/newuser [switches] [--] args
Switches:
  [-nomake      ]           -- If YP account, don't do YP make
  [-t          loc|yp]     -- Local or Yellow Pages account?
  [-sn         surname]    -- Surname
  [-gn         "given names ..."] -- Given Names
  [-ex         yy-mm-dd]   -- Expiry date
  [-no         studentnumber] -- Student number
  [-pw         password]   -- Encrypted password
  [-u          uid]        -- User ID number
  [-sh         shell]      -- Default shell, eg /bin/sh
  [-d          path]       -- Absolute path of home directory
  [-rm         room]       -- Room number
  [-wp         phonenumber] -- Work phone number
  [-hp         phonenumber] -- Home phone number
  [-f          emailaddress] -- Email forwarding address
  [-q          blkS blkH filS filH] -- Soft & hard limits, home filesys blocks & files
  [-ya         "mailalias ..."] -- YP mail alias to add user to
  [-la         "mailalias ..."] -- local mail alias to add user to
  [-yg         "groupname ..."] -- YP group name to add user to
  [-lg         "groupname ..."] -- local group name to add user to
  [-ng         "netgroupname ..."] -- netgroup to add user to
Args:
  username           -- (Mandatory) User name
  groupname          -- (Mandatory) Primary group name
```

(Note: netgroups are not yet implemented.)

All the switches are optional; the parameters they set will default if not given. The defaults depend on the primary group and on whether the account is NIS or local.

You can configure all `newuser`'s defaults by editing its parameters file. This is a perl file. Most of the defaults are defined in arrays that associate values with primary groups. For example:

```
# Whois is updated ONLY for users in the following groups:

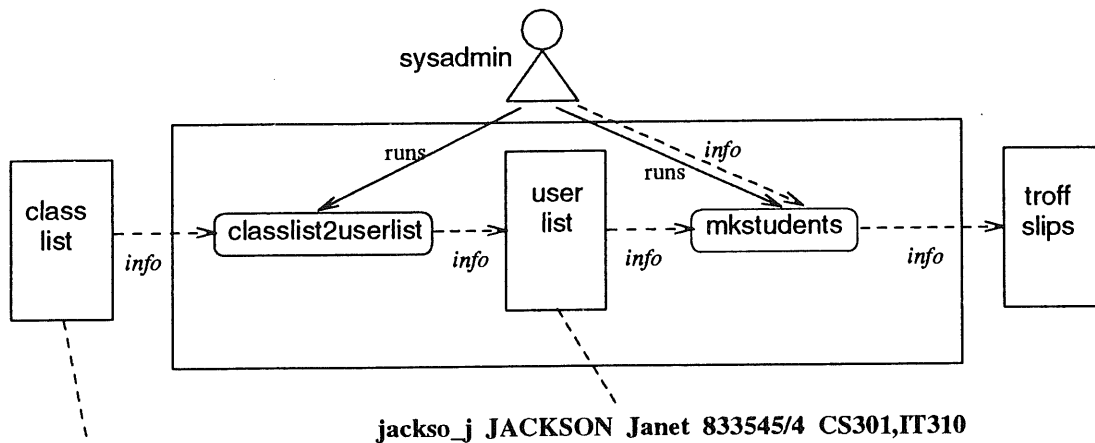
@WHOISGROUPS = ( 'staff', 'res', 'hons', 'diploma', 'cogsci', 'extern' );

and

%GROUPSHELLS = # Default shells for certain primary groups
(
  'year2' , '/usr/local/bin/zsh',
);
$DEFAULTSHELL = '/bin/csh'; # Default shell for all other groups
```

Adding a large group of users

Once I had `newuser` working, the next step was to write a tool to add several users at once by repeatedly calling `newuser`. `mkstudents` is a perl script based on the existing shell script mentioned under "What I already had" above. I then wrote `classlist2userlist`, a separate perl script that reads a class list in the format generated by the University's system, and generates a "userlist" in the format required by `mkstudents`.



```
1 230301 8335454 MS JR JACKSON ... Janet
1 231310 8335454 MS JR JACKSON ... Janet
```

(Note: the "..." in the user list represents some fields that `classlist2userlist` ignores.)

To add the 1992 third year class, the systems administrator might do the following:

```
classlist2userlist <92year3.classlist >92year3.userlist
mkstudents year3 <92year3.userlist >92year3.slips
```

The class list is keyed on student number and unit code. Each student appears once for each unit he or she is enrolled in. `classlist2userlist` generates a userlist keyed on username: for each student it generates a line containing their unique username, their gcos information (real name and student number), and a list of unit-related mail alias names. Unique usernames are obtained by using digits where necessary: if "jackso_j" was taken, it would try "jacks0_j", "jacks1_j", etc.⁶

`mkstudents` generates passwords for the accounts, creates them by calling `newuser`, and outputs a troff file of "password slips". At their first lab class each student is given a slip telling them their username and password, and how to change the password.

The systems administrator could use a pipeline:

```
classlist2userlist <92year3.classlist | mkstudents year3 | troff ...
```

but I prefer to generate the intermediate files so I can check them.

Why perl?

At first I tried to hack the shell scripts I already had, but I didn't get very far. perl turned out to be an easier solution because it integrates many useful features in the one language.

As shown above, perl's arrays and associative arrays gave me an easy way to make the programs configurable to suit site-specific requirements.

⁶ By giving a command-line switch, you can make it generate more friendly usernames. In the example above it would try "janet", "janetj", "jackson", "jrj", "jjackson" and "jrjackso". If all of those were taken it would give up and print a warning message.

perl is good at **handling text**, and user administration is based on text files: **aliases, groups, quotas, class lists**. The following example from `mkstudents` shows a few of perl's capabilities.

```
@userlist = <>;      # Slurp input into array, for using twice
...
userloop: foreach $uline (@userlist) {
    ($pword = shift @passwdlist) || die "$0: Error: Not enough passwords\n";
    chop $pword;
    ($clear_pwd, $crypt_pwd) = split( /\s+/, $pword );

    chop $uline;
    ($username, $surname, $inits, $stunum, $aliases) = split( /\t/, $uline );

    print STDERR "Adding $username\n";
```

(The “...” is not part of the code, but represents some lines I have not reproduced here.)

The first line reads all of standard input into an array, where each element contains one line of input. The loop goes through the lines. For each it obtains a clear text password and its encrypted counterpart by removing the next element from a previously generated array.⁷ This array element is a string in which the two passwords are separated by whitespace.

“chop” removes the last character (here, a newline) from a string.

The second “split” extracts the username and other fields from the input line, in which they are separated by tabs.

With its full set of **arithmetic** operators, perl is good at calculating and checking numeric information such as expiry dates and user ID ranges.

Error handling is straightforward in perl. The “die” operator in the previous example prints its argument and exits the script with an error status. There is a similar “warn” operator which does not exit. The familiar “exit” is also there, so you can return a particular exit status.

4. Removing a user

Removing a user is almost as complicated as adding one. Removing them by hand, by editing the password file and other databases and using `rm(1)`, is not difficult, but I often found myself forgetting something, such as removing the user's mail spool file, or setting up an email forwarding alias.

I decided to automate only the most frequent case - removing a single NIS user. Currently, I don't need to automate removal of a single local user. Apart from undergraduate students, the only local accounts we have are non-human, rarely removed, and not subject to the same complications as human users. Undergraduate student accounts are almost always removed in bulk at the end of the year. They are also relatively uncomplicated and as yet I have not felt the need to automate their removal.

`rmuser` is a perl program that uses the same parameters file as `newuser`. It is used as follows:

```
rmuser [-b] [-rmfiles] [-fwd addr [-ka "aliases..."]] username
```

Where appropriate, it removes the user from:

- The NIS group and aliases databases.
- The account-expiry system.
- The campus whois database.
- The NIS password database. For most kinds of user, the password entry is not actually removed, but disabled: the password is changed to “==GONE==” and the shell to `/bin/false`. This means that if the user left any files lying about the system, we can tell whose they were. The username is changed so that mail bounces and so that the name can be re-used. It is changed by prepending the

⁷ The array is generated by capturing the output of a C program.

year to it - in 1992, "janet" would become "92janet", and "jackson" would become "92jackso".

If you give "-rmfiles", `rmuser` removes the user's files. (This is not the default because I often want to leave the files in place for a while after closing the account.) It removes the user's:

- Mailbox.
- Home directory.
- Cellar directories, if any. Users who need a lot of disk space may have a second directory on a different filesystem to their home directory. We call this a Cellar. There is normally a symbolic link in the home directory pointing at the Cellar; `rmuser` finds Cellars by looking for such links.

`rmuser` will only remove mailboxes or directories that are owned by the user and start with certain paths, such as "/home/" and "/cellar/" (these are configurable, of course). It uses `rm -rf` to remove each directory, so does not check the ownership of files within that directory.

