AUUGN

The Journal of AUUG Inc.

Volume 16, Number 3
June 1995

Behind the Web at http://www.auug.org.au

Expanded WAUG section

Simple directory tools

Phil McCrea's last words as President

... plus reviews, Chapter news, and more

AUUG Inc.

UNIX & OPEN SYSTEMS USERS

ISSN 1035-7521

Print post approved by Australia Post - PP2391500002

AUUGN
Volume 16, Number 3
June, 1995

Table of Contents

AUUG Inc. acknowledges the generous support of its corporate sponsors:

IBM digital Sun

Contribution deadlines for AUUGN in 1995

Vol 16, #4 (August): July 21

Vol 16, #5 (October): September 22

Vol 16, #6 (December): November 17

AUUGN Correspondence

Please send all correspondence regarding AUUGN to:

AUUGN Editor
PO Box 366
Kensington NSW 2033

E-mail: *auugn@auug.org.au*

Submission Guidelines

Submission guidelines for AUUGN contributions are regularly posted on the **aus.org.auug** news group. They are also available from the AUUG World Wide Web site at **http://www.auug.org.au**

Alternately, e-mail to the above correspondence address, requesting a copy.

AUUGN Back-Issues

A variety of back-issues of AUUGN are still available; for price and availability details, please contact the AUUG Secretariat, or write to:

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

Conference Proceedings

A limited number of copies of the Conference Proceedings from previous AUUG Conferences are still available, at $50 each for members, and $60 for non-members. Contact the AUUG Secretariat for details.

Mailing Lists

Inquiries regarding the purchase of the AUUGN mailing list should be directed to the AUUG Secretariat. Telephone (02) 361 5994 during business hours, or Fax (02) 332 4066.

Editorial

Phil Anderson <phil@zikzak.net>

At the Great Sage once said, "everything changes!"; so it is at AUUG, and in the pages of AUUGN this month. Phil McCrea steps down as President; a new executive is established; the Western Australian chapter (WAUG) ceases to publish its own newsletter and moves that content to these pages (and very welcome they are too!); and I'm in the process of changing jobs. The last is the reason this issue's appearing a little later than normal; I trust this won't spoil your enjoyment of our offerings this time 'round!

I recently had a little feedback from some of you who've not had much contact with the World Wide Web. It seems that there are still plenty of folk out there who blinked and missed the radical changes in the way we access the Internet over the last 15 months or so ... and if that's the case in the UNIX/Open Systems community, imagine what it's like for all the other users out there! This issue's feature article tells of what's been involved in the Canberra chapter's establishment of their World Wide Web site, and the foundation of *The Internet Project*.

Speaking of articles ...

When I took over as editor of AUUGN, I promised myself that I wouldn't use the Editorial column as a gripe space ... so I won't. What I do want to say is that, except for the Western Australian and Canberra chapters, contributions and news from the AUUG state chapters seems to have dwindled to less than a trickle. No meeting reports, no reviews of guest speakers, no substantial announcements of any kind have crossed my e-mail box for months. Similarly, the number of members offering articles and/or papers seems to be down as well. Is it just an attack of the Winter Blues? I certainly hope so. If any of you have material you'd like to contribute to the journal, either pass it on to your chapter representatives, or mail it to me at **auugn@auug.org.au**.

AUUG members are keen to hear about relevant upcoming events as well; if you're involved in the organisation of conferences, courses or seminars of interest to the UNIX/Open Systems crowd, please be sure to forward the appropriate details to AUUGN as well. Publication deadlines notwithstanding, we'll add your event to the calendar, and put the details in our announcements section.

President's Report

Phil McCrea <pmc@syd.dit.csiro.au>

This is my last President's Column, as I step down from the lofty heights of my Presidential position in June. After being President for three years, I decided that AUUG needed some fresh blood at the top, and so I did not stand for election again. However, on the assumption that the referendum which accompanied the election is passed, I will remain on the Management Committee as Immediate Past-President.

I've enjoyed being President, and will certainly miss the role. Over these three years, UNIX has become more mainstream, and the percentage of AUUG members who form huddles in corners to discuss device drivers is dwindling (and ageing...). Our membership has grown substantially, and now represents a much broader base, as we have embraced more of an Open Systems focus. We now have around 1400 members, if you regard Institutional membership as two actual people. We send out in the vicinity of 1450 AUUGNs every two months.

Our public profile has increased as well, and AUUG is well known to the IT press. The weekly column in The Australian continues to give us good exposure, and the interest generated by the Annual Conference is impressive.

As far as our focus is concerned, the battle for UNIX has been won, at least to the extent that UNIX is firmly entrenched in the midrange, with the AS400 remaining as the only non-UNIX platform of any significance. But if you look at the AS400, its operating system, OS400, is becoming more and more UNIX like with the passage of time - at least from an external perspective. Even MVS is being cloaked with a UNIX-like mantle as it also becomes more Spec 1170 compliant.

Over the past year or two, AUUG's concept of openness has moved from the platform to the networking space, where TCP/IP embodies openness. Since TCP/IP is effectively UNIX's networking software, the scope of open networking based on TCP/IP fits quite nicely into AUUG's sphere of interest. This is reflected in this year's annual conference, which has the Internet as its main theme.

So, I bid you farewell as President, and thank you all for your support over the past three years.

Phil

AUUG BOOK CLUB & PRENTICE HALL AUSTRALIA

20% DISCOUNT TO AUUG MEMBERS

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

☐ ***Unix Systems Administration Handbook, 2/E (Book + CD - ROM)***
ISBN: 0131510517, Paperback, 1995, $98.95

☐ ***Applied UNIX Programming 4.2, Volume 1***
ISBN: 013304338X, Book + Disk, 1994, $82.95

*Deduct 20% from listed retail price

Name: ____________________ Position: ____________________
Organisation: ____
Address: ____
________________________________ Telephone: __________

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

Expiry Date: ____________ Signature: ____________________

Mail or fax completed order form to Prentice Hall Australia, PO Box 151, Brookvale NSW 2100

OR ✆ Use our PHONE SERVICE by calling Liz Guthrie SYDNEY (02) 907 5648

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

Conferences & Announcements

USENIX Conferences in July

Top Practitioners to Meet in NYC to Set Technology Agenda for Doing Business on the Internet

Everybody wants to do business on the Internet, yet relatively few companies actually are. Why not?

The USENIX Association will gather leading technologists, business folks, and practitioners of electronic commerce to debate and discuss the issues at this by-invitation-only workshop to be held at the Sheraton New York Hotel and Towers on July 11-12, 1995

What new technology is coming down the road? What are the legal issues? How do you receive payment? How do you secure your network? Who are the buyers? How will the World Wide Web be improved and extended? During refereed paper presentations, panels and working groups, workshop attendees hope to answer these important questions and help set the technical direction for the future.

Daniel Schutzer of Citibank will give the keynote address "Electronic Banking and Electronic Commerce on the SuperInformation Highway." Jay M. Tenenbaum, Founder and CEO of EIT, and Chairman of CommerceNet's Board of Directors, will also speak.

Following on July 13-14, 1995, will be the USENIX Association's Electronic Commerce Tutorial Program, also at the Sheraton New York Hotel and Towers. The Tutorial Program will allow dissemination of information to the security specialists and managers; technical staff and managers; and system and network administrators expected to implement electronic commerce systems for their companies.

Electronic Commerce Tutorial Program Announced by USENIX for July in New York

In response to the growing demand for the in-depth technical information required to implement electronic commerce, the USENIX Association announced their Electronic Commerce Tutorial Program to take place in the Sheraton New York Hotel and Towers on July 13-14, 1995.

How can our company secure our electronic commerce system from break-ins by outsiders? How do we take payment? What are our legal vulnerabilities? How do we create a Web site and use it effectively to market our product? These are a few of the questions that technical staff are being asked to resolve before companies can actually conduct business electronically, safely and successfully.

The USENIX Electronic Commerce tutorials will provide sophisticated answers to these critical questions and more. They are expected to attract Internet site developers, programmers, management, and authors wanting to learn about HTML, the page description language of the World Wide Web; security specialists and managers; technical staff and managers; and system and network administrators.

Electronic Commerce tutorial topics will include:

- The Law of Electronic Commerce for Non-Lawyers
- Security/Firewalls
- Cryptography Survey on payment mechanisms
- Agents: Active and passive, security
- Promoting your Internet business
- Creating Web documents with HTML

❧

USENIX is the UNIX and Advanced Computing Systems Technical and Professional Association. Since 1975 the USENIX Association has brought together the community of engineers, scientists, and technicians working on the cutting edge of the computing world. The USENIX conferences have become the essential meeting grounds for the presentation and discussion of the most advanced information on the developments of all aspects of computing systems.

For program and registration information, contact the USENIX Conference Office at 22672 Lambert Street, Suite 613, Lake Forest, CA 92630, 714 588 8649, fax 714 588 9706, e-mail **conference@usenix.org**.

Right now, Herb Johnson is also doing laundry, making dinner and talking to prospective customers with WebSite.™

Meet Herb. He's busy promoting his company on the World Wide Web, one of the fastest-growing networks on the Internet. And thanks to WebSite, a new server software package for NT 3.5 and Windows 95 users, he's getting lots of other important things done, too.

So can you. Think of WebSite as your opportunity to become a publisher on the Internet. What you communicate— and who it's for— is up to you. Maybe a progress report for people in your company, a research draft for reviewers, even recipes to share with anyone on the Internet. With our password and security system, you can easily target your message to the group of people you're most interested in reaching.

Fortunately, as Herb well knows, you don't have to be a rocket scientist to use WebSite. Its intuitive, graphical interface, online Help, and complete documentation will show you step-by-step how to publish on the Internet.

You simply won't find this power and flexibil-ity in a Web publisher anywhere else. WebSite's 32-bit server offers remote administration, multiple domain names, a graphical document manager, application or service capabilities, and the ability to include data from other programs such as Excel in Web documents using CGI. For more details, send email to **website@ora.com**

Or call Emma at woodsLane Pty. Ltd, on 1 800 803 443.

Look what it's doing for Herb.

THE WEB PUBLISHER FOR EVERYONE
WEBSITE
FROM O'REILLY & ASSOCIATES, INC.

O'REILLY & ASSOCIATES, INC.

Distributed in Australia by WoodsLane Pty. Ltd,

7/5 Vuko Place Warriewood NSW 2102 • Phone (02) 970 5111 • Fax (02) 970 5002

Paper:
Behind the Web at http://www.auug.org.au

Jeremy Bishop <jeremyb@auug.org.au>

Many AUUGN readers will be aware that AUUG now has its own World Wide Web site at **http://www.auug.org.au**, and some readers with access to the Internet will have browsed the AUUG Web site. For those of you who want to know more, here is how AUUG on the Web came to be.

To put these recent developments in context, it is firstly worthwhile providing a short history of dialup services that have been provided to local members by the Canberra Chapter in the past.

For many years, local members have had access to a system known as 'canb', a 386/SX25 PC clone, with 4Mb memory and 300Mb hard disk, running SCO UNIX. The system was located at Adept Software, a local company part-owned by AUUG member Stephen Hodgman. The system provided mail and news only, via a UUCP link to Adept's own system and then to ADFA. The system only had a single modem/dialup line. With only 4Mb of main memory, when a user dialled up shortly after a news feed had come in via UUCP, performance and response was less than ideal for the user.

About 2 years ago, recognising that the performance and services offer by 'canb' was becoming inadequate, we began work on 'canb2', an old Sun 3/160 workstation, donated by Sun Microsystems. The system had 24Mb memory, about 2Gb hard disk, mainly old Fujitsu Eagles donated by ADFA, which consume a not insignificant amount of power. Fortunately, the ANU Computer Services Centre (CSC) had graciously allowed us to located our system in the CSC computer room. News and mail was again via a UUCP link, although this time by direct serial link to an ANU system in the computer room. However, 'canb2' never got off the ground in terms of becoming a proper dialup system for our members. We struggled off-and-on for about a year to install and configure software, and most importantly, to get a reliable news feed. However, serial performance on Sun 3s was never their selling point - an interrupt per character sent or received, and it became clear that there was no way 'canb2' would be able to offer any significant performance improvement over the original 'canb'.

About this time we started to think more ambitiously, and the idea of a proper IP connection for our system was floated. After all, we were literally only metres from the AARNet ACT hub - never was the saying "so near, and yet so far" more appropriate :-)

Early discussions quickly led to the idea that the connection speed needed to be something greater than 9600 bps. However, connection charges to AARNet are not cheap, starting at $5000pa for a 9600 bps dialup link, $8000 for a 9600 bps leased line, and with only about 90 local chapter members, many of whom already had IP connectivity through their employers, the idea seemed infeasible.

"With hindsight, it was probably a good thing that the venture only involved PCUG and AUUG ..."

About this time the local PC User Group (PCUG), with about 2700 members, was beginning to look at the idea of providing Internet access to its members. The PCUG approached AUUG Canberra Chapter to explore the idea of cooperating to provide a joint Internet access service for all our members. Collectively, we also approached the local branch of the Australian Computer Society (ACS), and the local member of the Australian Public Access Network Association (APANA) for discussions. However, both the ACS and APANA declined, with ACS being involved in setting up their national ACSlink scheme, and APANA having concerns about committing to an essentially unknown expenditure. With hindsight, it was probably a good thing that the venture only involved PCUG and AUUG, as based on the many discussions and meetings held between PCUG and AUUG, reaching a three- or four-way agreement would have been very difficult!

Over the next month or two, numerous meetings were held with the PCUG to develop and refine the idea of a joint service. As mentioned, the PCUG has about 2700 members in and around the ACT, yet the local AUUG chapter has only about 90 members, and most of the discussions concerned how to provide a service that was fair and equitable to both parties given the disparate sizes of our organizations.

Whilst these discussions were taking place, we had approached the local office of Sun Microsystems looking for support and donations for "canb3", since it was clear that "canb2" was never going to be viable. Sun's generosity almost caught us by surprise, with Sun offering us a Sun 4/370 fileserver. Before they had a chance to change their mind, Merik Karman (Chapter President) and John Barlow (Chapter Secretary) quickly loaded the Sun 4/370 into the back of John's 4WD, along with a copy of Solaris 2.4, 32Mb of memory, two SCSI host adapters, two 670Mb SCSI disks, a 16 port MCP multiport serial adapter, and a QIC-150 1/4" cartridge tape drive. "canb3" was named "supreme", after the many pizzas that had been consumed during the now-aborted development of "canb2".

"Could we [AUUG & PCUG] jointly afford this?"

The discussions with the PCUG continued, and out of these came the basic concept that AUUG would contribute 'canb3' (the recently acquired Sun 4/370) and our UNIX expertise, whilst PCUG would contribute the critical mass of members necessary to get the venture going, administrative infrastructure to handle subscriptions, etc., and provide accommodation and power for the system at their premises in Fyshwick. We decided that the system should provide mail and news to members for free (from the AUUG perspective, a continuation of the service provided by 'canb'), which became known as "Basic Access", whilst the "Advanced Access" service would provide SLIP or PPP access via a terminal server for a charge to be determined. We had no firm idea of numbers of users, or what bandwidth would be required, and settled on initially providing 13 dialup lines and modems, and a 64kbps link to the Internet.

The selection of Internet service provider was made fairly quickly, with AARNet being the preferred choice. This was based on a number of factors, of which there are too many to detail here. Briefly, the other providers had charging schemes of varying degrees of complexity based on network utilisation, and put simply, we had little firm idea of what our utilisation might be, leading to worst-case scenarios of bills of $100,000 per year or more. Further, most providers seemed to have bottlenecks in their own connection to AARNet. Conversely, AARNet's charging scheme was simple, and it would be cheap to upgrade to faster links. AARNet have two charging models - VAR, with low bandwidth cost, but volume charges, or Affiliate, with high bandwidth cost, but no volume charges. We chose Affiliate, which would cost $25,000pa for a 64kbps link, plus a one-off setup charge of $5100.

Given the AARNet costs, plus the costs of a 64kbps ISDN connection, router, terminal server, modems, additional disk for the server, telephone line installation and rental, we calculated that jointly we would be required to invest about $55,000 for start-up costs, including the first years connection charges, plus an ongoing annual cost of $30,000, not including maintenance, repairs, and upgrades. Could we jointly afford this? Whilst AUUG (Canberra) had a healthy bank balance, we had concerns about the costs and possible "what-ifs?". After many different spreadsheet models were analysed by our treasurer, David Baldwin, we felt that the joint venture would need about 250 "Advanced Access" members at $120 each (a total of $30,000 revenue) as a minimum in the first year of operation to break even, and preferably 300 or so to make things comfortable.

So we decided to take the plunge. An agreement was drawn up between the PCUG and AUUG, and after several iterations and refinements (not to mention a lot of agonising), both parties were happy that it could go ahead. I should note here that the agreement is actually between the PCUG and AUUG, Inc., the national body, and not the local chapter. For those interested, the agreement is at **ftp://www.auug.org.au/pub/docs/agree.htm**. With the agreement in place, we then got the ball rolling by contributing $16,000 from Chapter funds, with PCUG putting in the same amount. Then we started the fun bit - approaching vendors around town for some "good deals". As mentioned above, Sun had already been very generous in donating the Sun 4/370 server, and the generosity continued from other vendors. Cisco gave us a great deal on two Cisco 2511s which are combined router/16-port terminal servers; Maestro gave us a super deal on 13 28800bps V.Fast modems; Functional Software gave us a copy of COS/ Manager, their system administration package (though we have to hope that AUUG never gets too religious, as the invoice stated that the software was "to be used only for non-prophet purposes" :-). We also had help from MKRL Consulting (ISDN Terminal

Adapter), Wardes (9Gb Seagate SCSI disk), Bay Networks (8-port UTP hub), and Dawn Technologies (8mm Exabyte tape backup unit), who supplied equipment at discount. Our thanks to all of these vendors for the assistance they have given us.

A name for the joint venture was chosen, and I doubt we get any points for originality - "The Internet Project". We set a target of February 1st 1995 as our official opening date, although we planned to have limited services available for testing prior to that. To help with testing, and to raise some early funds for cash flow, we decided to offer double time allocation for members who signed up for the project prior to the opening date. An early decision was that only administrators would have shell access to supreme, all other users would be restricted to a "menu shell" that would give them mail and news using Pine, plus some limited file transfer capability using rz/sz, and some basic housekeeping functions.