If you give "-fwd", `rmuser` adds a NIS mail alias to forward the user's mail to the address you specify, and keeps the user in any mail aliases specified with "-ka". Sometimes people want to stay in research-group mailing lists, for example.

Unless you give "-b" (batch), `rmuser` requires interactive confirmation before it removes the user, and before it removes each directory or mailbox.

`rmuser` does not check for the user in local (non-NIS) password, group or alias databases. This would mean checking every machine on our network and would only rarely be useful.

5. Discussion

I will discuss some of the good and bad aspects of the tools, roughly split into the subtopics Configurability, Ease of use, Potential improvements and extensions, and Was `perl` a good choice?

Configurability

You configure the actions of `newuser` and `rmuser` by editing their parameters file, a `perl` file. This is easy if you know `perl`, but a systems administrator unfamiliar with `perl` could easily make mistakes. It might be a good idea to use a simpler, more human-oriented file. The code to translate this into `perl` definitions wouldn't be hard to write.

`newuser` and `rmuser` are designed to be configurable, but I have not yet been able to determine whether they are configurable *enough*. In the short time I've been using them, our local conditions have not changed much. There have been a few minor changes - such as the addition of a new primary group with its own startup files - which the `newuser/rmuser` system handled easily.

If the format of the University's class lists changes, my program `classlist2userlist` will have to be changed to match, since the format is hardcoded in. It would be possible to make the program obtain the format from a human-oriented schema.

I wrote the tools discussed in this paper to solve the particular problems we have at UWA Computer Science. I did not try to design them so that other sites could configure them to suit their needs; however, they may well be useful to other sites. I have sent the code to a few people who asked to see it, but none of them have yet made any comments.

Ease of use

I have achieved my main aim - to make certain aspects of user administration easier by automating them. Adding or removing a user is now done in one step instead of several. With the account expiry system, our network is now kept free of old, unused accounts. Adding this year's students with `classlist2userlist` and `mkstudents` was a simple job compared to the convoluted process I had to use last year.

To add a user, I normally have to give a lot of command-line parameters to `newuser`, and I can't usually remember them all. I often find myself running it without arguments first to see the usage message.

`newuser` is not particularly fast. Before it does anything, it checks that all the parameters you gave are sensible (for example, you can't add a username that already exists) and determines defaults for all the rest.

This takes a few seconds, during which it produces no output. If the parameters are OK, it then adds the user. This takes even longer, but I perceive it as faster because it churns out lots of output telling you what it's doing. I think the overall speed is acceptable for the small amount of use the program gets.

`newuser` is rather garrulous. A "quiet mode" would be nice, particularly when batch-adding with `mkstudents`. It would be easy to add: in perl one can write

```
print STDERR "Don't panic - I'm still here\n" unless $quiet;
```

`rmuser` could be dangerous if used carelessly - especially if the "-rmfiles" switch is given. Therefore it does a lot of checking before it touches anything, and unlike the other tools, defaults to interactive operation.

My favourite feature of `rmuser` is its mail alias handling. This has saved a lot of hassle and bounced mail.

`mkstudents` is inefficient because it calls `newuser`, which can add only one user at a time. This means, for example, that the mail alias file is edited separately for each student, rather than doing one big edit. However, since it works well and is only used a couple of times a year, I am not inclined to make the radical modifications `newuser` would need to let it add several users at once.

Potential improvements and extensions

The account-expiry system could be modified to actually expire accounts, a feature I do not currently want. What would be more useful to me is to have it automatically mail the expired user as well as the systems administrator. The hardest part of this would be to design a form letter that would allow for special circumstances and not offend anyone!

There are a few useful things that could be added to `newuser`:

- It could record extra information about the user. For example: for "guest" accounts, I like to keep track of why they have an account and which staff member asked for it. This could perhaps be incorporated in the account-expiry database.
- It does not (yet) handle password shadowing or aging.
- As previously mentioned, it could have a human-oriented parameters file instead of the existing perl one.
- A menu or question-answer front end would be helpful, particularly when I'm not around.

`rmuser` could be extended to handle local accounts - then a "rmstudents" could call it to remove bulk students.

A useful extension to `classlist2userlist` would be to allow the systems administrator to specify different password generation schemes.

Was perl a good choice?

Having written the tools in perl rather than the more usual shell, awk etc, I can think about which aspects of perl helped and which hindered.

Calling external programs, which `newuser` and `rmuser` do many times, is messier in perl than shell. For example:

```
system( "rm -r $dir" ) == 0 || print STDERR "rm $dir failed\n";
```

in perl, but in Bourne shell:

```
rm -r $dir || echo "rm $dir failed" 1>&2
```

For more complex commands, particularly sed commands, the perl is even worse because to run the command, perl calls a shell anyway. Getting the argument quoting right can be tricky because of the extra level of processing done by perl.

The actions of many of the external programs could be done in perl itself, but I see no reason to write such code when the external programs work perfectly.

Since I don't have shell versions of my tools, I can't measure whether perl or shell is faster at this kind of work. However, perl's built-in arithmetic capabilities make it faster at calculations than shells, which have to call `expr(1)` or whatever. The same goes for other operations, such as pattern-matching, that are built in to perl. Shell versions of the tools would have to spawn many more processes.

6. Conclusion

When I decided to automate aspects of user administration, local requirements - such as differing kinds of users, large classes of students, and NIS - led me to write in-house, site-specific tools in perl. perl's features - such as block structure, arrays, text handling, system calls and error handling - suited the job and made programming easier.

I developed the following tools:

- The "account-expiry" system, which tracks temporary users with the help of `cron`. Rather than expiring accounts, it mails reminders to the systems administrator.
- `newuser`, which adds a user with almost everything automated.
- `rmuser`, which removes a user with almost everything automated.
- A pair of programs which automatically generate bulk student accounts from class lists.

I find the tools very useful. They work well, although there is some room for improvement. I think my use of perl for this has paid off.

References

- [1] Larry Wall and Randal L Schwartz, *Programming perl*, O'Reilly & Associates, Inc., 1991.
- [2] *SunOS Reference Manual*, Sun Microsystems, Inc., 1990

 Graham K Jenkins
 main!gkj@yarra.pyramid.com.au

Ipec Metro Distribution, 296 Whitehorse Road, Nunawading 3131.

A DISASTER IS WHEN ..

Many of Australia's most successful companies now use transaction processing packages written in Oracle, Ingress, Informix, Sybase and their equivalents, and these are most often executed under some form of a Unix operating system. In such an environment, a disaster is when ..

- (1) The cleaner pulls the plug on the mainframe so he can make his Electrolux reach under the communications controller.
- (2) The Manager takes his son to work whilst he is doing an end-of-month update in the Accounting System. The son decides to help.
- (3) A disgruntled employee leaves to establish own business, taking with him a list of current customers.
- (4) The main processor fails at 8 am on a Monday, leaving 100 customers waiting for taxis - and 100 drivers wondering where they should be.
- (5) A disk falls over at 5 pm on a Friday; it contained records of all transactions done on that day.
- (6) The air-conditioner suffers an internal hemorrhage at 3 am and starts filling the computer room with water.
- (7) The General Manager is requested to allocate funds so that a disaster mitigation scheme can be implemented.

The reader is encouraged to extend this list in relation to his own experience and environment. I would even suggest the "A Disaster is When .." game as a potentially fruitful source of ideas for those at computer society meetings. In the paragraphs which follow, I would like consider some of the disasters in my list, and develop some strategies for mitigating the chances and/or the consequences of their occurrence.

(1) .. THE CLEANER PULLS THE PLUG ON THE MAINFRAME

It's not always the cleaner. It may be the telephone technician installing your new ISDN link for increased data communications capacity and reliability. Or it could be the air-conditioner service person performing scheduled maintenance. Murphy's Law decrees that such people always arrive when the Systems Manager is out, and persuade the receptionist that what they have to do in the Computer Room "won't take long."!

I can not stress too forcibly the importance of keeping your Computer Room locked, and enforcing a strict "No Entry" policy. PHYSICAL SECURITY is important for every link in the chain which delivers computer service to the end user. Emergency power supplies, telecommunications and power distribution rooms and remote site data communications equipment are some of the essential links in that chain.

Your very best endeavours won't prevent someone digging up the data communications cable outside your premises. Your applications software should be designed to cope with such contingencies. The better relational database systems now incorporate journal and rollback facilities, and provide two-phase commit capabilities. You can also use screen prompts like: "Please ensure that all banking transactions have been printed satisfactorily before proceeding!" in your applications software.

(2) .. THE MANAGER TAKES HIS SON TO WORK

It doesn't have to happen at work. The manager may be rushing through his end-of-month updates from home through a modem, or even doing it in an aeroplane on his lap-top. In any event, the issue here is ACCESS SECURITY.