Another important decision was the amount of access time that users would be given. Initially we decided on 365 hours, to give a nominal 1 hour per day, but this was later "rounded off" to 400 hours. Rather than enforce a fixed time allowance per day, we decided to allow users to remain online for as long as they liked, but after the first hour online, to decrement their time allowance at a rate of 2-for-1 for the second hour, then 3-for-1 for the third hour, 4-for-1 for the fourth hour and so on. Thus the time allowance is not in real time but in what we christened "Monopoly Minutes".

We decided that Basic Access users (mail and news) would have 400 hours of "Monopoly" time at no charge, as part of their membership of PCUG or AUUG. Should they use this time up, they would be required to purchase Advanced Access time.

Advanced Access users have, in addition to their 400 hours Basic Access, SLIP/PPP access to AARNet and the Internet at large. 400 hours of Advanced Access "Monopoly" time costs $120 (nominally $0.30 per "Monopoly" hour), whilst smaller amounts of Advanced Access can be purchased in 50 hour blocks for $20 (nominally $0.40 per "Monopoly" hour). The full details of the charging scheme are explained at http://www.auug.org.au/tip/charging.html.

Whilst these prices are significantly cheaper than those of other Internet Service Providers, it should be emphasised that we are not, and do not wish to be seen as being, in competition with the VARs - we are simply providing Internet access to our members. The Internet Project is a non-profit venture for our members, and we are currently treated as an Affiliate of AARNet, not as a VAR.

The PCUG have their own premises with easy access, and it was a logical step to locate the system there. During January, services and functionality slowly came together as we grabbed software from the 'Net, built, installed and configured it, set up the first Cisco 2511, attached and configured modems, and waited for Telecom to install the phone lines and ISDN link to the AARNet hub at ANU. The ISDN link was installed, and a couple of late nights were spent getting the Cisco talking over the ISDN link to the AARNet hub. We have to thank Peter Elford and Mark Hales of Cisco for their late night assistance with this.

February 1st arrived, and the official opening was held at the PCUG centre. Our initial fears regarding (lack of) numbers of members was blown away by the announcement that The Internet Project (TIP) already had nearly 500 members, most signed up for advanced access!

"... we have recently added the 1100th member to the system, and the rate of sign-up shows little sign of slowing down ..."

It quickly became apparent that 13 dialup lines was not enough, particularly for AUUG members who had been used to relatively easy access to "canb". The Cisco 2511s support 16 serial lines each, so we decided to add an additional 3 modems and phone lines to the first 2511, making a total of 16 lines, to be used by the PCUG members, and to install another 7 modems and phone lines on the second 2511, to be used by AUUG members.

As I write this article in mid-May, we have recently added the 1100th member to the system, and the rate of sign-up shows little sign of slowing down. The vast majority of TIP members are from the PCUG, with about 40 AUUG members so far. Whilst the AUUG members only have 40 users competing for 7 dialup lines, the PCUG has nearly 1100 members competing for 16 lines. Needless to say, the PCUG modem pool is almost continuously in use (although 2am-5am is not so busy :-). Overall, the ratio of advanced users (SLIP/PPP) to basic users (mail/news only) is about 4:1.

Plans are currently underway to increase the physical memory on supreme, the server, to 64Mb, and probably to 96Mb or 128Mb sometime after that, with a view to locking some processes, like httpd daemons, in core. We are also expanding disk capacity, with AUUG national funding a 2Gb disk to be used for AUUG Web pages, ftp archive, and face-saver images from the AUUG conferences.

It has to be said that the growth and demand for Internet access by the PCUG membership has taken us somewhat by surprise, and whilst the solution seems obvious - buy some more modems and phone lines, there are a few considerations to be taken into account. Firstly, the funding for this needs to come from PCUG, and they have many other commitments, such as their Bulletin Board, and many other services to their members. Secondly, to adequately service 1100 members or more, who are expecting to have nominally 1 hour per day access to the system, some back of the envelope calculations quickly show that this requires at least 42 dialup lines, and preferably more to provide for demand at peak hours, so lets round it up to 48 lines. This is 32 additional phone lines to be paid for and installed, 32 additional modems to be purchased, and another two 16-port terminal servers, a not insignificant cost in total!

"... the major issue that we have to consider is the impact of the Telstra takeover of AARNet, and the changes to charging arrangements for AARNet's former commercial members."

Next comes bandwidth and usage of our AARNet link. The current 64kbps link is happily supporting the current peak time user base - 16 PCUG and 2 AUUG dialup users, plus a few users logged in across the link. At these times, the AARNet statistics (**ftp://ftp.aarnet.edu.au/pub/usage**) shows that the link is typically 75% utilized. However, adding another 32 dialup lines is obviously going to require a link upgrade, probably to 128kbps, an additional cost itself.

Looking to the future, the major issue that we have to consider is the impact of the Telstra takeover of AARNet, and the changes to charging arrangements for AARNet's former commercial members. Other issues that the administrative team are currently discussing is the option of giving all members, both Basic and Advanced, use of SLIP/PPP, but preventing access outside the TIP network to Basic access users by blocking their access at the gateway router.

With the establishment of the system, delegation of the **auug.org.au** domain has been transferred to the new system, and the AUUG mailing lists, previously located at munnari.oz.au, have also been moved. As well as AUUG national mailing lists, and Canberra Chapter lists, we are more than happy to host mailing lists for other chapters, and NSW chapter have already taken us up on this offer. For more information, contact **cauug@auug.org.au** or **postmaster@auug.org.au**

The mailing lists are being managed using Majordomo, under the oversight of our postmaster, Peter Wishart, who also fills his spare time as newsmaster, uucpmaster, and AUUG national secretary.

UUCP connections are available for members, with about 6 people currently receiving their news and mail this way. We hope to encourage more users, particularly PCUG users, to use UUCP in the wee small hours, in order to reduce demand for interactive dialup access.

We also have an AUUG FTP archive (**ftp://ftp.auug.org.au/pub/auug/**). There's not too much there yet, so if you have any contributions, please let us know. If you like, you can upload material to **ftp://ftp.auug.org.au/pub/** incoming. If you do, please send a short description of what it is, and where it should go, to **ftpmaster@auug.org.au** within 7 days, otherwise it will be deleted.

For 'Net, surfers, if you point your Web Browser at **http://www.auug.org.au/** you will get *The Internet Project* home page, with links to the PCUG, AUUG national, AUUG Canberra Chapter, and additional TIP pages. The AUUG national home page proper is at **http://www.auug.org.au/auug/**, with the Canberra Chapter page at **http://www.auug.org.au/cauug/**. AUUGN on the Web can be found at **http://www.auug.org.au/auug/auugn/**. If you have any questions about, or additions to, these pages, please let us know.

We would like to establish Web pages for the other chapters. If your chapter already has a Web page at another site, please let us know where so we can add a link to the AUUG home page. For those chapters wanting to set up a chapter home page you might like to take a look at the Canberra Chapter page (**http://www.auug.org.au/cauug/**) for some ideas.

Finally, I should mention a few names of key volunteers in this project. The Internet Project, and AUUG on the Web, would not have got off the ground without the assistance and many hours of volunteer effort by Merik Karman (Canberra Chapter President), Karl Auer (PCUG President), Lawrie Brown (WWW and DNS administration), Stephen Rothwell (user accounting software), Michael Lightfoot and Peter Davie (FTP administration), Peter Wishart (news and email), David Baldwin (Treasurer), John Barlow and Jeremy Bishop (security).

Special thanks should go to Karl Auer, AUUG member and President of the PCUG, who was the driving force behind PCUG's involvement, and has spent many hours setting up TIP Web pages, and supplying advice and help to PCUG users taking their first tentative steps on to the Internet and into the world of UNIX and Open Systems.❖

UNIX Tricks & Traps

Edited by Janet Jackson <janet@dialix.oz.au>

This month's theme is neat little programs that show why programmers love UNIX. Graham Jenkins has written a couple of tools that use **find** to recursively do a task on all files in a subdirectory. Michael Henning's **edpipe** lets you edit data on its way through a pipe. And my text-centring program is particularly powerful when combined with certain vi commands.

In the long, explanatory version of his program, Michael explains a number of common shell-programming idioms. Less experienced shell programmers should have a look.

You'll notice I'm saying "program", not "script". This is because I'm sick of people who think that programs written in interpreted scripting languages are somehow trivial, and not "real" programs. I suppose this is because such programs are often short, and because they are their own source code, which means they have no aura of inscrutability. I believe both of those are good things. So from now on, my editorial policy will be that scripts are referred to as programs.❖

Some Simple Directory Tools

Graham Jenkins <gkj@cbisa.com.au>

String Search

Did you ever have to search through all the files in your system news directory because you wanted to find an article about ultra-widgets? Or perhaps you forgot the email address of someone at ACME Corporation, but recollect that their name recently appeared in a news item.

You could try something like:

```
cd /usr/spool/news
egrep -i ultra-widget */* */*/* */*/*/*
```

But you would probably get back something like "arg list too long". An alternative is to use the following **grepdir** program, thus:

```
cd /usr/spool/news
grepdir -i ultra-widget
```

Here's the program.

```
#!/bin/sh
# @(#) grepdir Performs 'grep' for designated string on all files under
# current directory. Graham Jenkins, CBIS Aust., January 1995.

usage() { cat <<EOF
Usage: `basename $0` [ -bcilnsv ] [ -m ] expression
   e.g.: `basename $0` -i Xwindow

Performs 'grep' operation using the designated 'grep' options for
the expression shown on each file in the current directory and its
sub-directories. Troublesome control-characters are suppressed and
output lines are restricted to less than 80 characters.

If the '-m' option is included, then only those files within the
filesystem containing the current directory are considered.

EOF
exit 2
}

# Extract options, assemble into 'grep' options and 'find' options.
# Note that some machines may require '-xdev' in place of '-mount'.
#
while getopts bcilnsvm OPTION; do
   case $OPTION in
      b|c|i|l|n|s|v ) GREPOPT="$GREPOPT$OPTION";;
      m ) FINDOPT="-mount";;
      \?) usage;;
   esac
done
shift `expr $OPTIND - 1`

# Perform 'find' operation, pipe output into 'grep', delete troublesome
# control characters, cut lines to less than 80 characters. Note dummy
# search of /dev/null to force filename output.
#
if [ $# -eq 1 ]; then
   find . $FINDOPT -type f -print 2>/dev/null | while read F; do
      ( if [ -z "$GREPOPT" ]; then
          grep "$1" $F /dev/null
        else
          grep -$GREPOPT "$1" $F /dev/null
        fi
      ) | tr -d "[\001-\006]" | tr -d "[\016-\037]" 2>&1 | cut -c1-79 2>/dev/null
   done
   exit 0
else
   usage
fi
```

Content Comparison

I recently had occasion to beta test a new version of **unzip**, and needed to compare the contents of all the files in two different directories (produced by previous and new versions of **unzip**). The **cmpdir** program hereunder makes such a comparison easy. It can also be used to verify that a file-system copy has completed satisfactorily and/or to ascertain which files have changed since a file-system copy was made.

```
#!/bin/sh
# @(#) cmpdir   Compares files under current directory with those under another
#               directory. Graham Jenkins, CBIS Australia, August 1994.

if [ $# -eq 1 ]; then
   if test -d $1; then
      echo "Comparing `find . -type f -print|wc -l` files .."
      find . -mount -type f -print | while read F; do
      cmp $F $1/`echo $F | sed -e 's_\.\/__'`
      done
      exit 0
   else
      echo $1 is not a directory!
      exit 1
   fi
else
   echo Usage: `basename $0` dir .. compares each file under
   echo current directory with its equivalent under dir ..
   echo Note: Only those current directory files on the filesystem
   echo which contains the current directory are considered.
   exit 2
fi
```

A Program to Debug and Edit Pipelines

Michael Henning <michi@citr.uq.oz.au>

Below is a program that I found useful over the years. It is short, elegant and does a great job—all the things that made UNIX popular :-)

I didn't invent this all myself—I got hold of a primitive version during my university days, and improved it to "industrial strength" some years later. Sorry, but I can't attribute the original source any more.

The program is called **edpipe** and allows you to debug and edit pipelines. Very useful if you are into writing complicated shell programs a lot.

Use it like:

```
   cmd1 | edpipe | cmd2 | edpipe | cmd3 | edpipe ...
```

You can look at and edit the data that goes through the pipeline at any point.

Here is the program.

```
#!/bin/sh
#
# This makes sure that the script is executed by the Bourne shell, regardless
# of the login shell of the user (must be first physical line of file, and
# must start in column 1).
# To be safe, don't put a space between the "#!" and the "/bin/sh" -
# some UNIXes don't like it.
#


#
# Produce a usage message if invoked as "edpipe -?". The use of `basename $0`
# ensures that the correct message is produced, regardless of the name of
# the script.
#

[ $# -eq 1 -a "$1" = '-?' ] && {
    echo "usage: cmd | `basename $0` [editor_args...] | cmd" >&2
    exit 1
}

#
# Look up the EDITOR and TMPDIR environment variables and use their
# contents if they are set. If not set, default the values to
# /usr/bin/vi and /tmp.
#

: ${EDITOR:=/usr/bin/vi}
: ${TMPDIR:=/tmp}

#
# Set TMPFILE to a name in $TMPDIR, which is unlikely to clash with other
# names. Filenames generated are of the form "/tmp/.edpipe2538", where
# 2538 is the process id of this shell ($$ expands to the current process id).
#

TMPFILE=${TMPDIR}/.edpipe$$

#
# The trap is interesting, because it catches the most common signals
# (including process exit - signal 0), but also resets all the signals
# to be ignored once the trap is triggered. This is to prevent the trap
# from firing twice on receipt of an interrupt. If this was written as
#
# trap "rm -f $TMPFILE; exit 0" 0 1 2 3 15
#
# and a SIGTERM was sent, the trap would fire twice - once on receipt of
# SIGTERM, and a second time on process exit. In this particular case
# it wouldn't matter, but if the trap action does something that is not
# idempotent, it can cause problems.
#

trap "trap '' 0 1 2 3 15; rm -f $TMPFILE; exit 0" 0 1 2 3 15

#
# This line simply captures stdin in a temporary file
#

cat >$TMPFILE

#
# This line makes sure the editor is invoked correctly. Stdin and stdout
# redirected to the tty device, otherwise the editor will try to read
# from, and write to, the pipe. We don't need to redirect stderr, since
# it is connected to the tty by default.
# The expression ${1:+"$@"} passes any other arguments given to edpipe
# through to the editor. So, for example, if you write
#
#    echo hello | edpipe -R
#
# the -R option is passed through to vi to invoke it in read-only mode.
# Vi is invoked with the temporary file argument last, for example,
#
#  "/usr/bin/vi -R /tmp/.edpipe2538".
#
# The expression "$@" is used in preference to $* because it preserves
# embedded white space in arguments (prevents the shell from tokenizing
# on white space embedded in arguments when the command is expanded).
#

$EDITOR ${1:+"$@"} $TMPFILE </dev/tty >/dev/tty

#
# Once the editor quits, simply cat the temporary file to stdout, to
# pass the contents to the next stage of the pipeline and exit with zero
# (success) status. The trap at the top of the script takes care of removing
# the temporary file on exit.
#

cat $TMPFILE
exit 0
```

Centring text, with and without *vi*

Janet Jackson <janet@DIALix.oz.au>

I wrote the following program, **centre**, for doing SAGE-AU meeting notices, and have since found it extremely handy. It centres lines of ASCII text, ignoring existing leading and trailing whitespace. Each line is centred individually. This is against the AUUGN submission guidelines, so I won't demonstrate it. It's useful for putting headings in news postings and suchlike.

It can be used as a filter, converting text on standard input to new text on standard output, so you can use it with the vi editor's powerful program-calling commands. For example:

!!centre	*centres the current line*
!}centre	*centres each line from here to the end of the paragraph*
!Gcentre	*centres each line from here to the end of the file*

Power **vi** users will recognise "}" and "G" as the "go to end of paragraph" and "go to end of file" commands. All the "go to" commands can be used with "!" in this way.

```
#!/usr/bin/perl
# centre - centres input lines in width 80 or as given on commandline.
# Existing leading and trailing whitespace is stripped.
# Lines are assumed to end in a newline.

$USAGE = "Usage: $0 [-w width] files...\n";

$WIDTH = 80;

require 'getopts.pl';
$@ && die "$0: require getopts.pl failed\n";

&Getopts( 'w:' ); # -w takes arg
$opt_w && ($WIDTH = $opt_w);
( $WIDTH <= 0 ) && die $USAGE;

while (<>)
{
   # remove newline & leading and trailing space from line
   chop;
   study; # optimise
   s/^\s*//;
   s/\s*$//;

   # if line is blank, just print a newline
   /^$/ && ( print("\n"), next );

   # print space at left of line
   print( ' ' x int ( ( $WIDTH - (length($_)) ) / 2 ) );

   # print line, & newline
   print("$_\n");
}

exit 0;
```

From the Western Front

Edited by Janet Jackson <janet@dialix.oz.au>

Welcome to the bigger and better WA section. Our local newsletter **YAUN** is no more, so everything's in **AUUGN** now. Please send me your comments, suggestions and contributions.

AGM

We held our AGM on 17 May and elected a new committee, who are listed below. Instead of a speaker, we had a "UNIX Networking" session, where everyone stood up and gave a 1-minute description of their interest in UNIX (and in UNIX networking, if any). Even though we had done something similar last year, it was interesting to hear more about where people are coming from.

April talk

In April, WAUG member Tom Hallam gave an excellent talk on how to get Macs and UNIX systems to co-operate. I'm disappointed that no-one has sent me a review or summary of this talk – but it's not too late for you (yes, you) to write one for the next issue of AUUGN.

As I'm having no success in getting people to review/summarise meetings, I think I'll suggest to the committee that we appoint a couple of Meeting Reporters whose job it is to report on meetings for the benefit of those members who can't get there.

New venue

Our new venue, the Freeway Hotel in South Perth, is a major upgrade over the previous dive. The bar is more congenial, the room is bright and airy, and the food is about 500% better. And the Ladies' is actually *clean*. So you ladies (and gents) who haven't been coming along, try it now!

❧

Comings & Goings

As of May 1st, **Cameron Ferstat** is on a two year assignment in IBM's International Technical Support Organisation in Austin, Texas (Austin is where all the AIX and RISC System development goes on). He will be responsible for running projects to develop "how to" manuals and workshops for some of IBM's new products.

From the Chair

Adrian Booth <arena@DIALix.oz.au>

[This is the Chair's report for 1994-1995, as read at the AGM.]

Another successful year for WAUG.

Two major events were held: the special September meeting with Linus Torvalds as a guest speaker, and the summer conference. Both events were successful at bringing benefits both to AUUG members and to the Perth Unix community.

We also came within a whisker of having Gene Spafford over to WA, and will be lobbying hard for WA next time he comes down under.

Our meetings continue to be successful and well-attended thanks to the efforts of Mark Baker, our meeting organiser. And our change of venue to the Freeway Hotel seems to have been a very positive one.

Now for the year ahead.

Brent Chapman will be visiting Perth soon. David Buck from Industrial Micro Products is coordinating this and doing a fine job.

I think that WAUG's goal for the next 12 months should be to raise our profile in WA. I don't want us to get into a "get members for members' sake" rut, but I think we need to make the Unix community aware of our existence and of what we can offer so they can decide whether or not they want to participate. I hope to see this happen through greater publicity of our events and meetings. In fact I would like to see a "publicity officer" on the WAUG committee.

As many of you know I have decided not to be involved with the organisation of next year's summer conference. It has become (and in reality was always) far too complex for one person to look after, and to try to maintain a living at the same time. The next conference will be organised by a small committee which I would like to see largely comprised of WAUG committee members. (I will of course be providing advice to the conference committee, and will probably get involved again after a one year break.)

What does this mean in practice? That if you are thinking of nominating for the WAUG committee, you should consider seriously whether you have the time to make a positive contribution. While the time is not onerous, you should be prepared and able to accept some responsibilities along with the position.

Finally I would like to thank the WAUG committee for all of their efforts over the past year – many of them have spent large amounts of time performing invisible, thankless tasks. WAUG would certainly notice if they stopped doing them.

❧

Meeting information

WAUG meets at the **Freeway Hotel, 55 Mill Point Road, South Perth**. We meet at 6:15pm on the third Wednesday of each month.

Our meetings are advertised in the Diary column of the Computers section of Tuesday's *West Australian.*

If you need further information about the next meeting, please contact Mark or one of the committee.

SPEAKERS ARE NEEDED!

... especially ones who can actually commit to giving a talk on a certain date! So if you can give a talk, or know someone who can, please let us know. Mark (our meeting organiser) cannot produce them out of thin air.

❧

WAUG e-mail aliases and newsgroups

WAUG has the following mail aliases on uniwa.uwa.edu.au:

- **waug-membership** – for membership enquiries
- **waug-chair** – our Chairperson
- **waug-meetings** – our meeting organiser
- **waug-secretary** – our Secretary
- **waug-newsletter** – for newsletter contributions or enquiries
- **waug** – for general correspondence (will be read by the Secretary, as a paper letter would be).

So, for example, you may send general correspondence to **<waug@uniwa.uwa.edu.au>**.

Check out the newsgroups **wa.waug** and **aus.org.auug** for announcements and discussion.

❧

Committee contact details / Election results

Office-bearers

- Chair: Adrian Booth (09) 354 4936 **<abcc@DIALix.oz.au>**
- Treasurer: Patrick Ko (09) 483 8111 **<patrick@cs.curtin.edu.au>**
- Secretary: Vacant, but see below

Ordinary committee members

- Mark Baker (09) 491 6081 **<baker@telecomwa.oz.au>** (Meeting Organiser)
- David Buck *[no details provided]*
- Luigi Cantoni (09) 474 3700 **<lui@DIALix.oz.au>**
- Don Griffiths (09) 351 7691 **<griffith@cs.curtin.edu.au>**
- Tom Hallam (09) 380 2665 **<thallam@geol.uwa.edu.au>**
- Glenn Huxtable (09) 328 8288 **<glenn@fs.com.au>**
- Janet Jackson (09) 272 5061 **<janet@DIALix.oz.au>** (AUUGN Sub-editor)
- James Patton **<mrdwa@DIALix.oz.au>**

At the AGM there were no nominations for Secretary. The incumbent Secretary, Major, was not present, and it was resolved to approach him about continuing in the position. Major may be contacted at (09) 357 5076 or **<major@yarrow.wt.com.au>**.

The portfolios of the ordinary committee members have not been officially assigned yet, as the new committee has not met. However, it seems likely that they will continue as above. We may also appoint a publicity officer.

❧

For Systems Administrators: Local SAGE-AU meetings

The WA Regional Group of the Systems Administrators Guild of Australia (SAGE-AU) meets on the fourth Tuesday of each month at 6pm, in room G3 at the Alexander Library.

If you manage computer systems for a living, we'd like to have you along. SAGE-AU is not another Unix group. All systems and network administrators are welcome. We would particularly like to see more PC network administrators attending, so if you know any, send them along.

For more information, please contact the regional group chair, Janet Jackson **<janet@DIALix.oz.au>**, (09) 272 5061, or the meeting organiser, Mike Horton **<mikandfl@DIALix.oz.au>**, (09) 479 8424. For information about SAGE-AU in general, you may also look at **ftp://ftp.mel.dit.csiro.au/pub/SAGE-AU** and **http://sage-au.dstc.edu.au:8080/**.

Background:

A Better Way to Access Files - Mapping

Kevin Sheehan <kevin@uniq.com.au>
Uniq Professional Services

Reprinted from Open Software Review, with permission.

What is file mapping?

Most of the operating systems being used in the open systems market today use some form of demand paged virtual memory. In these systems, a portion of a process' address space is associated with some object, and as you **fault** them in (access a page that is not in memory) the system pages them in on demand (allocates a page and fills it with the contents indicated by the mapping).

While this discussion is UNIX oriented, due to the experience of the author, the techniques and benefits apply to any system employing this method. In UNIX, the facility is made possible by what is known as the Virtual Memory (VM) system. This first appeared in its current form in SunOS 4.0, and is a staple part of SVR4.

In SVR4, the **mmap()** system call is probably the least documented and best kept secret. It its general form, it takes a file or device and associates a part of your address space with it. When you access a page that is not mapped to anything, the system allocates a page for you, fills it in, then restarts your process at the access point.

As an example, if you **truss** (trace the execution of) a process **mmap()** is used extensively to start up the process. The run time loader first maps the pages of your executable, then maps the pages for any shared libraries you use as you access them. These are called Copy On Write (COW) or mappings.

When you read from a page, the system gets the page from the **vnode** (a generic abstraction of files in UNIX) and hands it back to you in read only form. When you write to the page, a private copy is allocated. If any other process is looking for the same read-only copy of the page (like the third page of the **/bin/ls** executable for instance) then it gets a mapping to the same physical page. If you fault on one of the pages you have written, you get your private copy.

This means that text and read-only data pages are shared, and private copies of the write data and uninitialized storage are private. When your process is running, it is generally a bunch of shared mappings of your executable and the libraries with which it was dynamically linked.

File mapping for I/O

So what does this buy the poor application writer looking to use the system more efficiently? The **mmap()** system call is available to them as well, and has many benefits versus the traditional calls such as **read()** and **write()**

First, let's look at how a **read()** call really works. You tell it which file you want read with a file descriptor, and you give it the address of a buffer in your address space where some offset of the file will be read. In order to do this, the system has to read in the file blocks to physical memory in one place, then allocate pages for your process and copy the contents over.

For a few bytes, this is not a terrible burden, but imagine if you are reading a huge file of 16MB. All 16MB of the file will have to be brought into memory, and all 16MB of your buffer will have to go through memory in the operation of copying and will require writing back to the paging device as it has been modified. All before you have touched the first byte!

Contrast this to an **mmap()** of the same file. You ask for the file to be mapped. This associates some part of your address space with the file. When you touch a page, the system will bring in the page of the file for this offset - *but* you, not a copy in your own buffer. You get the bytes you need, when you need them, and you don't use twice the memory. And if you don't modify the page, the system doesn't need to write it back.

Another benefit of this direct mapping is that everybody who maps the same file will get the same physical page mapped for them. Imagine that the 16MB file above was being used by two processes. If you are using **read()** to get the contents, you have 16MB for the file pages (most likely at two different times) and 16MB apiece for each of the processes. If both processes use **mmap()** instead, then only the pages they are both using of the file are in core, and both processes share them. In other words, the same benefit that shared objects provide, but for file I/O.

Other benefits of mapping

The other nice thing about mapping files is that you can either explicitly control or hint to the VM system about what to do with the pages. When you **read()** into your private buffer, there is no way to tell the system how to manage that physical memory, or what pages you will and won't need over time.

In SVR4, the **msync()** and **madvise()** system calls allow you to do just this. With **msync()** you can tell the system to flush pages to disk or invalidate the mapping (get a new copy the next time you touch it). With **madvise()** you can tell it that you do or don't need some pages, as well as giving it hints about the pattern of access, so it can decide better when to free memory or read ahead.

An example

The first example in which the author used **mmap()** was for a printing application. A 250MB 8-bit RGB image had to be dithered down to four colours for a CMYK printer on a Sun3. In this case, we decided to do the operation in–place, so what we had was an application that wanted to read, modify and write 250MB.

Using **read()** we had to wait for all 250 MB to be read into the private buffer, which then required 250MB of swap. Lunchtime for that alone wouldn't suffice, much less for the time to move through the 250MB and then go through the whole thing again when it came time for the **write() mmap()** when we touched the first page, it and the next (read-ahead) page were brought in. We then modified the pixels and moved on. When the next page was faulted, we had advised the system to do free-behind, so the page no longer needed was written to disk as we did the processing for the next page. In other words, exactly the amount of memory and I/O as required, and at the best possible times.

Summary

File mapping allows you to do exactly the work needed for accessing files. It allows you to share the physical memory used for files, and eliminates private copies. It allows you to advise the system on the best use of physical memory for a mapping. It makes application programming easier, as you can treat data as a pointer, not a buffer to be managed.

In short, it tastes great, it's less filling, and even cleans the kitchen sink. What more could you ask of a nice simple interface?? :-)

Kevin Sheehan is a director of Uniq Professional Services, *a company specializing in the needs of the open systems market. Kevin does work in drivers, kernel mods, performance analysis and engineering. More detailed information on the* mmap() *system call and its uses is available from papers presented over the years. For further information, please contact him as* **kevin@uniq.com.au**.

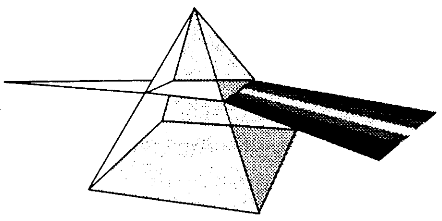
COBOL
compilers for UNIX

Micro Focus COBOL for

- **SUN**
- **IBM**
- **DEC**
- **HP**
- **Pyramid**
- **and SCO**

also for
AT&T/NCR, BULL, DG, ICL, Motorola, NCR, Novell, Prime, Sequent, SGI/MIPS, SNI, Stratus, Tandem, Unisys

Client/Server and Cross-platform Development Tools



Graphical and Character User Interface Tools
Micro Focus Dialog
Portable across UNIX, Windows, DOS and OS/2

ADVANCED SOFTWARE ENGINEERING
Workstation Development Specialists for all your COBOL needs

36 Camberwell Road
Hawthorn, Vic 3123
PH 03 9882 7522, FAX 03 9882 7306

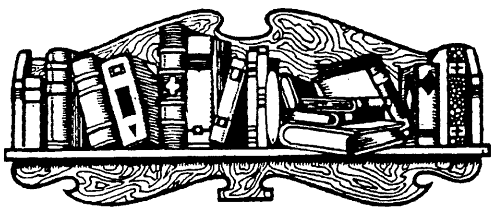
Level 1, 845 Pacific Highway
Chatswood, NSW 2067
PH 02 904 6111, FAX 02 904 6007

Book Reviews

Frank Crawford <frank@ansto.gov.au>
Review editor

Well another AUUGN rolls around, and more book reviews are published here, some good, some not so good (the books, not the reviews). Probably the most notable is Evi Nemeth, et.al. second edition of UNIX System Administration Handbook. This is especially pleasing as Evi will be in Australia in July at the SAGE-AU'95 Conference in Wollongong. Aside from this we have books ranging from an introduction to the Internet, to advanced UNIX programming and many in between.

There are quite a number of books being reviewed, and more in the works, which means that we have lots of scope for new reviewers. The current practice is to post a note to the newsgroup **aus.org.auug** when we have new books available. Unfortunately, this disadvantages members without network connections, or on the end of a low speed link. For people in such a position, either mail, via the AUUG PO Box, or fax me on (02) 717 9273, with your contact details and preferences.❖



UNIX System Administration Handbook, Second Edition

by Evi Nemeth, Garth Snyder, Scott Seebass and Trent R. Hein
Prentice Hall
1995, 780 pages inc. CD-ROM, Softcover
ISBN 0-13-151051-7
Reviewed by Frank Crawford ANSTO <frank@ansto.gov.au>

As a professional System Administrator, I have long believed that Evi Nemeth's book, *UNIX System Administration Handbook,* is a must for my professional library. Unfortunately, I've never quite got around to buying a copy, and after a while I started to believe it was getting a bit dated, being originally released in 1989. Now the second edition has been releases, which appears to have been completely rewritten and expanded.

This book is written by experienced System Administrators for System Administrators and touches on all areas of System Administration that I can think of. Just to give an indication of the wide range covered, it includes: booting and shutting down, filesystems and file management, controlling processes, adding users, devices and drivers, serial devices, disks, backups, logs, kernel configuration, networking and network management, the Internet, mail, news, printing, security, hardware maintenance, performance analysis, policies and politics. This is a fair indication of the breadth needed by a System Administrator, and thus covered by this book.

The book itself is broken into three large chunks: Basic Administration, Networking (new for this edition) and Bunch o' Stuff, and in an effort to give concrete examples covers six different operating systems: Solaris 2.4, HP-UX 9.0, IRIX 5.2, SunOS 4.1.3, DEC's OSF/1 2.0 and BSD/OS 1.1. This choice of OSs gives a good split between System V– and Berkeley–based systems, and consists of some of the most common systems.

As the title indicates, this is written as a handbook, broken into a number of different topics and giving step–by–step instructions on how to perform specific tasks for representative operating systems (often giving different list for each OS). As well, the authors often recommend publicly available software to perform these tasks, much of which is collected on the CD-ROM which comes with the book.

The book is not without its faults, there are a couple of errors and mistakes (for example when talking about **/etc/fstab**, the authors state that they don't know anything that uses the fifth field, i.e., the dump frequency, where reference to the dump **man** page for the 'W' option describes it), and some bias shows through (such as in the description of the System V/ Solaris print spooler and Solaris System Access Facility), but this doesn't detract from the value of the book.

All in all, this book is a must have for any working System Administrator, and will become one of their most valuable reference books, whether for new projects, or in how to handle tricky user management issues.❖

The Internet Business Companion

by David Angell and Brent Heslop
Addison-Wesley
1994, 242 pages, $25.95
ISBN 0-201-40850-3
Reviewed by Giles Lean
<giles@nemeton.com.au>

This is another Internet book, subtitled 'Growing Your Business in the Electronic Age'. The back cover promises that it details the entire process of building a virtual business on the Internet.

The authors needed to add '... in the USA', since all the information on service providers and communications costs are based on American products and services.

I have difficulty deciding who the book is targeted to (besides Americans) as it ranges from trite business advice to a discussion of C class internet addresses, HTML writing and PC and Macintosh software for connection to the Internet. Much of this information is redundant—who uses UNIX, Macintoshes AND Windows?—and is dated already, since Netscape is not mentioned.

More useful than the information on access software is information on what WWW, gopher, ftp and email are and the ways businesses may use them. The authors explain the non–commercial origin of much of the Internet and encourage provision of information and services by WWW, ftp and mailing lists.

The discouragement of inappropriate use of newsgroups and direct junk email is most welcome. (Yes, Canter and Siegel get a mention. I think the estimate of 30,000 responses to their infamous spamming might be a bit low though...)

Security gets a passing mention when it is pointed out that you might want to setup a firewall, but does not get an entry in the index. From a business perspective security is usually very important and some security information should have been included,

UNIX is generally dismissed as being too expensive for small business users but is still given some coverage and recognised as being the server platform of choice ... at least until more NT solutions become available. PC based UNIX solutions are ignored.

In conclusion, this book has too much detail to be suitable to provide to a manger who wants to know what the Internet is all about, and not enough to be a good guide to setting up an Internet point of presence in Australia.❖

Applied UNIX Programming Volume 1

by Bharat Kurani
Prentice-Hall
1994, 516 Pages (plus diskette)
ISBN 0-13-304338-X
Reviewed by Greg Black
Greg Black & Associates
<gjb@gba.oz.au>

Anybody writing a book with a title like *Applied UNIX Programming* needs to take into account certain books which have covered that ground previously, such as the 1984 classic *The UNIX Programming Environment* by Brian Kernighan and Rob Pike or the 1992 contender *Advanced Programming in the UNIX Environment* by W. Richard Stevens. Moreover, if they also plan to cover the C and C++ programming languages in the same volume, they should consider the 1978 classic (and its 1988 second edition) *The C Programming Language* by Kernighan and Dennis Ritchie. It is, unfortunately, not at all clear that there is yet a worthwhile book on C++.

Although none of the prior art that I have mentioned is perfect, there are certain features that stand out in these books (two of which came from the same publisher as this book). These good points are: clarity of writing; clearly–defined purpose; meticulous typesetting; intelligent discussion and illustration of concepts; and careful selection of examples and exercises to complement the text. None of these virtues is visible in *Applied UNIX Programming*.

Instead, it is written in a turgid bureaucratic style that conceals what little purpose lurks within; the spelling is dismal (especially when it is clear that many of the errors would have been discovered by any spell checker); the typesetting is a disaster; the discussion is vague and unfocussed; and the examples and exercises are of little value.

There you have the essentials. Read on if you are interested in more detailed discussion of the flaws, but don't expect to find much praise to take the sting out of the criticism.

To start at the beginning, the book came with a diskette that was supposed to contain all the sample code. There were no indications, either on the diskette or in the book, about the media format or nature of the contents and nothing I tried managed to extract anything from the diskette. I doubt if this is a real tragedy, but it was the first apparent defect in the package.

A quick flip through the first few chapters revealed typesetting gaffes that have to be seen to be believed, together with page upon page of tedious material that should simply have been referenced in the (nonexistent) bibliography.

The typesetting problems are of two kinds. In many cases, there are great slabs of text (often extending over several pages) that appear in italics - apparently where some editing mistake removed a font change instruction that nobody ever bothered to fix. And, although much of the text is set (for no good reason) in courier, there is a 30 page slab of C source printed in the normal text font, which renders it completely unintelligible.

The first chapter is 27 pages of waffle about open systems in the past and the future, standards and standards organizations. Much of it reads as though it was lifted straight out of glossy marketing publications and has little real content.