The list of prophylactic measures will be well-known to most readers. Ensure that passwords are kept secure, and that no user's password provides access to more than is necessary; use password aging to enforce changes every few weeks, and to disallow two changes within one week; ensure that user's are properly educated concerning password security, logging out of idle displays, etc. How many of you can honestly tick every one of these?

The hackers in our community have come up with some new games lately, like supplying free software to interested users (who subsequently find viruses at their installations), or using "uucp" or its equivalents to obtain copies of password files (which are then fed to "guess, encrypt and compare" programs). Be wary of these.

I don't have any magic formula for recovery after access security has been breached. A good set of dump tapes is the best suggestion I can offer. Whilst virus removal programs can be useful, there is a distinct possibility of the cure being worse than the disease.

(3) .. A DISGRUNTLED EMPLOYEE LEAVES TO ESTABLISH HIS OWN BUSINESS

The issue here is INFORMATION SECURITY. A System Manager's responsibility in this regard is to ensure that each employee has access only to the information which he or she needs.

One of the characteristics of a successful Systems Manager is that he or she has very broad shoulders - because that is where the General Manager is going to put the blame when something like this happens.

You may be able to ease the burden by denying print, screen capture and data transmit access to employees whose jobs do not require such access. You might also consider monitoring employee usage patterns, something like what the banks do with your credit card transactions. The printer log is a useful reference in that regard. With a bit of effort, you may be also able to train your Personnel Manager to alert you immediately concerning any variation in employee status.

(4) .. THE MAIN PROCESSOR FAILS AT 8AM ON A MONDAY

The processor may be going again by noon - or you may have to wait a couple of days for parts to arrive from overseas. In any event, the critical issue here is OPERATIONS CONTINUITY. One way of achieving this (at least in part), is through some form of easily accessed transaction log. Ideally, such a transaction log would be located at backup site, equipped with sufficient resources to enable operations to be maintained at a reduced level. A backup site might be owned by your company, or shared with other companies.

--
The resources which should be maintained at the backup site, and their degree of availability, will depend on the size of your company and the nature of its business. Thus a taxi company with its main operations centre in the city and a branch office near the airport might have a logging printer at the branch office for recording every booking transaction as it is taken. It could also have an arrangement for exchange-based telephone call diversion to the branch office in the event of catastrophe.

These backup resources would be sufficient to enable a small taxi company to continue operating, with bookings being taken on a manual basis. A larger company would need to think about additional telephone and data lines at its backup site, with accommodation for additional people and for a computer.

(5) .. A DISK FALLS OVER AT 5PM ON A FRIDAY

If the disk concerned is part of a RAID group, the Systems Manager may be the only one who knows there is a problem. Some mechanism (e.g. paging) should be in place so that he can be alerted to it immediately.

Some manufacturers offer mirror disks as an alternative to RAID, claiming faster access times and/or ease of archiving through disabling of one mirror piece. Again, if your live data is mirrored, your users and customers may remain blissfully unaware that anything is wrong.

Data which does not change throughout a normal working day may not be kept on a disk which is part of a RAID or mirrored group. Such data can be recovered from archive tapes. Traditional practice has been to take full system dumps at periodic (e.g. monthly) intervals, with incremental dumps each night of those files which have changed since the previous full dump. More recent developments in DAT and similar devices with integral compression capabilities have made it feasible to take full unattended dumps each night.

Whatever methodology is employed for archive purposes, it is essential that all media be re-read immediately for verification, then transported in a controlled-environment vehicle to secure off-site storage. In this age of "lights-out" operations, you may wish to consider data communications as a potentially faster means of moving your archive information off-site.

(6) .. THE AIR-CONDITIONER SUFFERS AN INTERNAL HEMORRHAGE AT 3 AM

Even the most modern "office environment" machines do not take kindly to water flowing around their under-floor cables. And tenants on floors below yours can become positively nasty when they find water all over their carpets. It is therefore desirable that when something like this happens, immediate shutdown action is initiated, and some responsible person is notified.

At Ipec Metro's installation, an environment monitor continuously checks temperature, humidity and underfloor surface conductivity against pre-determined limits. As soon as any limit is exceeded, the breakers feeding the installation's un-interruptible power source are tripped.

A monitor program is invoked at 5-minute intervals to check whether power is still being fed to the UPS. It does this by checking that the DTR signal is present (i.e. a getty, login or shell process is running) on at least one of two terminals whose power is fed from the UPS input. In the event that power loss (through environment alarm or mains failure) is sustained for more than 10 minutes, all users are warned, the Systems Manager is paged, and shutdown is initiated.

--

The same monitor program also checks the number of "bad" logins which occur in each 5-minute interval, and transmits an appropriate pager message to the Systems Manager when a pre-determined limit is exceeded. This might signify an attempted security breach, a data line failure, or bad case of Mondayitis for an operator.

(7) .. THE GENERAL MANAGER IS REQUESTED TO ALLOCATE FUNDS

General Managers tend to classify disaster mitigation initially as a "good thing", more or less in the same category as motherhood and apple pie. When they learn what it may cost, it usually gets moved to the "why bother?" category, along with contraception and church-on-Sunday.

As a Systems Manager, you should not let the matter rest there. Because, according to your analogy, you will be the one left holding the baby (or knocking on the pearly gates) when disaster strikes.

The terms "downsizing" and "lights-out" seem to have been in vogue recently, and they may be what you need to tilt the scales your way. With current trends in system design, a small machine may well have a greater capability than the one you now have. It may not need an air-conditioner, and it may offer savings in both media and operator costs through its use of compact high-capacity storage mechanisms. It may also offer savings through provision of Ethernet and/or ISDN in place of point-to-point local wiring and data communications arrangements.

Make sure you have the company Accountant with you when you speak with the General Manager, so that he can juggle the lease and depreciation costs to produce the desired result. He will also be able to help you answer the "How much would it cost in lost business?" type of question.

If all else fails, make sure that the General Manager has a full understanding of the possible consequences of disaster. Then, when disaster does strike, you MAY be able to hand the baby straight back to him!

An Update on UNIX-Related Standards Activities

Stephen R. Walli

Report Editor, USENIX Standards Watchdog Committee

[ENOBANDWIDTH]—No Band Width Left

An asynchronous signal was caught (such as SIGBALLOT, SIGPLSCOMMENT or SIGMORE2RD; See the description of <signal.h> in 3.3.1) while the programmer process was busy dealing with the last such signal. The current handler shall be interrupted after attempting to save context, and the new handler begins to execute. Signals shall continue to be delivered, interrupting one another, until {_POSIX_LAST_STRAW} is reached. Once the true value of {_POSIX_LAST_STRAW} is reached, the entire precarious stack comes clattering down. The last interrupt handler with enough context to still maintain some semblance of integrity returns [ENOBANDWIDTH]. A programmer process may attempt to notify its parent process of the condition in an implementation defined way, but it shall be ignored.

All joking aside, POSIX is finally beginning to strain the seams of the working groups, along with the standards process in which it lives.

There was a time when POSIX meant a working group of around 20 people defining a system interface for source code portability, based on the C language interface to UNIX (356 pages). Then it started to grow.

The shell and utilities were added (~ 900 + ~ 300 pages) making POSIX.2. Someone thought conformance testing should be done in a standard way, and since they started suspecting POSIX was going to grow some more, they decided to make a standard for defining test methods (47 pages). Then there were the test methods themselves. (Another 500 pages each for POSIX.1 and POSIX.2 so far.) Then came real-time, to take all the real-time sorts of things that POSIX.1 couldn't settle on, along with a few things of interest only to the real-time community. (Another ~ 400 pages, plus threads.)

You can continue in this vein for quite some time. The POSIX.20 (Ada Bindings for POSIX.4/.4a Real-time services) project has recently started

up. This doesn't take into account the other related IEEE Technical Committee on Operating Systems standards (1201 GUIs, P1224 X.400, P1238 FTAM).

The working groups now number between 350 and 400 people, who attend four one week long meetings a year. (As an aside, do the back of the envelope calculation as to the amount of money that represents. Using 350 people, \$2000/week expenses, and a loaded staff cost of another \$1500/week, I get \$4,900,000/year.) The working groups generate a considerable quantity of reading material, documenting discussions and decisions, reviewing and coordinating with related efforts, proposing alternatives, and commenting on other proposals. This all on the way to producing the actual draft document to ballot as a standard.

An average C programmer or applications designer would likely be interested in the base interfaces (POSIX.1), most of the real-time services (POSIX.4), some security (POSIX.6), and possibly some of the networking (Transparent File Access — POSIX.8, and Protocol Independent Communications — POSIX.12). And of course the shell (POSIX.2/.2a). Whew! That's a lot of paper. POSIX Zen koan: Does a tree scream in the wilderness every time a new project request is approved?