Chapter 2 devotes 55 pages to UNIX, but much of this is a glossary of dubious value, lists of header files and several pages of partial contents of some of these headers, with no pedagogical purpose apparent. This is followed by a section on signals, with more slabs of text lifted from standards or draft standards and some sample code that flies in the face of common practice in writing signal handlers. And, despite the well-known variations in signal handling in the many UNIX-like systems, there is no discussion of these differences, nor even an explanation of what systems the sample code might be expected to suit.

Then there is a discussion of error numbers, which is little more than a listing of the relevant part from intro(2) running over a dozen pages, and again with no real indications of which systems these errno values apply to.

The UNIX chapter then ends with some exceedingly trivial exercises, although they probably suit the level of knowledge the chapter might have imparted.

Chapter 3 devotes 92 pages to software design and testing and is in many ways the most puzzling part of this strange book. A great deal of space is devoted to motherhood statements with no theoretical basis offered. And most of the rest consists of a bureaucratic nightmare that purports to offer a framework for software development - mostly achieved via a plethora of forms and documents, all of which are named and described in numbing repetition. Some of these things are called: Software Specification Form, Software Design Form, Software Coding Form, Software Testing Form, Software Release Form and Software User Form; and each Form has a corresponding Document as well.

All this bureaucracy eventually leads to the sample program, a partial implementation of the specification for a Motif–based dictionary lookup tool. But there is no discussion of X Window System programming, or of using Xlib, Xt, or Motif. And, despite the 30 page slab of illegible code, the only comments are idiotic ones that do things like point to the end of a five-line while loop.

The great triumph of this program is a switch statement with 135 cases and a default, in which every case consists of a call to `strcpy` to put a string into a char array. This gem takes up 10 pages on its own. And the part that eventually calls `printf` uses extra pointers to reference char arrays, a pointer called newline that points to a string with a newline in it and which is referenced by '%s' formats in the `printf` call. This is real beginner's coding, and it doesn't get any better.

I didn't really bother with Chapter 4 on internationalization, apart from noting that it had several pages of illustrations of text in various languages printed in a large ugly dot matrix font whose only virtue appears to be its ability to print in Chinese, Taiwanese, Japanese and Korean characters.

Chapter 5, which represents some 20% of the book, covers the C language in a manner that offers no competition to K&R. The author never makes clear which C language he is covering, despite all the discussion of standards in Chapter 1, and it begins badly with a list of reserved keywords that is neither fish nor fowl - it contains all the keywords from ISO Standard C, together with two obsolete words that were in K&R-1 but are no longer part of the language. And all the example code is written without prototypes and using the old-style function definitions, again without any explanation of these issues.

There is so much in the C section that is plain silly that I don't propose to attempt to give any kind of comprehensive list of the blunders - instead I will simply show some samples of the most egregious failings.

The program that illustrates global variables uses a global int variable called **i**. This does not represent any kind of sane style and should never be an example in a book with a goal of teaching solid programming techniques.

Many of the sample command lines for compilation, library building, and so on are very system-specific, but here (as elsewhere in this book) there are no hints about the need to check the real commands in the local environment.

Many sentences are not merely vague but actually content–free, perhaps because of careless editing. Here's a sample: "Some systems with languages and floating-point coprocessors support 64–bit integers and 128-bit reals, respectively."

None of the sample code shows the main function returning a useful value to the invocation environment, despite contrary long–standing practice (under UNIX at least). And the error functions that call exit all hand it a value of -1, which has never been a sensible value.

The discussion of the cast operator contains ridiculous examples and no coverage of the real reasons for using it - or the important reasons for avoiding it in most places in Standard C. The coverage of the binary logical operators makes no mention of short-circuit evaluation.

Functions are introduced with the bizarre claim that "Almost all C programs consist of function calls." Ignorant errors abound, such as the assertion that argc is always greater than 0. The presentation of call by reference is confusing because it does not adequately establish that C only has call by value.

The discussion of pointers is completely absurd (and opaque into the bargain). Kurani uses some home-grown pseudo- mathematical method in his vain attempts to explain how to assign values to pointers. I defy anybody who first meets C's pointers here to make any headway with them at all.

And so the sorry story draws to an end with a trivial set of pointless exercises. My initial astonishment that the author presented the complex X11 program of Chapter 3 without first introducing C was reduced somewhat by the discovery that the belated introduction to the language would in no way equip a reader to understand the earlier code.

The final chapter, on the C++ language, is the book's longest. I did not bother to read it, as it seemed that the rest of this book was so poor that any coverage of this overly complex and large language could not possibly be effective within the 185 pages allotted to it here.

The book ends with a single appendix, which is nothing more than two-thirds of the **ascii(7)** man page, and an inadequate index. Leave it on the shelf if you see it. And don't wait too eagerly for the second volume.❖

C++ FAQs

by Marshall P. Cline & Greg A. Lomow
Addison-Wesley Publishing Company
1994, Paperback, 461 pages (inc. index), RRP $34.95
ISBN: 0-201-58958-3
Reviewed by Greg Bond
<gnb@bby.com.au>

The word "FAQ" (an acronym for "Frequently Asked Question") is well-known amongst network users, but may be unfamiliar to book buyers. A FAQ is a list of questions often encountered on the net, each with a short answer. The intention is to reduce the noise level by answering all the beginner questions before they get asked. This substantial volume is produced by two familiar names from the Usenet newsgroup comp.lang.c++, where Cline edits and compiles the fairly weighty C++ FAQ. As the name implies, this book is a greatly expanded and enhanced version of this electronic FAQ.

There are 470 FAQs in this book, split into 43 chapters, each chapter focusing on one topic. The format of each FAQ is the same: about a page in length, with a question, a one-sentence short answer (in demi-bold type for clarity), then usually a couple of paragraphs of more in-depth answer, often with annotated code snippets. The final (and very useful) component of each FAQ is a cross-reference, not only to other FAQs in this book, but also to relevant sections from the three "must-have" C++ reference books: Lippman's *C++ Primer, 2nd Edition* (Addison-Wesley, 1991); Stroustrup's *The C++ Programming Language, 2nd Edition* (Addison-Wesley, 1991); and Ellis & Stroustrup's *The Annotated C++ Reference Manual,* usually known as "the ARM" (Addison-Wesley, 1990).

The book follows the electronic version in style as well as content. Rather than an academic tone, it retains the casual and informal language of the net, at times pithy, at times idiosyncratic, especially in the one-sentence answers:

Q283: Why does the compiler complain about >> when you use a template inside another template?
A283: Maximal munch.

Q189: How do you initialize one member object using an expression containing another member object?
A189: Just Say No.

Being a recent book, the C++ language covered is also quite contemporary. The book assumes the reader has access to (and familiarity with) C++ implementations with templates and exceptions. The bool type is assumed (but instructions presented for faking it), and `dynamic_cast<>` is mentioned only long enough to recommend against using it (or more accurately, against designs that need to use it). Neither are the remainder of RTTI, the other explicit casts, the mutable and explicit keywords, nor namespaces mentioned (except that one FAQ seems to use name exposure that is part of the namespaces facility without indicating that this is a new feature.)

In keeping with the authors' backgrounds as trainers in Object-Oriented methods, fully a third of the FAQs are directed towards questions about OO design and philosophy rather than the mechanics of C++. They have a very firm view of how C++ should be used and of how OO design should be done in a C++ environment, and many of the FAQs touch C++ itself only tangentially. This is a view born out of constructing large software systems, in a large team environment, so is somewhat foreign to people such as me who have never worked on large team projects. Thankfully, in a book format, they can put forward such an approach without evoking the endless followups and flamewars about the One True Way of OO Design that so characterises the Usenet newsgroup.

The FAQs dealing with C++ itself are varied and cover a range of issues, from the obvious to the obscure. Topics to receive special coverage include pointers and free store, derivation and inheritance, exceptions, and templates - all topics that get a regular airing on the newsgroup. There are a number of "Dorothy Dixers" that strain the FAQ format somewhat in order to introduce a topic, and the OO design questions in particular are seldom seen on the net (stretching somewhat the definition of "Frequent"!)

Would I buy the book? Well, that depends.

It is certainly not an introduction to C++. You will need a reasonable overview of C++ before much of the book will make sense, so for a first book I would instead recommend something like the Lippman book. Nor is it a definitive or complete reference, for there are many areas left uncovered. For that, Stroustrup is a much better choice for the average reader, although the ARM is vital for the *really* nuts-&- bolts details. Nor is it particularly deep or broad in scope, and experienced C++/OO designers will probably find little of use.

Where this volume probably has a role is as a second C++ book, and a convenient and approachable pointer to other references. Having covered most of the mechanical details of the language with an introductory text such as Lippman, this book (like the electronic FAQ) makes a good launching point for follow-up queries. Provided your question is (close to) one that is in the book, it is a very convenient and concise summary with very valuable pointers to more complete or definitive answers. And for a casual C++ user the format of small, bite-sized pieces might make suitable bedtime reading.

It has value also as an introduction to the mindset of OO design that C++ both encourages and demands for best use. Being a novice OO programmer with reasonable experience in using C++ as "a better C", I found this aspect the most helpful. One area where it could be stronger is in explaining the common C++/ OO idioms, where it could again stand between new C++ programmers and more advanced texts such as Coplien's *Advanced C++ Programming Styles and Idioms* (Addison-Wesley, 1992), for example.

C++ FAQs, then, sits somewhere in the middle: not the first book I would buy, nor the last, but not a bad choice for a second book, useful for a short time, another step on the ladder to mastery of C++.❖

Advanced X Window Applications Programming, Second Edition

by Eric Johnson and Kevin Reichard
M&T Books
1994, 548 pages + CD-ROM, $89.95
ISBN 1-55828-344-7

Reviewed by Michael Haldey
Genasys II Pty Ltd
<michaelh@genasys.com.au>

Kevin Reichard and Eric Johnson are experienced authors and programmers who have published several books on various aspects of X Window and Motif programming and administration. The second edition of *Advanced X Window Applications*

Programming is updated to conform to the X Window Release 6 and comes with a CD-ROM that contains a source code from the book as well as the Release 6 source code and the contributed software.

The book is about X Window programming using Xlib. It provides detailed coverage of the strategies for X program- to-program communications, working with Window and Session managers and multiple display connections. If you are interested in these aspects of X programming the book is definitely worth looking at. If not - it is probably not a book for you - other aspects of Xlib programming are covered fairly briefly.

The book is divided into 20 chapters grouped into five Sections. The first Section is a brief introduction into X programming including working with X and Motif toolkits. The authors say that even an experienced X programmer should read it but I feel it can be easily skipped by anyone who programs using Xlib and has read any book on the subject. A number of examples are introduced to illustrate the concepts and are be used later when developing a simple toolkit. The Section is a very good reading for a professional programmer who is new to X but has some experience with other GUI environments.

The second Section is the 'meat' of the book. This is where the book accomplishes its purpose. It contains six chapters (from 10 to 15):

- In Chapter 10 a simple illustrative X toolkit is built; it is used to make further example programs smaller and easier to comprehend. (I wonder why this code is not presented in the Section I.) There is also an overview of the X program-to-program communication methods.
- Chapter 11 explains and illustrates the use of Properties and Atoms.
- Chapter 12 discusses various aspects of sending X events between two applications, grabbing the mouse pointer, translating coordinates and sending strings.
- Chapter 13 is a detailed guide on using selections. Using a small program the use of selections by the real X applications (like xterm) is explored.
- Chapter 14 covers working with Window managers (Motif and Open Look) and the rules set by Inter-Client Communications Conventions Manual (ICCCM).
- Chapter 15 deals with the Session management. It discusses the Release 5 (old) and Release 6 (new) ways of communicating with Session managers. Special attention is paid to the usage of the UNIX select call as well as new session management and inter client exchange libraries that come with Release 6.

Section III deals with multiple display connections. The authors create a small program - machine to machine chat - and use it to explain the conceptions and details. Then they briefly cover a number of issues when working with Multiple displays such as security, handling errors and writing to properties on multiple root windows. The section is very useful and well written.

Section IV is a brief review of the Shape X extension (non- rectangular windows) and the XIE (X image extension). The last Section is an overview of the new features of the X11 Release 6.

The Appendices provide the book list on the subjects, functions developed in the book and the CD-ROM contents. The book list is very poor and it looks like the authors really dislike the books published by O'Reilly and Associates—they have forgotten to mention any of them devoted to X and Motif programming.

The book is written in easy-to-read style, the code example are numerous, well balanced and not bulky. The book is not a reference but it quickly provides a clear conceptional understanding of the issues. Sometimes you really want to know the details; unfortunately the authors are not very careful in referencing even within the book itself. Be prepared to use the index often and use the Xlib Programming Manual by Adrian Nye (published by O'Reilly & Associates) if the answer is not found.

Overall the book is a very good buy for an experienced programmer new to Xlib programming. An experienced Xlib programmer would still find some good recipes, useful examples and tips.❖

The Underground Guide to UNIX: Slightly Askew Advice from a UNIX Guru

by John Montgomery
Addison Wesley Publishing Company
December 1994, 338 pages, $34.95
ISBN 0-201-40653-5

Reviewed by Wayne Bell
<wayneb@sydney.dialix.oz.au>

The Underground Guide to UNIX is an introductory guide to UNIX. The guide is of jokes, puns, asides and famous quotes. Although all necessary introductory areas are covered adequately, it's value will rely on the reader's opinion of the 'askew' content. Although probably not the best choice of introductory text for an organisation to buy, individuals may enjoy it.

The Underground Guide to UNIX describes itself in the introduction as "...that's what this book is about, using standard, old-fashioned UNIX commands in new (or somewhat new) ways. It assumes you already know something about UNIX, and are interested in UNIX ... You'll find serious information on getting the most out of the parts of UNIX you use everyday".

The book then, is an introduction to UNIX for new users. There are chapters on Basics, Shells, Issuing Commands, Files, Editing Files, Shell Programming, e–mail and Networking, the Internet, Common Problems, Editor Summary, Useful Perl Scripts and Security; most of the useful stuff a newcomer to UNIX would need. The tone of the book is light, chatty and full of puns - I mean *FULL* of puns, lots of dumb jokes, famous quotes and quirky asides and anecdotes. I can accept that an introduction to UNIX should not be too technical or 'heavy', but for mine this has too many jokes and puns. On some pages, fully one third of the content is jokes, puns anecdotes, quotes, and miscellanea. The six people I showed this book to all agreed that it contained too many dumb puns and jokes.

Appendix A - Most useful commands is probably the worst example. Each command is followed by a table of switches and what they do, but each command is inserted into a pun, e.g., 'On the mv' 'go to your bedrm', 'don't be DIFFicult', 'What SORT are you anyway', 'Su- oooey'. AAAAAAAAh! Everyone who saw this groaned, and put the book down.

Another issue with this book is the length, and material covered. At 338 pages, if the jokes are taken out, we are down to about 250 pages. A really useful guide to UNIX that will be used for more than a few months needs to be much longer than this. The information is useful as an introduction, but I doubt it will stay indispensable for long. The book does not cover topics deeply enough, for example, **awk** is covered in five pages, down to four once the jokes are removed. I accept that **awk** cannot be fully covered in an introductory text, but **awk** is covered here as a smart **grep**; once the command line gets over one line, the author gives up. The larger introductory texts cover **awk** in more depth than this, showing how **awk** programs can be built, which remain useful as references for years to come. I feel the underground guide will not be referred to after the first year of using UNIX.

The technical content is okay—I could not find any outright errors—but I spent 10 minutes looking and found three inaccuracies:

- Su instead of su,
- sed not being able to substitute a newline, - try using tr, piped to sed if necessary, the tool approach to UNIX, one command can't do everything, and,
- A paragraph long fume about the author forgetting to set binary mode in ftp, as if this is ftp's fault - try always using binary.

The sub-title of the book is 'slightly askew advice from a UNIX Guru', but throughout the book the author's tone is that of a newbie, reporting on what one finds after a few months with UNIX, but not trying to explain technical reasons, and confessing to not reading manual pages; hardly a Guru.

In summary then, the *Underground guide to UNIX* is an introductory guide that covers the main parts of UNIX acceptably well. The author seems unwilling to cover aspects that involve any programming, or complexities. Topics are not covered in enough depth to be really useful as a long term reference. It's loaded with paragraphs of jokes, puns, asides, anecdotes, quotes, etc., that some readers may appreciate, but annoyed me as well as the people I showed the book to. This title's worth a look if you are new to UNIX, only want to know enough to get around, and don't like programming or complications – and enjoy bad puns!❖

Managing Internet Information Services

by Cricket Liu, Jerry Peek, Russ Jones, Bryan Buus, & Adrian Nye
O'Reilly & Associates, Inc.
1994, 670 pages, $59.95
ISBN 1-56592-062-7

Reviewed by Lawrie Brown
Australian Defence Force Academy
<Lawrie.Brown@adfa.oz.au>

This book is another massive (some 670 pages) tome, packed with useful information and up to the high standards we expect from O'Reilly. It deals with the myriad issues that arise with installing and managing Internet Information Services; just about all of them; you name it, this book probably talks about it.

One of this book's greatest strengths is its clear distinction between the roles of Systems Administrator, responsible for the installation and management of the software; and Data Librarian, responsible for the creation and maintenance of the information.

One of the greatest concerns, and usually greatest failings of many current Internet information sources, is the lack of timeliness of the information provided. It is one thing to get a service up and running, it is another, much greater task, to ensure that the information provided remains correct and timely. In its clear delineation and discussion of these roles, always flagged distinctly, and often to the extent of separate chapters on each for the various services, this book does a great service which may well help improve the current situation. For this aspect alone, this book would be an invaluable addition to the library of any service provider.

The sections for the Data Librarians are well written, using the necessary minimum of technical systems content (primarily basic UNIX file creation and manipulation), which makes them all the more appropriate. Our local librarians have been studying some of these sections to assist them in the creation and maintenance of our new electronic library services, and they've found them most useful. In contrast, the Systems Administrator sections do assume a basic UNIX administrators background, and some familiarity with network configuration and services. However the book's introductory chapters 1 & 2 do revise this material. The remainder of the book walks through the compilation, installation, configuration and management processes for the various services and their variants.

Chapter 3 describes some simple information services provided using finger, inetd, or telnet. I rather liked the suggested use of FIFOs to provide a means of dynamically updating the `finger` **.plan** file each time it is queried. The use of **cat** as an **inetd** command for special services, or making client programs or menu systems shells for telnet were also described.

Chapters 4 to 6 deal with **ftp**, describing the installation and configuration of both the standard **ftpd** and the **wu-ftpd**. It includes a good discussion of the intricacies and pitfalls of creating a chrooted anonymous ftp area, as well as descriptions of lesser known features such as the sublogins and local groups for finer access control to `chrooted` **ftp** areas. Chapter 6 deals with the data management, including details on readme and message files, and the provision of ls listing files.

Chapters 7 and 8 describe WAIS, using the freeWAIS system. It covers compilation, configuration, and management, as well as describing the management of WAIS databases, with a discussion of the various document types supported, and how to appropriately index them.

Chapters 9 through 16 describe **gopher** and friends (**Veronica** and **Jughead**). The size of this section is obviously a reflection of the authors interests, and reflects the period when the book was written. It very comprehensively details the compilation and installation of both the free gopher, and the commercial **gopher+**, servers from the University of Minnesota. It then continues with a description of the creation of a gopher information tree, menu customisation, linking to other gopher information, inclusion of WAIS or NeXT indexes, and the use of scripts for customised indexing or dynamic document creation. As an example it showed how to query an apropos database, returning the results as links to the appropriate man pages. Next it discusses **Veronica** and **Jughead**, which provide indexes to all the **gopher** servers, or for your local server, and how to link to them. Finally it describes a number of major new features in **gopher+**, including forms support and their associated scripts.

Chapters 17 to 21 deal with the World Wide Web, in particular with the NCSA **httpd** server. It starts with a general introduction to web concepts, html, URLs, the http protocol, and mentions some of the servers available such as the NCSA, CERN and Plexus servers. Following this is a detailed description of the installation, configuration and management of the NCSA server. Next is a chapter on authoring for the web, with a good discussion of some of the issues with information design and the choice of suitable styles for documents.

Some of the authoring tools available are mentioned, along with an overview on creating clickable image maps. Then the use of gateway scripts to access information and return dynamically created documents is described, along with the creation of forms and their associated scripts. It includes some sample **perl** scripts for running simple searches on files.

There is a good description of some of the security implications of using scripts, and how input values should be validated. Various web to WAIS interfaces are then described and compared. Finally this section concludes with a look at access control mechanisms (in the NCSA server) and further discussion of some security issues. I found this whole section to be well laid out, and whilst the lack of detail on the CERN server was regrettable, much of the material is still applicable, even if modified a little. The detail on the creation of gateway scripts and forms is good, as this area is not well understood or widely documented.

Chapters 22 to 26 deal with email based information services such as file servers and mailing lists. Why use email services at all is discussed, basically since they provide the widest possible audience using probably the most fundamental network access tool. A description of a simple file server, based on a perl script `canned_reply` which returns one file per mail alias, is then given. Next the book moves on to mailing lists, with a comparison of approaches based on simple (alias) lists, majordomo and listproc (not discussed in detail). There is a chapter on simple mailing lists using the alias mechanism, with some comments on error handling, header rewriting, and the use of local exploders. Following this are 2 chapters on installing, configuring, using and managing lists with majordomo. These chapters appear pretty detailed, though not having any experience with majordomo I can't comment further. Finally there is a description of installing and using the **ftpmail** server.

The book concludes with some chapters on miscellaneous topics. First is an overview of firewalls, their interaction with information services, and some suggestions for where to locate servers, and how to relay client requests through firewalls. Next is a chapter on **xinetd**, an enhanced inetd server. Finally there are a couple of chapters on legal issues and intellectual property rights, laying out some of the problems, issues, and responsibilities. The book concludes with a number of Appendices describing various server options and configuration file formats.

Taking a deep breath after such an immense volume of information, some final reactions. Basically I'm impressed. I really wish I had this book 6 months ago, before I embarked on the installation of the ftp, gopher and web services on the AUUG (Canberra) and PCUG Internet project. Even so I've found several useful hints that I've been able to apply since. Its greatest weakness for me is definitely the lack of coverage of the cern **httpd** and its use as a proxy caching server. This is virtually essential for web service providers today. Its lack, as well as the heavy emphasis on gopher, is just symptomatic of the rapidity of developments in this field, and one I'm sure will be corrected in the next edition. Nonetheless, given its very many strengths, I have no hesitation in recommending this book for any current or potential Internet Information service provider.❖

AUUGN IS **YOUR** JOURNAL!

Without you, there is no AUUGN: if you've knowledge to share, share it through AUUG's bimonthly journal.

You'll be reaching over 700 individuals, and more than 300 organisations involved in the UNIX/ Open Systems world.

WE'RE LOOKING FOR:

Talk to your local Chapter contact for ideas, and see elsewhere in this issue for submission guidelines.

- Papers
- Reviews
- Articles
- News
- Comment

The Underground Guides

20% Discount to AUUG Members

The Underground Guide to UNIX

John Montgomery

It's no coincidence that UNIX is a four-letter word — the problem child of operating systems is notoriously arcane and inscrutable. In the **Underground Guide to UNIX**, scrutability expert John Montgomery takes you on an emotional roller-coaster ride through the vagaries of this hard-to-master and incomprehensible operating system. You'll find serious information on getting the most out of the parts of UNIX you use every day, as well as detailed advice on working better and faster, whatever flavour of UNIX you're stuck with... uh, prefer. Learn how to: Master the most popular unix text editors: vi, emacs, ex and sed; use and abuse basic unix security - hide potentially incriminating files like love.letter or resume.doc, or search your boss's files for your name and key words like problem employee; access the Internet and use ftp, telnet, gopher and Mosaic to get completely frivolous information from almost anywhere in the world; program the shells (C, Korn, Bourne) to do whatever you want. Every page has something you can use immediately. This book is packed wall to wall with advice, warnings, tips, bug reports, workarounds, and the kind of nitty-gritty explanations that could come only from someone who eats, sleeps and breathes UNIX.

CODE 0-201-40653-5 RRP $34.95 AUUG Price $27.95

For more slightly askew advice, try these other Underground Guides...

The Underground Guide to Word for Windows, *Leonhard*	(Code 0-201-40650-0)	$30.95
The Underground Guide to Excel for Windows, *Hudspeth & Lee*	(Code 0-201-40651-9)	$29.95

✓ ☐ *YES: I would like to order:* QTY: ______ TITLE: ______________________

QTY: ______ TITLE: ______________________ QTY: ______ TITLE: ______________________

☐ *I enclose a cheque for* $ ________ OR ☐ *Please charge my credit card No:* ______________________

☐ BANKCARD ☐ VISA ☐ MASTERCARD ☐ AMERICAN EXPRESS ➪ ID NO: ______________

EXPIRY DATE: / / AUUG MEMBERSHIP NO: ______________ PHONE NO: ______________ (WK HRS)

NAME: ______________________________ SIGNATURE: ______________________

STREET ADDRESS: ____

____ STATE: ________ POST CODE: ________

Send your order to ➤

Addison-Wesley Publishing Company

Unit A1, 6 Byfield Street, North Ryde, NSW 2113, Australia.
Telephone (02) 878 5411 Customer Service Fax (02) 888 9404

AUUG Business

UNIX & OPEN SYSTEMS USERS

AUUG Annual General Meeting: Call for AGM Agenda Items.

Peter Wishart <peter.wishart@auug.org.au>
Secretary, AUUG Inc.
Tel: (06) 261 2997
Fax: (06) 261 3806

AUUG Inc. will be holding the 1995 Annual General Meeting at AUUG95 in Sydney, Sept. 17-21. Members will be advised of the exact date and time, location and agenda in writing at least four (4) weeks before the meeting. We are now calling for suggestions from the membership for agenda items for the AGM.

Members are reminded of the provisions in the constitution which constrain discussion at the AGM to items on the agenda. The relevant section of the constitution is shown below.

> 15(2) Business conducted at such meetings shall be confined to matters included in the written agenda, reports from Officers, and resolutions instructing the Management Committee to conduct a formal ballot of the membership on matters of substance. Such resolutions shall not be binding on the Management Committee unless the meeting was attended by at least twenty voting members, or half the membership, which- ever is less, and the resolution was supported by at least three-quarters of the members voting.

Any member who wishes to have an item added to the agenda for the AGM should provide details of the item to the Secretary by **1st August 1995**.

AUUGN Submission Guidelines

Please submit all articles, comment and notices for publication in AUUGN in the following forms:

Text

- ASCII format
- Text left-aligned, and unjustified
- Blank line between paragraphs
- If used, please put footnotes, notes and bibliography at the end of the file

Images

- Please provide images, diagrams and illustrations as separate files
- Make sure that your text contribution indicates where graphics should go in the contribution
- We can read/convert most graphics formats; make sure that if providing images as Postscript, that you use Encapsulated Postscript (EPS) format
- If the above is not possible, small hardcopy illustrations are acceptable

Where to send your submissions

- E-mail your submission to **auugn@auug.org.au** with AUUGN submission in the Subject line, or
- Post it on a disk (DOS formatted is preferable, though both MAC and UNIX disks are also accepted) to

  AUUGN Submissions
  PO Box 366
  Kensington NSW 2033

AUUG Institutional Members

as at 30/03/95

AAII
Aberfoyle Resource Limited
ACAY Network Computing Pty.Ltd.
Actrol Parts
Adept Software
Advanced Software Engineering
Alcatel Australia
Amalgamated Television Services
Amdahl Australia Pty Ltd
Amdahl Pacific Services
Andersen Consulting
ANI Manufacturing Group
Animal Logic Research Pty. Ltd.
Ansett Australia
ANSTO
Anti-Cancer Council of Victoria
ANZ McCaughan
AT&T GIS
Atlas Computer Systems
Attorney-General's Department
Ausnet Services Pty. Ltd.
Australian Archives
Australian Bureau of Statistics
Australian Centre for Remote Sensing (ACRES)
Australian Defence Industries Ltd.
Australian Electoral Commission
Australian Film Television and Radio School
Australian Information Processing Centre Pty. Ltd.
Australian Medical Enterprise
Australian Museum
Australian National Audit Office
Australian National University
Australian Submarine Corporation
Australian Taxation Office
Australian Technology Resources (ACT) Pty. Ltd.
Australian Technology Resources Pty. Ltd.
Australian Tourist Commission
Australian Wool Research & Promotion Organisation
B & D Australia
Bartwon Water
Bay Technologies Pty Ltd
BHA Computer Pty. Limited
BHP Information Technology
BHP Minerals Exploration
BHP Petroleum
BHP Research - Melbourne Laboratories
BHP Research - Newcastle Laboratories
Bond University
Burdett, Buckeridge & Young Ltd.
Bureau of Meteorology
Butterworths
Bytecraft Pty. Ltd.
C.I.S.R.A.
Cadcom Solutions Pty. Ltd.
Cape Grim B.A.P.S
Capricorn Coal Management Pty. Ltd.
CelsiusTech Australia
Central Queensland University
Centre for Open Systems Pty. Ltd.
Chief Secretary's Department
CITEC
Clarity International
Classified Consulting Pty. Ltd.
Clegg Driscoll Consultants Pty. Ltd.
Co-Cam Computer Group
Coal & Allied Operations
Cognos Pty. Ltd.
Com Net Solutions
Com Tech Communications
Comcare Australia
Commercial Dynamics
Commercial Industrial Computer Services Pty. Ltd.
Communica Software Consultants
Composite Buyers Ltd.
Computechnics Pty. Ltd.
Computer Sciences of Australia Pty. Ltd.
Computer Systems (Australia) Pty. Ltd.
Compuware Asia-Pacific
Comsys International Pty. Ltd.
Concord Repatriation General Hospital
Continuum Australia
Copper Refineries Pty. Ltd.
Corinthian Engineering Pty. Ltd.
CSIRO Division of Information Technology
CSIRO Division of Manufacturing Technology
Curtin University of Technology
Cyberdyne Systems Corporation Pty. Ltd.
Cyberscience Corporation Pty. Ltd.
Cybersource Pty. Ltd.
Daedalus Integration Pty. Ltd.
Datacraft Limited
Datacraft Technologies
Dawn Technologies
DB Bain Group Services Pty. Ltd.
Deakin University
Defence Housing Authority
Defence Service Homes
Department of Business & Employment
Department of Communications and the Arts
Department of Conservation & Natural Resources
Department of Defence
Department of Defence (TC Section)
Department of Education QLD
Department of Environment & Natural Resources
Department of Family Services & Aboriginal & Islander Affairs
Department of Lands, Housing & Local Government
Department of Public Administration
Department of State Services
Department of the Treasury
Dept. of Industrial Relations, Employment, Training & Further Education
DEVETIR
Dialix
Digital Equipment Corp. (Australia) Pty. Ltd.
Dominion Systems Pty. Ltd.
DSTO, Lab 73
EASAMS (Australia) Limited
Edith Cowan University
Electricity Trust of South Australia
Electro Optics Pty. Ltd.
Engineering Computer Services Pty. Ltd.
Environmental Resources Information Network (ERIN)
Equity Systems Pty. Limited
Ericsson Australia
ESRI Australia Pty. Ltd.
Execom Consulting
Executive Computing
FGH Decision Support Systems Pty. Ltd.
Financial Network Services
Fire Fighting Enterprises
First State Computing
Flinders University
Fremantle Port Authority
Fujitsu Australia Ltd.
GEC Alsthom Information Technology
GEC Marconi Systems Ltd.
Genasys II Pty. Ltd.
General Automation Pty. Ltd.
GIO Australia
Golden Circle Limited
Great Barrier Reef Marine Park Authority
Haltek Pty. Ltd.
Hamersley Iron
Hannan Group Computer Services
Hansen Corporation Pty. Ltd.
Heath Insurance
Hermes Precisa Australia Pty. Ltd.
Hitachi Data Systems
Honeywell Australia Ltd.
Honeywell Ltd.
Hong Kong Jockey Club Systems (Australia) Pty. Ltd.
I.P.S Radio & Space Services
IBA Healthcare Pty. Ltd.
IBM Australia Ltd.
Ideas International Pty. Ltd.
Independent Systems Integrators
Informatel Online
Information Technology Consultants
Informed Technology
Insession Labs Pty. Ltd.
Insurance & Superannuation Commission
Intelligent Network Development
Internode Systems Pty. Ltd.
ISR Group Ltd.
James Cook University of North Queensland
Joint House Department
JTEC Pty. Ltd.
Keays Software
Knowledge Engineering Pty. Ltd.

Laboratory Systems Pty. Ltd.
Labtam Australia Pty. Ltd.
Land Information Centre
Land Titles Office
Leeds & Northrup Australia Pty. Limited
Legent Australia Pty. Ltd.
Logica Pty. Ltd.
Lotus Development
Lyons Computer Pty. Ltd.
Macquarie University
Main Roads Western Australia
Maintain Axis Computers
Matcom Technologies
Mayne Nickless Courier Systems
Mayne Nickless Information Tech. Services
Memtec Limited
Mentor Technologies Pty. Ltd.
Mercedes-Benz (Australia)
Message Handling Systems
Metal Trades Industry Association
Mincom Pty. Ltd.
Minenco Pty. Ltd.
Mitsubishi Motors Australia Ltd.
Mitsui Computer Limited
Moldflow Pty. Ltd.
Motorola Communications Australia
Motorola Computer Systems
MPA International Pty. Ltd.
MUA Pty. Ltd.
Multibase Pty. Ltd.
Multiline BBS
Multiuser Solutions Pty. Ltd.
National Library of Australia
National Resource Information Centre
NCOM Services
NEC Australia Pty. Ltd.
Northern Territory Library Service
Novell Pty. Ltd.
NSW Agriculture
NSW Public Works, Information Services
NSW Teachers Federation Health Society
Object Design Pty. Ltd.
Object Oriented Pty. Ltd.
Object Technology International Pty. Ltd.
Office of State Revenue
Office of the Director of Public Prosecutions
ONA
Open Software Associates Ltd.
Opentec Pty Ltd
OPSM
OSIX Pty. Ltd.
OzWare Developments Pty. Ltd.
Pacific Semiconductor Pty. Ltd.
Pacific Star Communications
Peter Harding & Associates Pty. Ltd.
Petrosys Pty. Ltd.
Philips PTS
Port of Melbourne Authority
Powerhouse Museum
PPIT Pty. Ltd.
Primary Industries & Energy
Process Software Solutions Pty. Ltd.
Prospect Electricity
pTizan Computer Services Pty. Ltd.
Pyramid Data Centre Systems
Qantek
QLD Department of Transport
QLD Electricity Commission
Quality Bakers Pty. Ltd.
Quality By Design Pty. Ltd.
Redland Shire Council
Rehabilitation Tasmania
Renison Golfields Consolidated Ltd.
Rinbina Pty. Ltd.
Royal Melbourne Institute of Technology
SCEGGS Redlands Ltd
Sculptor 4GL+SQL
Security Mailing Services
SEQEB Business Systems
Siemens Nixdorf Information Systems Pty. Ltd.
Smallworld Systems (Aust.) Pty. Ltd.
Smorgon ARC
Snowy Mountains Authority
SoftGen Pacific Pty. Ltd.
Software Plus (Australia) Pty. Ltd.
Softway Pty. Ltd.
South Australian Co-operative Bulk Handling
Specialix Pty. Ltd.
St. Gregory's Armenian School
St. John of God Health Care System
St. Vincent's Private Hospital
Stallion Technologies Pty. Ltd.
Standards Australia
State Revenue Office
Steelmark Eagle & Globe
Sterling Software
Storage Technology of Australia
Strategic Information Technologies Pty. Ltd.
Sunburst Regency Foods Pty. Ltd.
Sydney Electricity
Sydney Ports Authority
Systek Corporation Pty. Ltd.
System Builder Development Pty. Ltd.
Systems and Management Pty Ltd
Systems Development Telecom Australia
TAB of Queensland
TAFE NSW, Information Systems Division
Tandem Computers
Technical Software Services
TechNIX Consulting Group International
Telecom Australia
Telecom Australia Corporate Customer
Telecom Payphone Services
Telecom Research Laboratories
The Far North QLD Electricity Board
The Fulcrum Consulting Group
The Knowledge Group Pty Ltd
The Preston Group
The Roads & Traffic Authority
The Southport School
The University of Western Australia
Thiess Contractors Pty. Ltd.
Thomas Cook Ltd.
TNT Australia Information Technology
Toshiba International Corporation Pty. Ltd.
Tower Technology Pty. Ltd.
Tradelink Plumbing Supplies Centres
Transport Accident Commission
Triad Software Pty. Ltd.
Turbosoft Pty. Ltd.
TUSC Computer Systems
Unidata Australia
Uninet Consulting Pty. Ltd.
Unisys Australia Ltd.
University of Adelaide
University of Melbourne
University of New England
University of New South Wales
University of Queensland
University of South Australia
University of Sydney
University of Tasmania
University of Technology, Sydney
Unixpac Pty. Ltd.
Vanguard Computer Services Pty. Ltd.
Vanoco Pty. Ltd.
Victoria University of Technology
VME Systems Pty. Ltd.
Walter & Eliza Hall Institute
Wang Australia Pty. Ltd.
Water Board
WCS Australia Pty. Ltd.
Wesfarmers Limited
Western Mining Corporation
Westrail
Woodside Offshore Petroleum
Workers' Compensation Board of QLD
Workstations Plus
XEDOC Software Development Pty. Ltd.