Before all the Ada and Fortran programmers start howling about the "C" programmer reference, let me explain. Until recently, the Ada or Fortran involvement in POSIX has been two small working groups who realized very early in the game that this standard was going to be extremely important. They have each now produced a standard binding of their respective languages to the C-based POSIX.1 standard.

The Ada working group is now beginning to address POSIX.4 and POSIX.4a. The Fortran working group is beginning a Fortran 90 version of POSIX.1 (the original POSIX.9 standard being an F77 binding). These groups are well placed to begin opening the way for a large segment of the

systems building community that doesn't work in C to gain access to the benefits of a standard systems interface. They are, however, the exceptions to date, rather than the rule.

With several vendors delivering POSIX.1 interfaces and POSIX.2 shells on their non-UNIX-derived operating systems, the growth of interest in POSIX based open systems is only just beginning. This could bring in a badly needed injection of new blood to the POSIX working groups. But these people will not have the often needed historical context. They will require some time in the system to come up to speed on all of its perversities.

Let's argue that you do care about all of these related base standards. (You should! They are being forced upon you in many cases.) That means you should be involved enough to at least ballot. Maybe not every section of every document, but certainly sections in each of these drafts. You cannot stand by idly trusting that a good standard will fall out of the end of the process, and you can then use it. If you are a member of a technical user group such as USENIX, then the only difference between you and the people building the POSIX standards are that they've found the financial support of their organizations to attend the meetings. Think about it.

If you cannot attend the meetings, then you should ballot the draft documents. (Ballot early! Ballot often!) But this then brings us back to

[ENOBANDWDTH]. The balloting schedule for 1992 (as taken from a balloting status report of last December) looks something like the table at the bottom of the page.

A few of these are a little bit specialized, such as POSIX.10 (Supercomputing Profile), POSIX.13 (Real-time Profiles), POSIX.15 (Supercomputing Batch Interfaces), and POSIX.17 (Directory Services). Note, however, that these are the smaller documents. And I haven't mentioned the non-POSIX TCOS standards from P1201 (GUI), P1224 (X.400), and P1238 (FTAM) that are also sending drafts to ballot in 1992.

After a punishing year of balloting in 1991, the Balloting Vice-Chair is staging the drafts so as to not completely bury the balloting groups in which there is often a lot of overlap. This means a working group had better hit its target date, or it will be left circling the balloting schedule, looking for another time slot to hit.

This heavy balloting load is starting to lead to the break down of the mock balloting process. In the "good ol' days" a document could be sent to mock ballot to test the waters, before committing to the serious work of a formal ballot. With so many documents out for formal ballot, most mock ballots are finding themselves at the bottom of the pile. [ENOBANDWDTH]

The heavy balloting of large documents to large balloting groups is beginning to take its toll

Project	Ballot	Date	Size
POSIX.0 (Guide)	1st Ballot	July 92	~250 pgs
POSIX.1a (More .1)	1st Ballot	Summer 92	~100 pgs
POSIX.1/LIS	1st Ballot	Spring 92	~400 pgs
POSIX.2a(UPE)	3rd Recirc	Jan 92	~300 pgs
POSIX.3.2 (.2 Test methods)	1st Ballot	Spring 92	~500 pgs
POSIX.4 (Real-time)	3rd Recirc	Spring 92	~320 pgs
POSIX.4a (Threads)	1st Recirc	Winter 92	~200 pgs
POSIX.4b (More Real-Time)	1st Ballot	Fall 92	tbd
POSIX.5 (Ada)	Final	Spring 92	~300 pgs
POSIX.10 (Super. AEP)	1st Ballot	July 92	~75 pgs
POSIX.12 (Protocol Indep.)	1st Ballot	Fall 92	~400 pgs
POSIX.13 (Real-time AEP)	1st Ballot	Spring 92	~80 pgs
POSIX.15 (Batch)	1st Ballot	July 92	~175 pgs
POSIX.16 (C Binding)	1st Ballot	Spring 92	~200 pgs
POSIX.17 (Direct. Serv.)	1st Ballot	Spring 92	tbd
POSIX.18 (PEP)	1st Ballot	Spring 92	tbd

in other ways. The IEEE Standards Office can no longer absorb the entire cost of copying and distribution. For the first time in POSIX's history, the current invitation to ballot is charging a fee per document (\$25) to join the group. [ENOFINANCES]?

We've really only touched on the balloting load. If you are serious, and you find the financing, you can look forward to spending four weeks a year in exotic locations like Parsippany, New Jersey, at the working group meetings. If you stick strictly to a working group or two, you will actually be able to get out of the hotel for dinner. (Parsippany didn't have a lot to offer here, but then we have been to New Orleans.)

If you are interested in any of the cross group issues, such as Language Independent Specifications, Conformance Testing, or most important of all, the overall structure of the POSIX standards, and the process of building it, you can look forward to six 12 to 13 hour days (not five).

If you are a serious participant, i.e., you actually do work between meetings, you probably spend at least another week of your time on POSIX for every week of meeting time. Unless you are self-employed, it is unlikely you do this in plain sight at your place of business. It's generally frowned upon. After all, your manager already feels he's paying for you to attend four of these "conferences" a year. [ENOSUPPRT]?

It is often surprising to discover how little corporate support exists for people in the process. There are many stories of people who cannot find the consistent support to participate, either moral or financial, who work for companies you would assume have an obvious stake in POSIX. While their companies love to bask in the glow of their "obvious commitment to open systems," their people are put through the wringer trying to get the job done right. (Please see the last line of the ENOBANDWDTE definition again.)

So the coming year will be interesting. As many of the POSIX projects reach ballot, the number of documents circulating in ballot grows. The relentless and blind pressure from the marketplace to build more standards faster will grow rather than fade. The number of POSIX projects will continue to grow, before we've even figured

out how the current ones integrate. Hark — I hear a tree screaming.

Report on POSIX.6: Security

[srw—Please note that Dan's report on POSIX.6 covers the October 1991, meeting in Parsippany. Timing and the Holidays were against me getting it published in the last issue with the other October reports. Apologies to both Dan and the reader.]

Daniel D. Ujihara
<dujihara@EBay.Sun.COM> reports on the October 21–25 meeting in Parsippany, NJ:

The October meeting was a transitional one for POSIX.6. After three and a half years in the making, POSIX.6 went out for ballot. Draft 12 (dated September 1991) was sent to the IEEE for balloting before the October meeting.

Since the document is out for ballot, the functional subgroups have disbanded and reformed into three new subgroups:

- POSIX.6 futures
- technical reviewers for the ballot
- test assertions writers

POSIX.6 Futures

The futures subgroup spent much of the meeting investigating work items for POSIX.6a. They determined that there is interest in the following areas:

- access controls (integrity, commercial)
- data interchange formats
- general terminal interfaces
- identification and authentication
- test assertions

The following areas were rejected:

- accountability and audit—they are addressed in the current draft
- accuracy (i.e. the health of your hardware) — the implementor should do this
- availability
- language independence specifications
- temporary access control

The subgroup has also begun tracking other active POSIX groups in which it is interested (i.e., POSIX.0, POSIX.4, POSIX.7 and POSIX.8).

Test Assertions

There was insufficient critical mass for the test assertions group at this meeting, thus no progress was made. POSIX.3 provided training on writing test assertions to POSIX.6, in an effort to bootstrap the process. While this tutorial was well received, it revealed that the POSIX.3 test assertion methodology may have to evolve some to accommodate the multi-option nature of POSIX.6.

Technical Reviewers

The technical reviewers spent the October meeting defining the procedures and processes for ballot resolution. Mike Ressler of Bellcore is the technical editor and ballot coordinator for POSIX.6 and he has put together a group of twelve technical reviewers. The reviewers are responsible for addressing ballot objections and coordinating responses between themselves. The ballot coordinator is responsible for making sure each objection is addressed.

Future meetings will be dedicated to the resolution of any ballot objections, a new project request (PAR) for POSIX.6a, and test assertions for POSIX.6.

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

Bill O. Gallmeister <bog@lynx.com> reports on the January 13-17 meeting in Irvine, CA:

Summary

This meeting, we concentrated on the real-time profiles document (POSIX.13), and on the new real-time interfaces (POSIX.4b). POSIX.13 was released to ballot at the end of this meeting. In addition, technical reviewers for POSIX.4 and POSIX.4a were able to make good progress on their respective drafts. POSIX.4 may complete balloting in the Fall of 1992; POSIX.4a will probably take another six months after that. Our current intention is to ballot POSIX.4b after the October meeting. By that time, POSIX.4 itself should be through the IEEE balloting process and into an ISO ballot, though we will probably still be balloting POSIX.4a.