Is your organisation an Institutional member of AUUG?

Institutional membership gives your organisation: a vote in all AUUG elections & ballots; membership rates to all AUUG national and chapter activities for the primary contact and up to two additional representatives; and *two* copies

STANDARDS

An Update on Standards Relevant to USENIX Members

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

The following reports are published in this column:

- POSIX.1:System Interface
- POSIX.4: Realtime Interfaces
- A View from the Chair

Our Standards Report Editor, **Nick Stoughton**, welcomes dialogue between this column and you, the readers. Please send your comments to *<nick@usenix.org>*.

Report on POSIX.1: System Interface

Shravan Pargal <pargal@sdiv.cray.com> reports on the October 17-21, 1994 meeting in Seattle, WA:

This was my first POSIX meeting. Given this fact, my perspective on the proceedings of the meetings attended should be viewed with a large pinch of salt. This report is not necessarily factually accurate or binding in any way.

The POSIX.1 working group (WG) was small, with four seniors and two freshmen (including me) present the first day. Both of us freshmen were accepted warmly, so there were two less freshmen on the last day of the meeting. The WG was now growing – proof of life and success!

The Seattle WG meeting was dominated by outstanding interpretations and ballot issues on Draft 11 of the POSIX.1a ballot. While other issues were discussed, the ballot status and issues surrounding Checkpoint/Restart seemed to keep resurfacing. Issues discussed during the week included:

- Trailing Slashes
- Application conformance
- Optional environment functions like *walkenv*(), *savenv*(), and *loadenv*()
- Checkpoint/Restart
- Removable Media Study Group

Trailing slashes

The problem with the current standard is lack of clarity in the definition of pathname in the case that the pathname contains one or more trailing slashes.

It was decided that this is really a problem with the definition of "pathname resolution," and not with trailing slashes in particular. Thus the solution lay in clarifying the definition of "pathname resolution."

A proposal was put forth to tighten the requirement on pathnames ending in trailing slashes to allow them only in the case of existing directories – the behavior of BSD based systems. The definition of pathname resolution will now include something like the following:

"A pathname that ends with one or more trailing slashes shall be interpreted as if it ended with slash-dot, except that the slash-dot does not count in {PATH_MAX} calculations."

Application conformance

Two points were proposed here. First, the need for an additional definition to be added to POSIX.1 from C language – a definition that would allow for an application to conform to both POSIX.1 and to ISO 9899 C. Second, a plea for consistency in the wording of definitions in the different POSIX standards – at least those that relate to conformance issues.

The additional definition was that of a "Rigorously Conforming POSIX.1 C Application." This is defined as an application that requires the services and facilities only described in ISO/IEC 9945-1 and ISO/IEC 9899. Furthermore, with regard to the use of the C language, it shall not produce output dependent on any unspecified, undefined, or implementation-defined behavior, and shall not exceed any minimum implementation limits, as defined in ISO 9899.

The second point was about consistency in wording referred at least in part to the difference in the definitions of Strictly Conforming POSIX.1 and POSIX.2 Applications.

The language above was taken from the "Application conformance issues" paper by Derek Jones of Knowledge Software. This is not a complete definition of the "Rigorously Conforming POSIX.1 C Application." The complete definition can be got directly from the paper.

Optional environment functions

There was a discussion of the environmental functions (*xxxenv*()) in POSIX.1a. The WG felt that such functions were probably useful in threaded environments, and should thus probably be optional. There was some discussion about *walkenv*() actually making things worse for threaded environments. The final decision was to discuss at the next meeting if the functions should be optional and if there could be an alternative function to *walkenv*() that would be thread-safe. [*Subsequent telephone meetings of the WG have decided to remove these contentious functions from the next draft of POSIX.1a. Ed.*]

Checkpoint/Restart

There was a joint meeting with the Fault Tolerant (SRASS) group and the Super Computing Profile (POSIX.10) groups about checkpoint/restart. There are still a lot of conflicting goals for this. The POSIX.1 WG would like to see a useful minimal interface specification, something that can be implemented and used by a reasonable set of applications. The SRASS group would like to see the name changed from checkpoint/restart to something like "batch services," since the functions do not go far enough for their particular needs, and were invented originally within the batch profile.

The POSIX.1 WG would like to see a common name agreeable to all groups. It will oppose the possibility of two similar but different interfaces coming into existence for checkpoint/restart.

Removable Media Study Group

This group discussed issues associated with tapes (generally true for any removable media) and how work supporting it might proceed. They expect to submit a Project Authorization Request (PAR) in January describing the scope and goals of the group.

The next meeting will be in sunny Ft. Lauderdale FL, January 15-21, 1995. I enjoyed the last meeting enough to be planning on going to Fort Lauderdale (well, it is Florida in January).

Report on POSIX.4: Realtime Interfaces

Joe Gwinn <GWINN@sud2.ed.ray.com> reports on the October 17-21, 1994 meeting in Seattle, WA:

Why are pthreads taking so long?

As many of you may have heard, final approval of the pthreads draft POSIX standard (was called P1003.4a, now renamed P1003.1c by the IEEE) was delayed for six months by a negative coordination ballot from the POSIX.5 Working Group (the Ada folk). The ensuing dispute finally bubbled up to the SEC (Sponsor Executive Committee, the governing body of POSIX), who at the POSIX.5 Working Group's request stalled POSIX.1c and sent the two Working Groups (POSIX.5 and POSIX.4) off to come to a fuller understanding of each other's position and issues, in the hope that the problems would then be resolved.

After three months of weekly telephone conferences and much email, the issues were indeed clarified, but the core differences remained irreconcilable. At the next SEC meeting, in July 1994, the POSIX.5 WG asked for further delay and also that the P1003.1c Balloting Group be queried by letter, while the POSIX.4 WG asked that the stall be rescinded and the matter closed. The SEC accepted POSIX.5's request for a query, but voted to limit the stall to 3 more months, to allow time for the query responses to be received.

The results of the survey of the pthreads balloting group was unequivocal. If the change requested by the POSIX.5 WG was incorporated into P1003.1c, approval would drop from 82% to something like 60%, way below the required 75%. If the change was not incorporated, approval would drop by less than one percent. A total of 186 balloters responded yes or no to the query. To state the result baldly, if the changes requested by the Ada folk were accepted, pthreads would be dead, and we would have to start over.

At the October 1994 POSIX meeting, the SEC voted to allow P1003.1c to resume progress towards standardization, and to allow the POSIX.5 WG's objections to accompany it.

(Because standards require consensus, not unanimity, the majority of standards have unresolved objections.) The next step is CD Registration and concurrent balloting at ISO, followed by presentation to the IEEE Standards Board. This will take some months, but further changes to P1003.1c (now at draft 10) are likely to be minimal to nonexistent, and the Ada logjam is broken.

Why was Ada allowed to hold pthreads up? IEEE standards require consensus, not just majority. This is necessary if the resulting standards are to be widely implemented and used. However, it does permit a vocal and determined minority to delay things.

By now, many readers are wondering just what was this core difference that caused so much trouble. As one might expect from the history of POSIX, signals were to blame. Specifically, according to P1003.1c, in a multi-threaded process, *sigprocmask*() is not required to (but is allowed to) mask signals globally, for all threads within the process. The change requested by the POSIX.5 WG was to require (rather than just allow) *sigprocmask*() to globally mask and unmask signals to all threads of the process. In either case, this would be in addition to the per- thread masking controlled by *pthread_sigmask*(). The Technical Editor and Reviewers of P1003.1c demurred, saying that requiring *sigprocmask*() to have global effect would lead to timing races in multi-threaded processes, particularly when those processes reside on multiple processors, and that having two masks could lead to use-update conflicts, and suggested that the requested change would also cause Ada implementations some trouble. The POSIX.5 WG disagreed. It was suggested that the POSIX.5 WG simply require implementors of P1003.5b, the draft Ada binding to Realtime POSIX (1003.1b-1993), to support the global *sigprocmask*() option. The POSIX.5 WG declined, saying that they feared that the UNIX platform vendors would not in fact support the option just to support Ada, leaving Ada high and dry.

There were many other arguments, too many to recount here. The root problem seems to be that the fundamental model of Ada, a language plus an operating system, differs greatly from the model of C and UNIX, and the Ada folk are having some difficulty coming to terms with having lost the language wars to C, and thus being forced to bend to what they consider to be an inferior model. Even within the US Department of Defense, Ada's star is waning.

What else is the POSIX 1003.4 Working Group doing?

The POSIX.4 WG has 3 other major projects: P1003.4b (now renumbered as P1003.1d), P1003.1i, and "P1003.4d," discussed below.

POSIX.4b (P1003.1d), 130 pages, currently in ballot resolution, contains a number of realtime interfaces and options that arrived too late to be included in 1003.1b-1993 (which itself consists of POSIX.1 and POSIX.4 combined). The major new interfaces and options are:

- *spawn*(), a functional merger of *fork*() and *exec*(), needed both for efficiency and for allowing use on platforms lacking memory management hardware;
- a sporadic-server scheduling policy, used to prevent asynchronous high-priority processing from totally consuming the computer;
- cpu-time clocks and timers, used to both measure and bound use of cpu by processes;
- *devctl*(), the successor to the *ioctl*() of classic UNIX;
- Interrupt Control, a set of interfaces intended to allow direct application-level control of devices such as array processors and radar signal processors; and
- Advisory Information, a set of interfaces that allow an application to declare to the kernel that, for instance, a specified file will be read sequentially, allowing the kernel to optimize performance.

A number of existing interfaces are also being augmented by the addition of variants supporting time-outs.

P1003.1i (also known as "POSIX.4 technical corrections") consists of correcting the POSIX.4 interfaces to remedy clashes detected when POSIX.4 was merged into POSIX.1-1990 to yield 1003.1b-1993, as well as minor problems discovered by early implementors of 1003.1b-1993. P1003.1i is a fast-track effort with a very limited and specific scope, and is expected to go to ballot about January 1995. As the changes are small, simple, and few, approval is expected to be rapid.

"POSIX.4d" (no formal IEEE number assigned as yet), approved as a project by the SEC at the October 1994 POSIX meeting, and for which a 73-page draft already exists, contains a number of realtime interfaces and options that arrived too late to be included in POSIX.1d. The major new interfaces and options are:

- Typed Memory, a set of interfaces supporting the *mmap*()-like mapping of diverse kinds of physical memory (e.g., SRAM, DRAM, ROM, EPROM, EEPROM) via multiple and/or diverse physical paths used for instance to access special hardware and memory via attached VME busses;

- *nanosleep_abs*(), a high-resolution *sleep*() allowing the user to specify when to awaken, rather than how long to sleep;
- Barrier Synchronization, a set of interfaces intended to support efficient implementation of parallel DO/FOR loops on massively parallel computers;
- Reader/Writer Locks, used to allow efficient parallel access to data in situations where reads vastly outnumber writes;
- Spinlocks, a very fast synchronization primitive for use on shared-memory multiprocessors; and
- Persistent Notification for Message Queues, an option for POSIX.1b-1993 Message Queues.

POSIX.13 draft 6, 100 pages, in ballot resolution and at about 75% approval, is a family of four related realtime profiles ranging in size from the very small through a full-featured platform conforming to essentially all of POSIX.1b-1993 (POSIX.1 plus POSIX.4) and POSIX.1c (threads), with realtime options chosen. The smaller profiles specify just that subset of POSIX interfaces needed to "clothe" widely-used small kernels such as pSOS, WindRiver, and VRTX32, which although very similar in function differ greatly in interface details. Standardization of these interfaces will yield the same benefits for embedded and realtime as standardization of UNIX did for workstations. In addition, the POSIX.13 interfaces are chosen to allow multi-computer distributed systems to be built, such as those used in factory automation. Such systems are typically set up as a hierarchy, with a few large-profile machines at the top, and a large number of smaller profile machines at the bottom controlling this or that piece of machinery, perhaps with an intermediate layer of supervisory machines between top and base, and all communicating with peers, superiors, and subordinates as needed.

Why are POSIX Document and Working Group Names so Confusing?

Good question; answer lost in prehistory. Sorry. The WGs are currently named (well, numbered) after the first standard they produced. When the WGs later took on added work, the WGs retained their original names. To further the confusion, the IEEE renamed all the documents and standards, and even the POSIX faithful are having some difficulty keeping the names straight. For example, the POSIX.4 Working Group was founded to produce the POSIX.4-1993 standard, known as "P1003.4" in its days as a draft standard. The POSIX.4 WG later started the POSIX.4a, POSIX.4b, and POSIX.13 draft standards; all but POSIX.13 have now been renamed. There is now a push to change WG names to an approximation of English, while the standards will continue to be numbered. The current set of non-numerical names aren't noticeably more informative than the numbers they replace. Stay tuned.

A View from the Chair

Lowell Johnson <3lgj@unirsvl.rsvl.unisys.com> reports on A View From the Chair:

The Portable Applications Standards Committee, PASC, is obviously going through a period of change, and nobody can be sure exactly what the future will bring, but a couple of things are fairly certain.

The first is that we will have to create quality standards with the reduced participation level we have experienced recently. Even though the economy will probably be improving, I do not think we will see a significant increase in participation for a variety of reasons. Companies are dissatisfied with how long it takes us to generate a standard and many are not even sure they want a complete set of open system standards (they might have to comply with them all).

So the second certain change will be that we have to develop or improve our processes so that we can reduce the time between project initiation and the granting of final ISO standard status. There are several things we will be discussing about how we can accomplish this, but I know from personal experience that huge standards, especially those over 1000 pages, take unacceptably long to both write and ballot.

A more subtle change will be in the types of standards we produce. The recent decision by JTC1 to allow public specifications to enter the fast track process to become ISO standards (under certain conditions) will mean that popular subjects, developed or supported by broad user bases, may go that route to standardization, rather than go through a formal standards development organization (like IEEE).

I think we will still have some major items to work on, but the emphasis will change in some situations. We will have to provide more small standards on specialized topics and to fill holes not covered by existing major standards. Also, someone will have to define environments where all the base standards that have been developed can work to form an integrated whole. Fortunately, we know how to do that: they are called profiles.

You all know how difficult it is to define an open system, since basically everyone has their own definition, and most of them are not truly open (at least in the sense of portability). While contemplating this (and trying to write another article), I began thinking that the concept of a "Strictly Open System" would be more useful.

A Strictly Open System would strictly conform to all its underlying standards, and nothing else. Strict conformance to a base standard means conforming to all the interfaces defined in it, but with no extensions. Therefore, a Strictly Open System would not rely on any extensions to any standard.

Unfortunately, we all know that no system is strictly conforming since we do not have a complete set of system standards, and even if we did, most vendors use specific (and often undefined) hooks into the system to increase performance or otherwise improve salability. There is also the problem of how such a system could be proved to be conforming.

One possibility (although very gross) would be to measure the amount of documentation it would take to specify all the extensions used. However, I do not think even this is practical for most systems, but it would make an interesting thought experiment to estimate how large this pile of hypothetical documentation would be. Obviously, a perfect Strictly Open System would not require any of this documentation since no extensions would be needed.

A simple view of our future would be to work on anything which could be used to reduce this imaginary pile of extensions documentation. We will work on many other things (such as profiles), and may work in some new areas (such as application conformance), but we will also have to develop new ways to work with non-traditional standards groups.

USENIX Supports Student Participation and Professional Development

The USENIX Association sponsors an Educational Outreach Program for faculty members of computer science departments around the world. These faculty liaisons provide their students access to USENIX publications, upcoming events as well as conference scholarships for full-time students. If you would like more information on the benefits or are interested in becoming a liaison please contact Diane DeMartini via email *<diane@usenix.org>*.

The following student benefits are offered:

- A $500 cash prize is awarded for the Best Student Paper at the annual Winter Technical Conference. (Students are eligible also for Best Paper and other awards.)
- Membership in USENIX for full-time Students is only $25. As a member, your benefits include:
- Free subscription to *;login:*, technical features, system administration tips and techniques, international calendar of events, SAGE News, media reviews, Snitch Reports from the USENIX representative and others on various ANSI, IEEE, and ISO standards efforts, and much more
- Free subscription to *Computing Systems*, the refereed technical quarterly published with The MIT Press
- Discounts on registration fees for the large, multi-topic technical conference, the System Administration conference (LISA), and the various symposia addressing single topics such as object-oriented technologies, security, operating systems, high-speed networking, and mobile computing – as many as seven technical meetings every year
- Discounts on proceedings from USENIX conferences and symposia and other technical publications. Discounts on the USENIX Association book published by The MIT Press. Now available: *The Evolution of C++: Language Design in the Marketplace of Ideas*, edited by Jim Waldo of Sun Microsystems Laboratories
- Discounts on BSDI, Inc. products. BSDI information: 800 800-4BSD
- Discounts on the five volume set of 4.4 BSD manuals plus CD published by O'Reilly & Associates and USENIX
- Savings (10-20%) on selected publications from McGraw-Hill, The MIT Press, Prentice Hall, John Wiley & Sons, O'Reilly and Associates, and UniForum
- Special subscription rates to *The LINUX Journal*. Phone: 206 527-3385
- Opportunity to join SAGE, the System Administrators Guild
- And maybe most important, participation in leadership in the systems community

For membership information please contact:

The USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
510/528-8649
FAX 510/548-5738
Email: *<info@usenix.org>*
WWW URL: *<http://www.usenix.org>*

STANDARDS

An Update on Standards Relevant to USENIX Members

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

Snitch Reports

Not every article that is published in this column is met with total delight by all the readership. Now, that shouldn't be a surprise, but occasionally I receive some of the flames that result. That's part of my job, and I encourage you to let me know if something appears here with which you take offence.

The authors of the articles that appear here write on a voluntary basis (OK, USENIX pays the tab to take them out to dinner once in a while), and write as individuals, expressing a personal opinion. "Snitch" reports are not, ever, corporate opinions. POSIX, possibly more than most other International Standards groups, is driven by engineering principles rather than corporate politics. If you read this column regularly, then you should, if I'm doing my job right, get informed of progress (or the lack of it), of the reasoning behind apparently strange decisions, and of the overall direction of UNIX related standards. You should be told by those people in the thick of it. Of course, if the author's in the thick of it, and there have been two radically different approaches discussed, you might get to hear only one side of it, since the opinions are personal.

This column is largely factual, coloured by opinion, with a hint of controversy. I want it to be readable, not only by those who are in the midst of the action, but also, and mainly, by those around the world who cannot otherwise find out what's happening without waiting for the book to appear. If one of the authors of these reports says something you don't like, please submit a response for publication.

The IEEE process that we follow demands openness. Anyone can join the debate. The process demands consensus from all involved. If you disagree with the way that POSIX is going, then get involved in the debate. Of course, sometimes there has to be a losing side. If you're on the losing side, then you must simply accept the consensus opinion.

The following reports are published in this column:

- ANDF BOF
- NII/GII Information Infrastructure BOF
- POSIX.0: Guide to POSIX OSE
- POSIX.6: Security Extensions

Our Standards Report Editor, **Nick Stoughton**, welcomes dialogue between this column and you, the readers. Please send your comments to *<nick@usenix.org>*.

;login: VOL. 20 • NO. 2

Report on ANDF BOF

Chuck Severance <crs@msu.edu> reports on the July 1994 meeting in Nashua, NH:

Background

Five years ago, I attended an OSF member meeting and went to a presentation on "Architecture Neutral Distribution Format" or ANDF. ANDF is a proposed technology which will allow vendors "shrink-wrap" software which will run on a wide variety of computer systems independent of the CPU/Operating System combination. The only way to accomplish this effectively is to distribute the source code and allow it to be compiled on whatever system the user happens to have. Of course there are several obvious problems with this approach. It is much easier to pirate source code to a new system. The user can change the source code, recompile it, and end up introducing bugs in the program which the company will have no way of knowing what is going wrong. Also competitors can easily find out trade secret techniques used in a vendor's product.

Needless to say source code distribution is not practical for any software other than public domain software.

ANDF was proposed as a solution to this problem. ANDF represents a compromise approach somewhere between distributing a completely compiled binary and the original source code. With ANDF the software company partially compiles the source code into ANDF (this step is actually called "producing" rather than compiling in ANDF). ANDF is a representation of the source code which is easily converted into the final machine language for the target system. Each CPU/ operating system has a product called an "installer" which finishes the compilation process producing an executable for the system.

As I listened to the presentation, I became very excited about the possibilities for this new technology. Coupled with the possibility of OSF/1 becoming a ubiquitous operating system, ANDF promised a "brave new world" in which one could go to the local store, buy a single copy of a word processing package and run it on any system you can find.

What Happened?

We certainly cannot go out today and buy applications in ANDF and install them on our computers. Well, several things went wrong in what seemed to be a fairy tale with happy ending. First, ANDF cannot completely compensate for a lack of source code portability between systems. ANDF can solve many simple portability problems, like slight changes in parameter conventions, or the lack of a particular library routine on a particular system. But ANDF cannot convert an X-Window system application to a MS-Windows application. ANDF enhances existing portability between systems but does not create portability where no source code portability exists.

Also, vendors are a bit afraid to adopt ANDF. Portability is a two-way street. You can gain customers or you can lose customers. Certainly that appeals to users but it is very frightening to a vendor who has a tough enough time keeping customers from bolting. If all it takes to switch vendors is to buy a new system and "re-install" all of your software (assuming all the licenses were transferable) the buying public can become very fickle. Today, investments in software often are the key reason for staying with a particular vendors hardware.

The net result from my perspective (probably because I stopped going to OSF member meetings some time ago) is that ANDF fell off the face of the earth. I have not even heard the term mentioned by a vendor in the last few years.

As such, I was very much looking forward to the ANDF presentation at the recent POSIX meeting. For me it was like seeing an old friend at a 5-year class reunion.

So What's Up?

The ANDF BOF was well attended including presenters from OSF and the Defense Research Agency from the United Kingdom. About 20 people from POSIX attended the meeting. The first presentation was by Andrew Johnson of the OSF. The first part of his presentation was an overview of ANDF which summarized the goals and approaches that I described above. I was beginning to think that I was about to hear the same talk I heard before at OSF but there was more to come.

The ANDF format has been changed. The original ANDF was based on a compiler techniques from the 1970s. The format limited the potential for optimization for more advanced CPU architectures. The new ANDF format is a "tree-structured" representation of the program. This is the intermediate format used by most of today's highly optimizing compilers. This shift in intermediate format makes it possible to write an "installer" process which generates code that is as well optimized as any of today's vendors compilers.

OSF has compiled a number of demonstration applications. First, an ANDF version of the Spec92 benchmark executes within 5% of the performance obtained using the native compiler. And sometimes ANDF does better than the vendor compiler. They have also compiled several large production applications (1 million lines +) and have demonstrated that the ANDF versions execute with equivalent performance and reliability to the versions with native compilers.

In the next part of the presentation, they described a project which developed a front-end to gcc (public domain) which accepted ANDF as input and produced binaries using the rest of the gcc compiler. This can act as an installer function which can convert ANDF to any of the systems supported by the gcc compiler. They have put the source code to this ANDF installer in the public domain.

The second half of the presentation was given by the Defence Research Agency (DRA) of the UK. The DRA is one of the developers of the ANDF technology. DRA is working very hard on producing real products based on ANDF. They have spent a great deal of time on the reliability of the C producer. One of the challenges of a producer is to delay decisions typically done by a pre-compiler until just before the "link" phase of the compilation. This required the development of special versions of the C include files which captured the essence of function calls without actually generating function calls until the installer runs. This technology is called TICL (don't ask me what it stands for).

The TICL is generated directly from the text of a standard such as POSIX 1003.1. Because the TICL only allows the application to use features specifically described in the standard, it actually identifies areas where the application is using features beyond those specified in the standard. This aspect of TICL has some interesting uses in testing application conformance to standards.

Other organizations working with the DRA and OSF are expanding the suite of ANDF technology.

One of the goals of the BOF was to discuss the possibility of producing some sort of standard for ANDF. The ideas of making ANDF an X/Open standard and/or an IEEE standard were both discussed. While no specific plans were made, one or both of these organizations might be working on an ANDF standard within a year in my opinion.

There was a lot of discussion of how to move ANDF forward from the point of view of computer vendors, users, and application developers. Folks generally agreed that users had the most to gain followed by application developers. However users will not insist on ANDF if it is not available. This was termed the "chicken-egg-rooster" problem.

Of course there was a strong suggestion to put the entire ANDF kit into the public domain to encourage experimentation with the technology. OSF and DRA nodded wisely when this was brought up but did not have any solution in hand.

People interested in ANDF who haven't looked at it for a while should take another look. With nearly the entire kit in the public domain, there should be ample opportunity for folks to experiment. They have an ANDF binary of Mosaic available which would be an interesting test that we all could do.

For more information on ANDF, take a look at *http://riwww.osf.org:8001/ri/andf.papers/ANDF-intro.html* on the WEB and *riftp.osf.org* on anonymous FTP.

Report on NII/GII Information Infrastructure BOF

Charles Severance <crs@msu.edu> reports on the October 1994 meeting in Seattle, WA:

Last October, a Birds-of-a-Feather (BOF) session was held at the Seattle POSIX meeting to introduce the attendees to some of the issues in the National/Global Information Infrastructure (aka NII). The BOF was hosted by Jim Isaak, the acting chair of the IEEE SCC33, the IEEE standards coordinating committee that acts as a focal point for identifying areas for NII standardization for the IEEE.

In the first part of the presentation, Jim gave an overview of the organizations working on NII standardization.

- ANSI IISP – Information Infrastructure Standards Panel. This is a focal point for the Information Infrastructure standards requirements and responses.

- IEEE SCC33 – The focal point within the IEEE for Information Infrastructure standards efforts (Jim Isaak is the Acting Chair of this group).

- CSPP – Computer Systems Policy Project – This group consists of the CEOs of the major computer manufacturers. They have written a document which provides a high level vision of the NII.

- XIWT – Cross Industry Working Team – This group is made up of a broad range of computer, telecom, and cable companies (42).

In the second part of the presentation, the group discussed some of the broader issues in the NII. The first observation is that there are many perspectives on what the NII should actually become. Some folks believe that the Internet is the prototype for the NII. Others feel that the purpose is to have 500 cable channels with a little remote control so folks can make orders without using their telephones. Yet another perspective is that it is a super phone network with ISDN and beyond.

The good news is that because each of the industries represented (telephone, computer, and entertainment) is very powerful, the NII will most likely be a combination of the dream systems of all of the groups.

Report on POSIX.0: Guide to POSIX OSE

Kevin Lewis <klewis@gucci.enet.dec.com> reports on the January, 1995 meeting in Ft. Lauderdale, FL:

The best news a ballot coordinator can get is that the draft standard he is working on has completed recirculation with only 3 objections. Such was the case with the second recirculation for the POSIX.0 guide document. When the ballot closed December 19, a total of 6 ballots had been submitted containing 69 comments and 3 objections. Two of the 6 ballots were "NO" votes. One was converted, the other remains a negative vote. This brings the ballot results on the guide to the following figures:

81 eligible voters
56 affirmative votes
8 negative votes
64 total votes
= 87% AFFIRMATIVE

The pathway to approval is now clear. All comments and one of the three objections will be reflected in the next draft of the guide which will be draft 18. This will be submitted to the IEEE on February 3 for distribution to the IEEE Review Committee (RevCom). RevCom will meet on March 15 in Piscataway, NJ for final review at which time a recommendation to the IEEE Standards Board will be made. I anticipate no problems whatsoever in obtaining approval by the Board. I am told that once a draft is approved, it takes approximately 6 months for it to be published.

The March board meeting is quite symbolic in this development effort for it was exactly seven years ago at the March 1988 POSIX meeting that the working group had its first meeting. I've been writing as one of the original snitches since that time. I shudder to think how many times I forecasted (and in error, I might add) when this work would reach completion.

At the international level, draft 18 will be submitted to the SC22 Secretariat for the last ISO ballot which will be as a DTR (Draft Technical Report) within JTC1.

Now, as for the future, there are several directions that the POSIX.0 working group could go. Currently, POSIX.0 has spawned the OSE (Open Systems Environment) User Profile Study Group which is focusing on the development of a methodology for identifying OSE user requirements. In addition to this, several people have shown interest in starting work in the distributed systems area, commonly referred to as "middleware." Others have shown interest in examining how ODP and OSE should come together. And there are others who are interested in expanding the current guide document further, specifically in the reference model and services areas. Right now, the aforementioned study group is the only effort that has been organized. The other areas are waiting for a champion to come along. Could that be you?

Report on POSIX.6: Security Extensions

Lynne Ambuel <ambuel@dockmaster.mcsc.mil> reports on the January 1995 meeting in Fort Lauderdale, FL:

Security Ballot Reforms – and Lives to Tell about It

One curious thing about human nature is the compulsion to avoid change. The manager that fights every effort to automate the office. The worker in a dead-end job that doesn't accept the opportunity to move into a better one. The battered women who refuses to leave the abusive relationship. Stories upon stories are told of people who would just as soon stay in difficult, if not unbearable, situations because they know what to expect and they can't be sure that the change wouldn't be worse. It is funny how we can be so comfortable in our discomfort.

By 1994, much to my dismay, the Security Extensions ballot resolution group had fallen into this same paradigm. We had grown comfortable with our draft and the ballot group. The problem was we didn't realize how miserable we really were. By that time the ballot group was almost four years old. It had become an adventure to find the 275 people in the group every time a new draft was released for ballot. Once we found them it was even harder to make them care. It would take up to three months of concerted effort just to get enough ballots returned to close ballot. It was also a challenge to get enough non-abstentions to be a valid ballot in the eyes of IEEE. On several issues we had reached stalemate. (The ACL mask was swapped in and out at alternate drafts.) Looking back, we should have been screaming for a change.

But – nooooo. Early in 1993 the IEEE suggested that we re-form our ballot group. We resisted; quite adamantly, I might add. We knew what our balloters were going to say. A new group may have had a totally different idea of what the standard should contain. They wouldn't have the "history" and would set us further back by rehashing debates that were decided years ago. We knew our situation was bad, but we were sure that the change would be worse. By the end of 1993 IEEE insisted that we re-form our ballot group. We had no choice. There was great weeping and gnashing of teeth but we finally submitted.

Much to our surprise, things went smoothly. The new ballot group of 82 people was formed by the end of 1993. There were some snags in letting people know they needed to sign up, or re-sign up; but those were relatively mild. The new group was comprised up of a mixture of people from the old ballot group as well as new members (37%). Several people who had wanted to comment on the standard but had not been involved when the original ballot group was formed were able to join. Others that had lost interest and would either vote affirmative without reading the draft, abstain or not return the ballot at all did not rejoin. Somehow the new group contained a balance of background and experience that equaled, if not exceeded, that of the original group.

What was even nicer was the ballot closed within two days of the original closing date, without any prompting. The most difficult thing was warning the old-faithfuls that were used to having an extra month or so to finish their comments that the ballot period was about to close and they had better get their ballots to IEEE ASAP.

Even our fears that the comments and objections would send us in a new direction were ill-founded. Our affirmative votes dropped from 61% to 47%, however, very few ballots had any really hard issues. There were no new hard issues. There were some new points of view, but they mostly affirmed the general direction we had been taking. And yes, there were some objections that requested changes way out in left field. (Sorry we can't solve world hunger in the security standard.) This will take a little education time. All in all, the ballot resolution task set before us was the most manageable that it had been since we began the process of resolving ballots on this standard.

Because of all these factors we are about to release our next ballot draft (D15), a mere three meetings after closing the last one. We also believe that we have done enough work to significantly increase our affirmative votes. I think we will match our all time high of 61% and may even exceed it. We also think we can put out a follow-on draft by the end of the year, if the ballot is closed by the April POSIX meeting. That would be the first time we have released two drafts in the same year.

Change is good.

STANDARDS

An Update on Standards Relevant to USENIX Members

by Nick Stoughton
USENIX Standards Report Editor
<nick@hoskyns.co.uk>

The following reports are published in this column:

- •POSIX.1: System API
- •Language Futures
- •POSIX Conformance Testing

Our Standards Report Editor, **Nick Stoughton**, welcomes dialogue between this column and you, the readers. Please send any comments you might have to *<nick@usenix.org>*.

;login: VOL. 20 • NO. 3

Snitch Reports

Report on POSIX.1: System API

Nick Stoughton <nick@hoskyns.co.uk> reports on the January 16-20, 1995 meeting in Ft. Lauderdale, FL:

If You Can't Beat 'Em, Join 'Em

Well, after tiring of twisting people's arms (usually unsuccessfully) to snitch on the work of the POSIX.1 Working Group, where I am usually to be found during a POSIX meeting myself, I thought I would have to submit at least one real snitch report myself. Show everyone how its really done... but the only trouble is, I have all those other great snitches on the other groups to live up to!

POSIX.1 is a small group; typically only about 6 or 7 show up at a meeting. Until this meeting, it had only two items of work: POSIX.1a and Interpretation requests. However, a new Project Authorization Request for removable media has just been passed and assigned to this group, so perhaps I'll have some more people to dragoon in April!

Another Year, Another Draft

The great highlight of the January meeting was that Lee Damico, our technical editor, had managed to ship draft 12 of POSIX.1a to the IEEE a week before the meeting. This was after many late nights, weekends, and even sacrificed Christmas vacation trying to get all the ballot resolutions into the draft.

Several sections were substantially rewritten after the first round of balloting; most notably the checkpoint-restart interfaces. These are highly contentious for a number of reasons:

- They are not based on existing practice, and are substantially invention by the committee. As a USENIX and EurOpen (the European equivalent of USENIX) representative, this makes me decidedly uneasy. However, they are optional, and I don't believe they are unimplementable. Let us hope some brave reader tries to build these interfaces before the standard is published so we can have some existing practice and experience to work with!
- Two different groups within POSIX require them for future extensions. The Batch profile, which was responsible for wording the original (up to Draft 11) version, needs some interfaces to allow long running jobs to be stopped in their tracks and restarted later. In addition, the work being done by the Services for Reliable, Available, Serviceable Systems (SRASS) working group needs interfaces to checkpoint processes if the system is failing. This would allow those processes to be restarted later in, potentially, quite a different environment after various devices have disappeared.

The batch and SRASS arguments look set to roll on for some time; the two groups want quite different things. However, it is also undesirable to have two different checkpoint interfaces: one for batch, one for SRASS. The ballot group just hates the whole idea because it is invention. But if we remove the interfaces, then these two follow-on groups are in trouble.

The other recurrent long-running argument in POSIX.1 is over the interpretation of trailing slash characters on pathnames. Historically, System V based systems ignored all trailing slashes, whatever. BSD based systems only permitted trailing slashes on pathnames that were existing directories. In draft 12, the BSD behavior is demanded. From an engineering perspective, this is a much cleaner answer. However, the weight of existing System V platforms that would lose compliance as a result of this is substantial.

From the application developer's, as opposed to the system implementor's, point of view, it is a good change. If you wrote an application that depended on trailing slash behavior before POSIX.1a, it was not portable. After the publication (well, we have to be optimistic), such an application *will* be portable.

For most system implementors it is not too hard a change to get working in a future release, and most have at least one release a year.

News Flash

As ballot coordinator for POSIX.1, I can tell you that the ballot on draft 12 closed on time, with a 44% affirmative vote. Most of the ballots look as if we should be able to work through them quickly, so progress can be expected in April.

Report on Language Futures

John L. Hill <HILL@po3.bb.unisys.com> writes on Language Futures

Abstract

This article declares that there are at least two things that the industry could, indeed should, pursue vigorously. Those are language independent application programming interfaces and formal description techniques. The information technology (IT) industry needs to develop them further and employ them more broadly in order to maintain the current rate of improvement in software technology.

The Setting - Abstraction is the Root

From a historical perspective, the art of software development is progressing to become a mature science. The progression (from art to science) is becoming increasingly visible because there is only now beginning to emerge the precepts of a science. First, the IEEE began a movement to establish software engineering as a formal, certifiable profession. The existence of a formal profession is one mark distinguishing art from science. Software engineering is beginning to reveal its maturity. Second, a body of standard, formal definitions, and agreed upon principles is emerging. As time passes software engineering should continue to lose much of its mystique. It will grow in strength as monumental changes take place.