POSIX.13: Real-Time Profiles

The big news here is that POSIX.13 is now ready to ballot! After a number of small mock ballots, the working group closed on what the functionality should be in each of the four profiles contained in POSIX.13. POSIX.13 is the first POSIX profile document to reach ballot. With any luck, we will begin POSIX.13 ballot resolution at the April, 1992 meeting.

POSIX.4b: New Real-Time Interfaces

Our goal this week was to settle on the contents of POSIX.4b and to produce first drafts of each chapter. We came close. Some first chapter drafts should be coming in the mailings.

POSIX.4b includes chapters for: interrupt control; timeouts on all blocking functions; enhanced memory allocation; sporadic server scheduling; spawn (a combined fork/exec); the ability to wait on multiple things, also known as event flags; CPU time budgeting (the ability to set time budgets for other processes/threads and examine time spent by those processes/threads); and more interfaces for driver/application communication (i.e., ioctl). Some of these chapters actually did make it into the draft by Friday. Other facilities were brand new, and are still being debated (CPU time budgeting, event flags and ioctl were especially contentious). The contents are not cast in stone, but it indicates the small groups which the working group will form at the next meeting.

Ioctl

The standard method of driver/application communication, aside from the obvious read/write, is ioctl. POSIX.1 elected not to standardize ioctl, instead preferring to provide alternate standard interfaces to control those devices, such as TTYs, for which a standard set of ioctls existed. For devices which were not standard, POSIX.1 did not wish to provide a standard interface for doing non-standard things.

There was some agreement with this approach in POSIX.4, but the idea had been raised that perhaps there were a suitable set of standard devices which might benefit from standard ioctls (SCSI devices, IEEE 488 devices, and so forth).

A small group (me) gathered up a list of standard devices for consideration for a standard

ized control interface. Whether the interface was `ioctl` or not is, at this point, largely immaterial. When this list was presented, it was rather short; it turned out that most of the impetus for `ioctl` is not the desire to perform standard I/O operations (rewind a tape, eject a floppy), but rather to do non-standard things (rotate a radar, drop some control rods, e.g.). We decided to form a small group to further explore this issue.

Enhanced Memory Allocation

An interface by which memory can be allocated from various special pools (NVRAM, Video Ram, for instance) is standard practice in much of the real-time community. This is the idea behind an enhanced memory allocation interface. It was pointed out that the `mmap()` facility could be used to carve off parts of an appropriate memory area, and an allocator could be easily constructed at application level based on this. This was seen to be the correct way of allocating special memory.

Asynchronous Interfaces

Some have requested that POSIX.4 provide asynchronous interfaces to blocking services. Some objected that we have already done so; POSIX.4a provides an asynchronous interface to any system interface by allowing threads to be created for the purpose of using those interfaces without suspending the original thread. For instance, an asynchronous open routine can easily be emulated by creating a separate thread to perform a synchronous open, then signal the original thread and go away. Despite this, we have an agenda bullet to explore this possibility in Dallas this April.

POSIX.4 and POSIX.4a Draft Status

POSIX.4 has completed its third round of balloting and the ballot objections are dropping off in a gratifying fashion. The draft should be out for its next-to-last recirculation by the April meeting. IEEE approval is expected in the Fall of 1992. POSIX.4a is approaching its next circulation and should be recirculated by the Fall meeting as well. POSIX.4a will actually be reballoted rather than recirculated, because it was not able to achieve a 75% approval rate. This means that the entire draft will be open for comment, rather than just

the parts of the draft that have changed from the previous draft. The balloting group, however, will remain the same.

Report on POSIX.5, POSIX.20: Ada Bindings

Del Swanson <dswanson@email.sp.unisys.com> reports on the January 13-17 meeting in Irvine, CA:

The POSIX.5 working group is producing an Ada binding to the C-based POSIX.1 document. The group is also involved with a new project, POSIX.20. This will be an Ada binding to the programming language independent specification of POSIX.4 (Real-time Extensions) and POSIX.4a (Threads).

The meeting in Irvine was spent on the finishing touches to the POSIX.5 document, which is finishing ballot, and getting a good start on the new POSIX.20 work.

The group had been somewhat shocked at the overwhelming positive response to the first recirculation of the POSIX.5 document. There was almost 90% approval by the almost 300 balloters. This is a compliment to the great ballot resolution job on the part of the technical reviewers. It was a slight embarrassment, however, since there were known errors in the draft. In fact, the draft had been sent out for recirculation even though an error had been discovered at the last minute. We assumed that it could be fixed in the next recirculation.

The group decided to "do the right thing", and make one more recirculation with minor editorial changes and a couple of error corrections. Draft 8 is expected to be the final recirculation. It should have a fast turn-around, and it is expected to be approved at the June meeting of the IEEE Standards Board.

Several new members have joined the group because of their interest in the POSIX.20 binding of the real-time extensions. They showed themselves to be eager participants, astute observers, and acquainted with the issues we will be facing in this task.

The single issue that will demand our attention more than any other during the binding deliberations is how the relationship of Ada tasks to

POSIX.4a threads will be represented in the binding. As background, most of us are convinced that the p-threads as defined in POSIX.4a will provide an adequate capability for Ada implementations to use for a direct mapping of Ada tasks. Further, we believe that the services provided in the POSIX.4 materials will be sufficient for a reasonably straightforward implementation of Ada tasking semantics by Ada runtime libraries.

Nevertheless, the semantics of tasks are not identical to the semantics of threads (for example, Ada allows no orphans). Further, since Ada as a language introduces concurrency along with the rules of interactions, the runtime libraries of Ada implementations like to be able to make assumptions about the state of tasks. Disrupting this state would likely be disastrous. For example, if an application directly aborted a thread (which had been mapped as a task) the runtime library could become quickly confused.

Opinions about how to deal with this situation range widely, and tend to be held strongly. Some participants believe that services such as *pthread_create()* that are available via language construct (declare a task) should not be visible to applications by direct call to the operating system. Others believe that every interface should be made directly visible by way of the Ada binding. Many attempts are being made to integrate the needs of the various factions. This issue will arise in many ways throughout the entire POSIX.20 project.

One issue that was resolved was the basic character of the binding. The IEEE POSIX working groups have taken the lead from ISO WG15 (ISO POSIX) that all POSIX standards will be formulated as language independent specifications (LIS), and this will be accompanied by a series of thin programming language bindings. While there continues to be skepticism evidenced by some members of the group, who doubt the wisdom (or the possibility) of this approach, we agreed that we would be converts to the cause.

We will produce a thin binding to the LIS versions of POSIX.4 and POSIX.4a. We will do so enthusiastically. We will help the POSIX.4 working group insofar as we can to produce a viable LIS version.

We are fortunate that a couple of our members have received funding to give us a head start

on some labor-intensive activities. The U.S. Navy funded a solid first draft of the test assertions for POSIX.5. The U.S. Army funded a first draft thin Ada binding (or transliteration) of the existing C-based POSIX.4 document.

In summary, we are expecting some significant progress at our next meeting, now that most people are oriented to the issues. Several people have volunteered to write position papers, and to start working on chapters to present for consideration in April.

Report on P1224: X.400 API

Steve Trus <trus@duke.ncsl.nist.gov> reports on the January 13-17 meeting in Irvine, CA:

Summary

The Irvine meeting was a productive one for the P1224 API working group, and we are well under way to completing an IEEE standard.

We worked on and discussed:

- A recommendation for the International Standardization of the IEEE Networking APIs,
- IEEE balloting status for P1224,
- IEEE balloting plans for P1224.1,
- The test methods sections for the P1224 and P1224.1 documents.

International Standardization Recommendation

I presented a paper entitled "Recommendation for International Standardization of IEEE Networking APIs" to the P1224 working group and the Distributed Services steering committee. This paper recommends the international standardization of the IEEE P1224 (X.400 API), POSIX.17 (Directory Services API), and P1238 (FTAM APIs) within jtct/22/WG15 (ISO POSIX). The Chairman of the Distributed Services steering committee will forward this recommendation to S1 (Test and Office Systems), SC21 (OSI), and SC22 (Programming Languages, which includes the ISO POSIX working group).

P1224 Balloting Status

P1224 is the document defining a network object management API, to be used by P1224.1

(X.400) and POSIX.17 (Directory Services). Balloting of the P1224 document began on January 1, 1992, and it will have ended on January 31, 1992. The balloting pool consists of 73 members. As of the meeting only 5 responses (all yes) had been received. The members of the group agreed to call the members of the balloting pool to ensure that we receive the minimum of 75% responses required by the IEEE for a ballot to close.

The test methods for P1224 will be included in the first recirculation of the P1224 document. Balloting cannot complete until the test methods are balloted.

P1224.1 Balloting Plans

Invitations to join the IEEE P1224.1 Balloting group will be sent out immediately after this meeting to:

- the general TCOS balloting pool,
- members of the X.400 API Association,
- members of X/Open,
- members of the Electronic Mail Association.

The P1224.1 balloting period will begin April 15, 1992 and it will end May 15, 1992. The test methods will be included in the initial ballot of the P1224.1 document. Iain Devine, the P1224 technical editor, will be the ballot resolution reviewer, assisted in technical matters by members of the X.400 API Association and X/Open. The group planned the mechanics of the first phase of balloting.

Test Methods Review

X/Open funded a contractor to develop the test methods for the P1224 and P1224.1 documents. This work was completed before our meeting.

The P1224.1 test methods were distributed to the members of the group at the meeting. The group spent much of the week reviewing these methods, and numerous changes were made. When the changes have been incorporated into the P1224.1 document, it will be sent to the IEEE for balloting.

The P1224 test methods were also distributed to the members of the P1224, as well as POSIX.17. The groups spent part of an afternoon reviewing them, assisted by a test methods consultant from

POSIX.3 (POSIX Test Methods). When these changes, along with the ballot objections and comments, have been incorporated into the P1224 document, it will be sent to the IEEE for recirculation.

Report on TAGs

John Hill <hill@prc.unisys.com> reports on U.S. Standards Technical Advisory Groups:

Standards, like automobiles, come in several sizes, shapes, and colors suitable to the use to which they will be put. With cars there are options for just about everything from appearance, to utility, to emissions control. Not only that, cars are also sold with features specifically accommodating the place in which they are to be used. If you don't have a strong grasp of what that means then try driving your '61 Ford Fairlane down any London street during crush hour. Your car won't fit, it's too wide. Its outside mirrors aren't a breakaway type, so you are dangerous to pedestrians. Not to mention that you steer on the wrong side of the car for the side of the street on which you are driving. In London cars have features to accommodate their situation, just like your '61 Fairlane fits local US requirements in the early sixties.

What does this have to do with standards? A lot. The analogy helps highlight some of the problems of developing worldwide standards. Driving and cars provide interesting examples. It would be helpful if it were simpler to drive in foreign places without being a hazard to the local inhabitants, as well as to oneself. What's more, it would be convenient to have the same situation, in terms of driving conditions and cars, presented all over the world. But in the UK there are unique local requirements for cars, just as there are in all other nations of the world.

Consider the situation. In countries the world over, the local people know their driving conditions better than anyone and from them can determine the requirements for a suitable car. After all, they face the situation every day. But conditions differ from country to country. This places different requirements on the cars to be operated in each country. From a producer perspective, this results in different product variants to accommodate each country in which the product is

used. From a world user perspective, this results in inconvenience during use. (It does, however, simplify the matter of determining where you are. All you have to do is look at the car you're driving.) From a general interest perspective, it is confusing at best and baffling at its worst.

Information technology exhibits many of the same properties as cars. In the case of telecommunications, there truly are a multitude of factors which take on local values. One trivial example is the assignment of characters to numbers on the telephone keypad. A tougher problem is the variability in the quality of phone line service across the broad spectrum of transmission rates. Please recognize that these examples are objective, deterministic, and based on universally accepted physical laws and phenomena. Software is, well, softer. In any event, there is great interest in having a single worldwide standard wherever feasible.

Standards bodies

Enter IEC, CCITT and ISO. These are the organizations which develop formal worldwide standards. Their names are actually French and won't be cited here. It is useful to recognize that, as a general characterization of their scopes, they handle electrotechnical (e.g., safety, EMI), telecommunications (e.g., ISDN) and everything else (e.g., food storage, mechanical contraceptive devices, fuels and lubricants) respectively. Development of standards for information technology spans the traditional boundaries of both ISO and IEC and, as such, is managed jointly by both ISO and IEC in Joint Technical Committee 1 (JTC1).

The membership of ISO, IEC, CCITT and JTC1 is at a higher level than organizations. CCITT is a treaty organization; the U.S. is represented by the Department of State. ANSI represents the U.S. as the member body in ISO, the national committee in IEC, and the national body in JTC1. AFNOR, BSI, SIS, DIN and ON, as examples, are the national body representatives of France, the UK, Sweden, Germany and Austria respectively. Other standards bodies participate in a limited manner. The most visible of these is the regional standards developer, the European Computer Manufacturers' Association (ECMA).

If one were to examine the scope of technical activity of JTC1, the unavoidable conclusion is that

it's broad and deep. Not only does it cover a lot of ground (media to programming languages) but it does so in excruciating detail. It is unrealistic to expect that a single organization can develop and promote a national body position in all the JTC1 technical activities. In the U.S., the concept of Technical Advisory Group (TAG) was put forward to handle this.

TAGs

TAGs are enabled by ANSI to develop the U.S. position on a limited set of items for specifically identified JTC1 activities. What's that mean? Essentially, that ANSI assigns to individual standards bodies the responsibility of handling U.S. technical interest for specific international standards activities.

Most of the time the enabling of specific organizations to act as a TAG for an international standards development effort is simple and not controversial. Individual programming languages, developed in working groups of JTC1 subcommittees, are typically simple. For example, there is only one U.S. group developing standards for POSIX, IEEE's P1003, and one for COBOL, X3's technical committee, X3J4. As a result, the identification of the TAG for those working groups is straightforward.

Sometimes the assignment of TAG responsibility is not so easy. Take the example of JTC1's subcommittee on languages, SC22. There are several U.S. standards development organizations which makeup the TAG to SC22. The IEEE has POSIX and Modula-2; CODASYL has FIMS; the Department of Defense has Ada; and X3 has a multitude of others. Within X3, a subgroup manages the issues that overlap those TAGs or that do not fall within the activities of any of them.

As a sidelight allow me to point out that the U.S. even needs a TAG to JTC1 itself.

Let's regroup for a moment. There are TAGs in the U.S. for all levels of ISO, IEC, CCITT and JTC1 in which the U.S. has an interest. Using JTC1 as a representative example, this means there are TAGs with responsibility for JTC1 itself, for the JTC1 subcommittees, for the JTC1 special working groups, for the JTC1 technical subcommittees, and for the working groups of each JTC1 technical subcommittee in which the U.S. is interested. If

there is activity anywhere within JTC1 that is of interest to the U.S., there is a TAG. Someone has to do the work.

Work of TAGs

A logical question now is "What are these TAGs empowered to do?" Simple question, but a complex answer. There are, however, some rules of thumb to help out.

- the more important the decision, the higher the U.S. consensus that is required
- the more detailed and technical the subject, the higher the weight given to the opinions of technical experts

Some examples are useful. One action JTC1 places a heavy value on is the final approval of standards. As a result, JTC1 rules require that a ballot be taken of its members (remember them, they're the representative national standards organizations) for adopting a document as a worldwide standard. Similarly, JTC1 members must vote to approve new work items. One way to think of this is that JTC1 makes decisions that affect the beginning (i.e. approval of new work items) and ending (i.e. final approval) for standards. JTC1 also makes all decisions involving the establishment and dissolution of its immediate subgroups.

JTC1 TAG establishes the U.S. positions on matters such as new work items and final approval of standards.

Let us examine a lower level, technical decision of JTC1. If we apply the second rule from above, we will conclude that the decisions at this level are delegated to the working group (of the technical subcommittee of JTC1). One such decision is the determination of what text should be

contained in a draft standard. It doesn't get much more detailed or technical than that. The working group determines that all by themselves.

There is a whole world of gray between the two examples cited above. Typically the decisions revolve around "recommendations to forward" documents in various stages of completion for consideration or approval by higher authority committees. A complete description of the decisions and recommendations is not really appropriate for this article. However, if you're interested, allow me to point you to two companion documents.

- the JTC1 directives (available from ANSI)
- the JTC1 TAG technical document 1 (available from CBEMA)

They are in the form of procedures but for ease of use have comprehensive indices.

There are a few messages for you to carry away.

- TAGs are empowered by ANSI
- TAGs represent the U.S.
- there are TAGs for every worldwide standards activity that is of interest to the U.S.
- the more important the decision, the higher the need for national body consensus; the more technical the decision, the lower the need for national body consensus

This has been a brief exposure of technical advisory groups in the U.S. As you have observed, the subject is complicated. The best way to understand them is to join one that is in your particular area of interest and expertise. Rather than fuss and fret about the problems you see with them, it is recommended that you become a part of the solution.

Book Review:

The Design and Implementation of the 4.3 BSD UNIX Operating System Answer Book

by Samuel J. Leffler and Marshall Kirk McKusick
(Prentice Hall, 1991, 85pp, ISBN: 0-201-54629-9)

Reviewed by George Neville-Neil, University of California at Berkeley

Since 1989, there have been some unanswered questions in the UNIX community. These questions were posed to us in a book called, *The Design and Implementation of the 4.3BSD UNIX Operating System*. Now, the answers are out.