The original "hard-wired" but variable program computers gave way to truly soft programmable computers. At first the languages in which one wrote programs were "close to the machine." Early languages, notably machine code and assembly languages, required a programmer to know the internal structure and operation of the specific target machine. Limitations and difficulties revealed themselves in droves. Assemblers were not applicable to any but the single, target computer architecture. This limitation of portability affected not only the program but the programmer as well. Yet another complication was that the programs were often so "personalized" to both the target machine and the programmer, and difficult to understand, as to not be maintainable by anyone but the author.

With the inexorable passage of time, champions of software engineering dealt with the weaknesses of those early generation programming tools. They invented higher order languages. The high order languages masked programmers from the internal workings and structure of the computers on which they were working. It was simply the application to computers of something the world's great painters and sculptors embraced many decades before... abstraction. By way of analogy, I assert Grace Hopper was the logical equivalent of Mondrian in terms of abstraction. In concentrating on the essence of what the application required, rather than the means of satisfying those requirements, the programmer was relieved of "unnecessary" computer operational details.

In the visual arts an entity's form, color, and texture are the abstractions of its physical representation. Programming languages are similar in terms of abstraction. Executing a single verb often invokes multiple primitive operations in the computer. By making those verbs natural-language-like, the programs themselves became easier for mere humans to write, reread, and maintain. Data manipulation became simpler, too, with the appearance of more and varied data types coupled with the verbs to manipulate them reliably.

Another aspect of abstraction is becoming evident in computer operating systems. Operating systems are becoming increasingly modular. Functionally different elements are separating from the core operation system and from one another. As a by-product, this provides the means for functionally expanding and tailoring individual OS elements. The current evolution of middleware is a representative example.

Yet another leap is taking place in software engineering technology with the truly generative languages. These are frequently referred to as fourth generation languages (4GLs). 4GLs allow the programmer to functionally describe the application to be programmed and leave the generation of the traditional high order language program to the 4GL.

As these improvements in software programming tools and languages are taking place, the technologies that are used to implement them changed, too. Improvements have come rapidly to the physical computer system itself. Many of these important advancements have not greatly affected the programmer since programmer activities tend to take place at a higher level of abstraction. The result is that the programmer largely remains insulated from the rapid advancements taking place in computer hardware. There are notable exceptions, such as self-identifying peripherals and dynamic configuration. These allow the programmer to be virtually ignorant of the physical devices and their physical, and often their logical relationships. Some changes in computers that challenged programmers include new input and output devices as well as communications protocols and media.

Programmers, resilient by nature, have readily adapted to all these improvements. There is a large challenge, however. That challenge is in retaining the current rate of improvement in software engineering within a framework of continually changing hardware and related platform capabilities. As has been the case previously, the solution lies in abstraction.

There are two areas in which the application of abstraction will benefit the IT industry. They are language independent methods of specifying application programming interfaces, and formal description techniques. Adoption of the principles of these will yield enormous dividends.

Language Independent Application Programming Interfaces

An application programming interface, or API, is that artificial boundary over which an application program obtains a service it requires in order to obtain some result. A representative example can be found in the programming language binding to a database manager. The Ada binding to SQL is one such API. By a programmer using that API, the Ada programmer has access to the extensive data management services of SQL. Without that access, the programmer would have to perform data management functions within the program itself.

As powerful and useful as the Ada/SQL binding is, it and others like it, suffer from language dependency. That is, the binding is good only for Ada. One likely consequence is the sacrifice of interoperability in a mixed language environment. Perhaps a bit of explanation is desirable. If the environment of the SQL database is diverse, as far as the programming languages in use to access its data are concerned, then the syntactic capabilities of those languages' bindings will differ. Because programming languages very frequently differ semantically, the functionality of the bindings will differ. This results in reduced interoperability when viewed from the point of view of the SQL database.

The application of the concept of abstraction leads to one way of avoiding the sacrifice. If, instead of developing language specific APIs, one develops a language independent specification of the API and then develops language specific bindings to that language independent specification, then the bound service (in the instance of the example SQL is the service) could reliably exist in a mixed language environment. This preserves interoperability.

Naturally, there are difficulties with the implementation of this proposal. First is the methodology for expressing the language independent specification. The methodology must be sufficiently robust to encompass the broadest range of programming language paradigms. The IEEE's standards development committee for POSIX developed one effective methodology. Its application outside the POSIX environment is certainly possible.

Second is the actual development of the language independent specifications themselves. Because there is no direct, financial value to having a particular language-independent specification (after all, the language dependent binding is valuable to the programmer and not the language independent specification to which it is bound) it is difficult to justify development. The IT industry needs an economically viable means for developing language independent specifications.

Formal Description Techniques

A formal description technique, FDT, is a mathematically precise (complete with proofs) means of semantically describing systems. Systems described using FDTs exhibit properties such as symmetry and completeness. The use of FDTs to describe computer programming languages and applications will result in high-quality, low-defect implementations.

As was the case with language independent APIs, there are some practical difficulties with FDTs. First is the syntax used by the FDT. There are currently several projects, at various stages of completion, developing standards for FDTs. Examples include ASN.1, LOTOS, and ESTELLE, each targeting a portion of the data communications realm. Both VDM-SL and Z, while earlier in the development process than their communications counterparts, apply to the programming languages and diverse applications realms. Second is the lack of product backing so necessary to broad-based development. There are few product possibilities stemming from FDTs and they are highly specialized. The value of FDTs derives not from the tools that it uses but in the quality of the applications it produces.

Conclusion

The IT industry today is developing two very valuable tools, language independent APIs and FDTs, albeit at a low priority. Their value is indirect. They are complicated and highly specialized. Their value, though, is undoubtedly high having extensive possibilities for implementation. In order for the world to continue the rate of improvement in software engineering technology, the IT industry needs to heartily embrace them.

Report on POSIX Conformance Testing

Kathy Liburdy <kliburdy@eng.clemson.edu> reports on the January 16-20, 1995 meeting in Ft. Lauderdale, FL:

Should Test Methods Be Standardized?

A point of contention which arose during the rebellion against required test methods reappeared in the January POSIX meeting: Should test methods be standardized at all? Although working groups are no longer required to develop test methods, the proper dress for test methods has yet to be established.

In the midst of concerns focused on testing (e.g., 2003.5 conformance to 2003), members of the POSIX.5 Ada working group reconsidered the role of testing in the Big Picture of open systems standards. While the POSIX.5 working group unanimously agreed that a testing capability was needed for the POSIX Ada Language Interfaces, there were concerns about the standardization of test methods. The following points were raised by members of the group:

1. Standardization of the test methods invites problems with interpretations in the event the base standard disagrees with the test method standard.
2. Standardization of the test methods prior to actual use violates the conventional wisdom of standardizing based on "existing practice."
3. The benefits of test method standardization are unclear, while the disadvantages (e.g., the time and labor involved) are painfully obvious.

A meeting with representatives of the POSIX testing community was requested in hopes of resolving, or at least understanding, these issues. Both Roger Martin, chair of 2003 and the Sub Committee on Conformance Testing (SCCT) and Lowell Johnson, chair of the IEEE Portable Applications Standards Committee (PASC), attended this meeting. During the discussion which circled these issues, the idea of publishing 2003.5 as a Recommended Practice was proposed as a sort of multi-purpose compromise which would:

- keep the test methods in the official PASC process,
- remove any doubt about "who wins" in a disagreement with the base standard,
- remove concern about conformance with 2003 (since the scope of 2003 is test methods "standards"), and
- serve as an experiment to help determine the appropriate role of test methods in IEEE POSIX.

This suggestion was met with general favor, and the POSIX.5 working group agreed to submit a request to the Sponsor Executive Committee (SEC) to modify the status of 2003.5 from that of a Standard to a Recommended Practice.

What are Test Methods?

In order for the POSIX community to reach consensus on the appropriate niche for test methods (e.g., standard, recommended practice, guide, or none of the above), attention must be given to uncloaking the meaning of this much used and ill defined term.

According to the January draft of 2003, test methods are "software, procedures, or other means to measure conformance. Test methods may include a PCTS, PCTP or an audit of PCD." (For the nonconformance tester, PCTS is the POSIX Conformance Test Suite, PCTP is the POSIX Conformance Test Plan, and PCD is the POSIX Conformance Document). In practice, the use of test methods may refer to the actual implementation of test methods or the specification of test methods. While the stated scope of 2003r is test methods, the emphasis is almost entirely on the development of assertions about required behavior of a conforming implementation. However, the definition of test methods makes no explicit mention of assertions.

Ballot objections to 2003r reflect this confusion and concern: what exactly are test methods? Many of the ballot objections submitted against 2003r address fundamental definitions such as test methods and assertions, and question the intuitiveness of recently crafted definitions. As an example, an assertion is defined to be a test specification. The terminology "assertion test" is then introduced to mean the executable form of the assertion, requiring the reader of the document to comprehend a "test specification test." The wobbly state of the terminology reflects the immaturity of the area of conformance testing, which should send a bright, loud, blinking warning to anyone considering the merits of standardizing test methods.

Study Group Formation for Automated Testing

The initial concern in the development of automated test methods for the Ada Language Interfaces was for achieving conformance to 2003r. This concern was well-founded, since 2003r embraces a manual approach to test development. As the interest in automated testing increased, the question of conformance to 2003 began to be replaced by the question: "Is there an approach for developing test specifications based on this experience which could be useful to others embarking on a similar effort?"

This question, with an affirmative answer, was brought before the SCCT/2003 group for discussion. Two possible alternatives were identified:

- include an automated testing methodology in 2003r, or

- submit a PAR for a separate document to address automated testing

Since 2003r ballot resolution had not been completed, and the issues related to the requirements of an automated testing methodology would benefit from an open review, the group approved the idea of requesting a study group for the April meeting. The purpose of the study group is to determine the requirements of automated testing, and to evaluate 2003r with respect to these requirements. The request for an Automated Testing Study Group was brought before the SEC and approved.

Documents to be considered by the study group include 2003r, the current draft of 2003.5, the Clemson Automated Testing System Language Reference Manual and the Assertion Definition Language (ADL) Reference Manual. Members of Sun Microsystems Laboratories are scheduled to present recent updates and experiences with ADL. Because of the relationship between automated testing and formal specification languages, a representative of the X3J21 Committee on Formal Description Techniques is scheduled to provide an overview of ongoing work in formal methods. Suggestions for additional topics are invited.

AUUG Inc
UNIX AND OPEN SYSTEMS USERS

AARNet
MAIL SERVICE

Dear Site Administrator,

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

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

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

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

The required initial registration form is attached below. It should be completed and forwarded to AUUG's (postal) mailing address at the bottom of the form or faxed to (02) 332 4066. If you have any queries on the AUUG/AARNet arrangements please direct them to Catrina Dwyer at the AUUG office on (02) 959 3656 (catrina@swift.sw.oz.au) or myself (frank@atom.ansto.gov.au).

Regards,
Frank Crawford
AUUG-AARNET Administrator
AUUG Inc.

AUUG Inc. UNIX® AND OPEN SYSTEMS USERS

AARNet

Mail Service Application

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

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

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

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

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

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

AUUG Inc

UNIX AND OPEN SYSTEMS USERS

AARNet

MAIL SERVICE AFFILIATE MEMBERSHIP APPLICATION FORM

PLEASE PRINT CLEARLY!

Date: ____________________

Name of Organisation/Owner: ____

____

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

Name: ______________________________ Position: ______________________________

on behalf of the organisation named above.

Address: ____

____

____

______________________________________ Postcode: ____________________________

Administrative Contact: __________________________ Title: ______________________

E-Mail: __________________________________ Phone: (   ) ____________________

Fax: (   ) ____________________

Technical Contact: __________________________ Title: __________________________

E-Mail: __________________________________ Phone: (   ) ____________________

Fax: (   ) ____________________

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

Domain Names Requested: ____

____

Gateway Addresses: ____

____

____

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

* * * *

Send this page only to:

AUUG Incorporated
PO Box 366
Kensington NSW 2033

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

AUUG Inc UNIX AND OPEN SYSTEMS USERS

AARNet

MAIL SERVICE
AFFILIATE MEMBERSHIP
APPLICATION FORM cont'd.

Note A. Mail Delivery Information

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

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

Example Mail Directory Information request:

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

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

Note B. Getting Connected

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

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

MEMBERSHIP APPLICATION
INDIVIDUAL

AUUG Inc
UNIX® AND OPEN SYSTEMS USERS

To apply for AUUG membership, complete this form and return it with payment in Australian Dollars to:
REPLY PAID 66, AUUG MEMBERSHIP SECRETARY,
P.O. BOX 366, KENSINGTON, NSW 2033, AUSTRALIA
Tel: +61 2 361-5994 or 1 800 625 655 • Fax: +61 2 332-4066

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

NOTE: Please do not send purchase orders - perhaps your purchasing department will consider this form to be an invoice. Foreign applicants please send a bank draft drawn on an Australian bank.

INDIVIDUAL OR STUDENT MEMBERSHIP:
I, ______________________ do hereby apply for:

Renewal/New membership of AUUG	❒ $ 90.00
Renewal/New Student membership	❒ $ 25.00 (please complete certification portion)
International air mail	❒ $ 60.00

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

I agree that this membership will be subject to the rules and by-laws of AUUG as in force from time to time, and that this membership will run from time of joining/renewal until the end of the calendar or financial year.

____
Signature

____
Date

LOCAL CHAPTER DESIGNATE:
You can participate in the activities of a local AUUG Chapter. Part of your fee will be given to the chapter to support those activities. By default a regional chapter will be selected for you. If you would rather nominate a chapter, please specify here ______________ (indicate NONE for no chapter).

TO BETTER SERVE YOU, PLEASE PRINT YOUR CONTACT INFORMATION:

Name/Contact: ______________________
Position/Title: ______________________
Company: ______________________
Address: ______________________
______________________ Postcode__________
Tel: BH______________ AH______________
Fax: BH______________ AH______________
email address: ______________________

Over for Institutional Membership

STUDENT MEMBER CERTIFICATION: *(to be completed by a member of the academic staff)*
I, ______________________ certify that
(administrator)
______________________ is a full time student at
(name)
______________________ and is expected to
(institution)
graduate approximately ______________.
(date)

____
Signature

______________ ______________
Title *Date*

Please charge $__________ to my
❒ Bankcard, ❒ Visa, ❒ Mastercard
Account number: ______________________
Expiry date: ______________________
Name on card: ______________________
Signature: ______________________

AUUG Secretariat Use

Chq: bank ______________ *bsb* ______________
a/c ______________ *#* ______________
Date: ______________ *$* ______________
Initial: ______________
Date processed: ______________
Membership # ______________

AUUG Inc. as a user group, exists to provide UNIX® and open systems users with relevant and practical information, services, and education through cooperation among users.

Notification of Change

You can help us! If you have changed your mailing address, phone, title, or any other contact information, please keep us updated. Complete the following information and either fax it to the AUUG Membership Secretary on (02) 332-4066 or post it to:

AUUG Membership Secretary
P.O. Box 366
Kensington, NSW 2033
Australia

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

AUUG Inc
UNIX® AND OPEN SYSTEMS USERS

❐ The following changes are for my personal details, member #:________________

❐ The following changes are for our Institutional Member, primary contact.

❐ The following changes are for our Institutional Member, representative 1.

❐ The following changes are for our Institutional Member, representative 2.

PLEASE PRINT YOUR OLD CONTACT INFORMATION (OR ATTACH A MAILING LABEL):

Name/Contact: ________________
Position/Title: ________________
Company: ________________
Address: ________________
________________ Postcode________
Tel: BH ____________ AH ____________
Fax: BH ____________ AH ____________
email address: ________________

PLEASE PRINT YOUR NEW CONTACT INFORMATION:

Name/Contact: ________________
Position/Title: ________________
Company: ________________
Address: ________________
________________ Postcode________
Tel: BH ____________ AH ____________
Fax: BH ____________ AH ____________
email address: ________________

AUUG Secretariat Use

Date: ____________
Initial: ____________
Date processed: ____________
Membership # ____________

MEMBERSHIP APPLICATION
INSTITUTIONAL

AUUG Inc
UNIX® AND OPEN SYSTEMS USERS

To apply for AUUG membership, complete this form and return it with payment in Australian Dollars to:

REPLY PAID 66, AUUG MEMBERSHIP SECRETARY,
P.O. BOX 366, KENSINGTON, NSW 2033, AUSTRALIA
Tel: +61 2 361-5994 or 1 800 625 655 • Fax: +61 2 332-4066

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

NOTE: Please do not send purchase orders - perhaps your purchasing department will consider this form to be an invoice. Foreign applicants please send a bank draft drawn on an Australian bank.

We, ______________________________
(Company Name)

do hereby apply for:

Renewal/New* Inst. membership of AUUG	☐ $350.00
International air mail	☐ $120.00
TOTAL REMITTED	AUD$________

(Cheque, money order, or credit card)

I/We agree that this membership will be subject to the rules and by-laws of AUUG as in force from time to time, and that this membership will run from time of joining/renewal until the end of the calendar or financial year.

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

Signed______________________ Date ________

Title ______________________

INSTITUTIONAL MEMBER DETAILS:
To be completed by institutional members only.

Following are our specified contacts. The primary contact holds the full member voting rights. The two designated reps will also be given membership rates to AUUG activities including chapter activities. By default a regional chapter will be selected for you. If you would rather nominate a chapter, please specify in space provided (indicate NONE for no chapter). (Please print clearly or type)

Primary Contact______________________

Position/Title ______________________

Address ______________________

______________________ Postcode ________

Bus. Tel: ____________ Bus. Fax: ____________

e-mail address ______________________

Local Chapter Pref. ______________________

1st Rep.______________________

Position/Title ______________________

Address ______________________

____

Bus. Tel: ____________ Bus. Fax: ____________

e-mail Address ______________________

Local Chapter Pref. ______________________

2nd Rep.______________________

Position/Title ______________________

Address ______________________

____

Bus. Tel: ____________ Bus. Fax: ____________

e-mail address ______________________

Local Chapter Pref. ______________________

Please charge $____________ to my

☐ Bankcard, ☐ Visa, ☐ Mastercard

Account number:______________________

Expiry date:______________________

Name on card:______________________

Signature:______________________

AUUG Secretariat Use

*Chq: bank*____________ *bsb* ____________

a/c ____________ *#* ____________

Date: ____________ *$* ____________

Initial: ____________

Date processed: ______________________

Membership # ______________________

AUUG Inc. as a user group, exists to provide UNIX® and open systems users with relevant and practical information, services, and education through cooperation among users.