In *The Design and Implementation of the 4.3 BSD UNIX Operating System Answer Book* you will find all the answers to the exercises that were given at the end of each chapter of the original book.

When I first got the book and saw that it only had eighty-five pages, I figured this would be a short read. I attempted to read the book by reading each question and answer in order while keeping the original book close at hand for use as a reference. The problem, if it is a problem, is that I read a question, thought about it, read the answer, and then thought some more. It's really a fun and fascinating read, but not one that can be done quickly.

Of course, most people will probably want to use it as a companion book when teaching a class and for this it is well suited. The book is divided into twelve chapters, which correspond to chapters two through thirteen in the original book. There is also a short preface, as well as a page of references, to round out the book.

In keeping with the system in the first book, the questions in the answer book are marked for difficulty. The exercises that are unmarked were answered directly in the preceding chapter. Exercises marked with a single asterisk will need some thought. Finally, the exercises marked with

two asterisks are considered to be "... major design projects or open research questions." The answers given to the two asterisk exercises take the form of suggestions on how to begin, and where to look when researching the answer to the question posed.

The book, in general, has a conversational tone. You get the feeling that the authors are answering your questions as if you had asked them in person, making the book easier to read. There is also a sense of humility in the writing so which keeps you from feeling that you are being lectured. The authors also admit when they have made a mistake, as in chapter 13, question 8:

Q: The reboot system call causes the system to halt or reboot. Give three reasons why this system call is useful.

A: The reboot system call is useful for these reasons:

- Safe halting and rebooting of a system requires support from the kernel.
- The existence of a system call permits applications to halt or reboot a system without operator intervention.
- The third reason was lost in an improper system shutdown.

In short, I really like this book. I found it to be interesting and informative. It is an indispensable resource for anyone planning to teach a course using the original book. The two books should probably be marketed to educators as a set.

AUUG
MANAGEMENT COMMITTEE
MINUTES OF MEETING 18 May 1992



Held at ACMS, Paddington

Present: Frank Crawford, Glenn Huxtable, Rolf Jester, Peter Karr, Chris Maltby, Michael Tuke, and Liz Fraumann

Meeting commenced at 10:10am

Richard Cousins and Piers Lauder of MHS Software presented a satisfactory proposal to the committee regarding a significant discount to AUUG members for the purpose of purchasing network connection software and it was agreed to by all parties.

This means that MHS will offer the service with a discount for members and we'll advise members via *AUUGN*. We committed to providing MSH a page in each *AUUGN* to keep members up-to-date on the service.

Ellen Gubbins of Symmetry gave a Public Relations report to the committee.

1. APOLOGIES

Apologies were received from Scott Merrilees, Pat Duffy and Jogada Crawford

2. MINUTES OF LAST MEETING (6 April 1992)

Moved (MT/RJ) that the minutes were accepted. Carried.

Items not specifically cited here are incorporate in the business of this meeting. The following items were carried over from the previous meeting:

- 2.1 AARNET Subscriptions (\$250.) and connections for WF & PK - little action had been taken. A new form has been developed and it was suggested that rather than sending a separate invoice for subscriptions it should be combined with the membership renewal forms. So moved GH/AG carried. It was also agreed that subscribers more than 6 mo. in arrears should be invoice now to alleviate the gap, and then will start the 6 mo period.

ACTION : WF /CM

2.2 Tax exempt status - No overt action was taken. It was suggested a tax lawyer should be consulted and necessary changes to the charter be drafted. FC will pick up this action item from MT.

ACTION: FC

2.3 Registration of Assets - It was unclear to WF as to which assets were to be registered. It was determined LF will draw up a draft list of assets and work with the Exec. Committee to locate such things as computer equipment, signage, banners, etc and maintain a log.

ACTION: LF

2.4 Feedback forms from summer conference - GH stated few of the summer conferences had formal critiques, but will attempt to do a summary.

ACTION: GH

2.5 Nomination Forms - It was moved and carried that the election process or Executive Committee would be extended... RJ/AG

2.6 AUUGN back issues - The surplus of back issues are organised and labeled in Softway store room with 2 copies each being held at ACMS. There is a significant gap of issues for 1991. It was suggested to make one complete set to be held at ACMS and by the AUUGN Editor. In addition, numerous copies of 1990 and 1991 conference proceedings were found and agreed these would be advertised for sale at \$50.00 (AUS) at the conference in September.

3. PRESIDENT'S REPORT

No report was given.

4. SECRETARY'S REPORT

Rolf Jester reported:

4.1 61 unfinancial members have been dropped from the AUUG membership.

4.2 Collection of nomination forms by RJ has been accomplished and he will interface with the returning officer, John O'Brien, and WF to facilitate balloting and mailing to the general body.

<u>Name</u>	<u>Position(s)</u>	<u>Nominated by</u>
David Baldwin	GC	Lawrence Peter Brown Ross John Hand Stephen Hodgman

<u>Name</u>	<u>Position(s)</u>	<u>Nominated by</u>
Frank Crawford	TR,GC,RO,ARO	Jagoda Crawford Ian Crakanthorp John O'Brien
Ross John Hand	GC	Lawrence Peter Brown Stephen Frederick Rothwell Stephen Hodgman
Glenn Huxtable	GC,VP	Janet Jackson T(?) Pedaste - nom of Uni. W.A. Major
Rolf Jester	SEC,GC	Eng Teoh Chris Jankowski Ron Danko
Peter Karr	GC	Patricia Duffy Rolf Jester Glenn Huxtable
Michael Kearney	GC	Stephen Frederick Rothwell Lawrence Peter Brown Peter Wishart
Chris Maltby	GC,ARO	Andrew Gollan P.J.Chubb Glenn Huxtable
Phil McCrae	PR	Chris Maltby Peter Chubb Andrew Gollan
John O'Brien	VP,GC,RO	Frank Crawford Jagoda Crawford Ian Crakanthorp
Greg Rose	GC	P.J.Chubb Lucy Chubb Rolf Jester
Stephen Rothwell	GC	Ross John Hand Lawrence Peter Brown Stephen Hodgman

<u>Name</u>	<u>Position(s)</u>	<u>Nominated by</u>
Michael Tuke	RO,ARO	Stephen Prince Stephen(?) Watson Barry Watson
Peter Wishart	SEC,GC	Ross John Hand Graeme Tinney (Nominee: Dept of Foreign Affairs & Trade) Elizabeth Keith

ARO=Assistant Returning Officer
GC=General Committee Member
PR=President
RO=Returning Officer
SEC=Secretary
TR=Treasurer
VP=Vice President

ACTION: RJ/JO/WF

4.3 Membership flyer will be updated and ready for distribution at the conference.

ACTION: RJ/LF

5. EDITOR'S REPORT

Frank Crawford reported:

- 5.1 The next issue of *AUUGN* is ready to go to the publishers and quotations from local printers are being gathered. A new section on book reviews will be inserted into the publication. Approximately 6 books a month will be reviewed. RJ/PK moved to accept and carried.
- 5.2 Advertising is permitted in *AUUGN*. LF is to remind FC re: invoicing for ads on a bi-monthly basis.
- 5.3 The contest for cover design will be announced in the next issue.
- 5.4 RJ/PK moved and seconded to accept report.

6. PUBLIC RELATIONS REPORT

Ellen Gubbins of Symmetry reported:

6.1 Insertions were placed in Computerworld, 17/4; PCW, 01/5; and Financial Review is interested in a story about AUUG 92 for 16/5 issue.

6.2 Available stories include:

- System V Release 4 - A Users Perspective by Frank Crawford
- Dept Mines and Energy
- Using a UNIX computer system as a file and print server for MS-DOS LANs
- Wide Area Information Services by David Baldwin
- Computer Crime by Ken Day.

6.3 Advertising Schedule for AUUG '92 is as follows:

- May
 - UniForum NZ

- June
 - Open Systems Review
 - UniForum NZ
 - UniForum Singapore

- July
 - Open System Review
 - UniForum NZ
 - UniForum Singapore
 - Computerworld*
 - Canberra Times
 - Australian*
 - Melbourne Age

* These publications will be repeated twice each during the month.

6.4 It was also recommended to complete housekeeping on the AUUG database to ensure its integrity

6.5 The next issue of the conference will be released at the conference.

7. TREASURER'S REPORT

Frank Crawford reported:

- 7.1 Review of PR dollars - it was determined that after the AUUG '92 conference that PR would be done on a project by project basis and the current revenue for this would be redirected to extend LF hours to accommodate additional responsibilities. Moved and seconded by RJ/GH. Carried.
- 7.2 Several categories of monies were redistributed and the overall budget remains as issued. See attached.
- 7.3 *Open Forum* - it was moved and determined that this publication would be discontinued. RJ will draft a letter to EG on behalf of PD to announce this.

ACTION: RJ

- 7.4 Discussion of a computer and laserprinter for the production of *AUUGN* took place. PK offered a SUN IPC for \$1000.00 FC will follow up.

ACTION: FC

8. CHAPTER SUB-COMMITTEE

Glenn Huxtable reported:

- 8.1 Work is still in progress. Brisbane is underway in the establishment of a chapter. Discussion ensued with Bernadette Garrin of Stallion /GH & LF
The sub-committee is comprised of: GH, SM, RH, JB.
- 8.2 Corporate Membership & Corporate Sponsors were discussed with continuing dialogue to take place and formalise a recommendation/
- 8.3 No progress had been made on the summer conference kits to date.

ACTION: GH/LF will produce
a conf. kit for organisers on

9. CONFERENCE

Liz Fraumann reported:

9.1 Programme committee met on Friday, 15 May for final selection of papers. A total of 49 papers were submitted. Notification was distributed via post with a copy of the Speakers' Guidelines on Monday, 18 May.

9.2 Peter Karr recommended an insert/tear-out for tutorial advertising/registration to be July issue of OSR. Cost would be \$400-\$500.

9.3 It was recommended that LF contact Elizabeth at Softway for information on accreditation for tutorials and streams

ACTION: LF

9.4 AG recommended a "show daily" similar to one provided at USENIX. PK offered staff member David Crow to do editing and slide equipment will be used for production. Format recommended was A3.

9.5 AG recommended a competition (Go, Next most important feature to be added to UNIX, etc.) It was suggested that the prize for the winner would be a Porsche.

9.6 It was determined that there will be both a "final program and Visitor Catalog" at this conference. This will enable a complete listing of attendees except for those registering at the conference.

ACTION: LF/WF

9.7 LF negotiated donated UNIX desktop slide production equipment for the conference.

9.8 Conference speakers' gifts will be addressed at the next meeting.

9.9 BoFs and Hospitality suites are being encouraged for companies not exhibiting.

9.10. It was agreed to pursue a "Face Saver" program sponsor to facilitate a directory as a direct member benefit.

9.11 MHS has agreed to look into providing messaging/mail for conference

9.12 KRE and Andrew Mc Rae are pursuing terminal room for conference.

9.13 Certificates of completion will be distributed for tutorial attendees.

9.14 AGM will be held on Wednesday, 9 September.

9.15 It was recommended to sell USENIX docs, and standards docs at stand.

10. OTHER BUSINESS

10.1 AARNET article. CM promised to complete the article for publication OSR.

ACTION: CM

10.2 Draft of survey for membership was submitted. Minor edits made and will be distributed before next meeting.

ACTION: LF/WF

10.3 Review the current share of the conference and exhibition with ACMS.
Set date for meeting on 2 June 1992.

ACTION: RJ/CM

10.4 It was moved and so passed that mail alises should be updated printed in AUUGN. RJ/GH

ACTION: SM

10.5 PK suggested in the membership brochure AUUG's mission statement is released.

10.6 It was recommended that LF attempt to negotiate discounted modem purchase for members with company(s) like Avtech, Cyco, Maestro, Interlink, Cytech, etc.

ACTION: LF

11. NEXT MEETING

Monday 13 JULY 1992 - 10:00am @ ACMS

Meeting adjourned at 3:40pm

Respectfully Submitted,

Elizabeth A. Fraumann

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 if you are currently really an AUUG member, examine the mailing label of this AUUGN. In the lower right corner you will find information about your current membership status. The first letter is your membership type code, N for regular members, S for students, and I for institutions. Then follows your membership expiration date, in the format exp=MM/YY. The remaining information is for internal use.

Check that your membership isn't about to expire (or worse, hasn't expired already). Ask your colleagues if they received this issue of AUUGN, tell them that if not, it probably means that their membership has lapsed, or perhaps, they were never a member at all! Feel free to copy the membership forms, give one to everyone that you know.

If you want to join AUUG, or renew your membership, you will find forms in this issue of AUUGN. Send the appropriate form (with remittance) to the address indicated on it, and your membership will (re-)commence.

As a service to members, AUUG has arranged to accept payments via credit card. You can use your Bankcard (within Australia only), or your Visa or Mastercard by simply completing the authorisation on the application form.

AUUG Incorporated

Application for Institutional Membership

Australian UNIX* systems Users' Group.

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

To apply for institutional membership of the AUUG, complete this form, and return it with payment in Australian Dollars, or credit card authorisation, to:

AUUG Membership Secretary
 PO Box 366
 Kensington NSW 2033
 Australia

• Foreign applicants please send a bank draft drawn on an Australian bank, or credit card authorisation, and remember to select either surface or air mail.

This form is valid only until 31st May, 1992

..... does hereby apply for

- New/Renewal* Institutional Membership of AUUG \$325.00
- International Surface Mail \$ 40.00
- International Air Mail \$120.00

Total remitted

AUD\$ _____

(cheque, money order, credit card)

* Delete one.

I/We agree that this membership will be subject to the rules and by-laws of the AUUG as in force from time to time, and that this membership will run for 12 consecutive months commencing on the first day of the month following that during which this application is processed.

I/We understand that I/we will receive two copies of the AUUG newsletter, and may send two representatives to AUUG sponsored events at member rates, though I/we will have only one vote in AUUG elections, and other ballots as required.

Date: ___ / ___ / ___

Signed: _____

Title: _____

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

For our mailing database - please type or print clearly:

Administrative contact, and formal representative:

Name:

Phone: (bh)

Address:

..... (ah)

Net Address:

Write "Unchanged" if details have not altered and this is a renewal.

Please charge \$ _____ to my/our Bankcard Visa Mastercard.

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

Name on card: _____ Signed: _____

Office use only:

Please complete the other side.

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

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

Who: _____ Member# _____

Please send newsletters to the following addresses:

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

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

Write "unchanged" if this is a renewal, and details are not to be altered.

Please indicate which Unix licences you hold, and include copies of the title and signature pages of each, if these have not been sent previously.

Note: Recent licences usually revoke earlier ones, please indicate only licences which are current, and indicate any which have been revoked since your last membership form was submitted.

Note: Most binary licensees will have a System III or System V (of one variant or another) binary licence, even if the system supplied by your vendor is based upon V7 or 4BSD. There is no such thing as a BSD binary licence, and V7 binary licences were very rare, and expensive.

- System V.3 source
- System V.2 source
- System V source
- System III source
- 4.2 or 4.3 BSD source
- 4.1 BSD source
- V7 source
- Other (*Indicate which*)
- System V.3 binary
- System V.2 binary
- System V binary
- System III binary

AUUG Incorporated

Application for Ordinary, or Student, Membership

Australian UNIX* systems Users' Group.

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

To apply for membership of the AUUG, complete this form, and return it with payment in Australian Dollars, or credit card authorisation, to:

AUUG Membership Secretary
P O Box 366
Kensington NSW 2033
Australia

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

This form is valid only until 31st May, 1992

I, do hereby apply for

- Renewal/New* Membership of the AUUG \$78.00
- Renewal/New* Student Membership \$45.00 (note certification on other side)
- International Surface Mail \$20.00
- International Air Mail \$60.00 (note local zone rate available)

Total remitted

AUD\$ _____

(cheque, money order, credit card)

* Delete one.

I agree that this membership will be subject to the rules and by-laws of the AUUG as in force from time to time, and that this membership will run for 12 consecutive months commencing on the first day of the month following that during which this application is processed.

Date: ___ / ___ / ___ Signed: _____

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

For our mailing database - please type or print clearly:

Name: Phone: (bh)

Address: (ah)

Net Address:

Write "Unchanged" if details have not altered and this is a renewal.

Please charge \$_____ to my Bankcard Visa Mastercard.

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

Name on card: _____ Signed: _____

Office use only:

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

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

Who: _____ Member# _____

Student Member Certification *(to be completed by a member of the academic staff)*

I, certify that
..... *(name)*
is a full time student at *(institution)*
and is expected to graduate approximately / / .

Title: _____

Signature: _____

AUUG

Notification of Change of Address Australian UNIX* systems Users' Group.

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

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

AUUG Membership Secretary
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# _____

AUUG Incorporated

Application for Newsletter Subscription

Australian UNIX* systems Users' Group.

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

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, 1992

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:

- | | |
|---|----------|
| <input type="checkbox"/> Subscription to AUUGN | \$ 90.00 |
| <input type="checkbox"/> International Surface Mail | \$ 20.00 |
| <input type="checkbox"/> 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# _____