

AUUGN

AUUG Inc. Newsletter

Volume 14, Number 1

February 1993

The AUUG Incorporated Newsletter

Volume 14 Number 1

February 1993

CONTENTS

AUUG General Information	3	
Editorial	5	
Letters to the Editor	6	
AUUG Institutional Members	7	
AUUG President's Report	9	
ABCs of UNIX	<i>Andrew Phillips</i>	11
AUUG'93 Preliminary Announcement and Call for Papers	12	
SESSPOOLE	15	
WAUG and Perth News	16	
WAUG Meeting Reviews	17	
Open System Publications	19	
ACSnet Survey	20	
AUUG Book Club - Order Form	23	
Report on Usenix, San Diego Jan 1993	<i>Greg Rose</i>	24
!AUUGN - from AUUGN Volume 1, Number 1		
UNIX in a Hostile Environment or Mouse Watching by a Cat	<i>Peter Ivanov</i>	29
Cost Savings through Standards	<i>Rolf Jester</i>	32
Graphic User Interfaces Made Easy? A Tcl/Tk Tutorial	<i>Robert Biddle</i>	54
From login: - Volume 17, Number 5		
An Update on UNIX-Related Standards Activities	71
From login: - Volume 17, Number 6		
An Update on UNIX-Related Standards Activities	80
Calendar of Events	85
From EurOpen - Volume 12, Number 3		
USLE Column	86
OSF Column	89
Summary of Management Committee Minutes - 4th December 1992	92
AUUG Membership Categories	94
AUUG Forms	95

Copyright © 1993 AUUG Incorporated. All rights reserved.

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

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

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

AUUG General Information

Memberships and Subscriptions

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

Membership and General Correspondence

All correspondence for the AUUG should be addressed to:-

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

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

AUUG Business Manager

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

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

AUUG Executive

President	Phil McCrea <i>phil@softway.oz.au</i> Softway Pty. Ltd. 79 Myrtle Street Chippendale NSW 2008	Vice-President	Glenn Huxtable <i>glenn@cs.uwa.oz.au</i> University of Western Australia Computer Science Department Nedlands WA 6009
Secretary	Peter Wishart <i>pjw@lobo.canberra.edu.au</i> EASAMS Australia Level 6 60 Marcus Clark St. Canberra ACT 2600	Treasurer	Frank Crawford <i>frank@atom.ansto.gov.au</i> Australian Supercomputing Technology Private Mail Bag 1 Menai NSW 2234
Committee Members	Rolf Jester <i>rolf.jester@sno.mts.dec.com</i> Digital Equipment Corporation P O Box 384 Concord West NSW 2138		Chris Maltby <i>chris@softway.sw.oz.au</i> Softway Pty. Ltd. 79 Myrtle Street Chippendale NSW 2008
	John O'Brien <i>john@wsa.oz.au</i> Whitesmiths Australia P/L #5 Woods Centre ANSTO Business & Tech. Park Lucas Heights NSW 2234		Michael Paddon <i>mwp@iconix.oz.au</i> Iconix Pty Ltd 851 Dandenong Rd East Malvern VIC 3145
	Greg Rose <i>ggr@acci.com.au</i> ACCI 723 Swanston St Carlton VIC 3053		

AUUG General Information

Next AUUG Meeting

The AUUG 1993 Summer Conference Series are to be held between February and April 1993.

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

Advertising

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

Mailing Lists

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

Back Issues

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

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

Conference Proceedings

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

Acknowledgement

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

Disclaimer

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

AUUG Newsletter

Editorial

Welcome to AUUGN Volume 14 Number 1. Some members might remember we ran a competition for a new cover design for AUUGN last year. As there was no response, Frank and I have modified the cover design and this is what will be used in future issues of AUUGN, however further comments are welcome.

In this issue we have a number of interesting articles: The AUUG President, Phil McCrea talks about the Novell takeover of USL, Greg Rose reports on the Usenix conference, and Adrian Booth on the WAUG meetings. Two papers have been included, one that was presented at AUUG'92 but did not appear in the proceedings and the other which was presented at UniForum NZ '92.

Once again I have received a letter from one of the members, commenting on an article published in the last issue of AUUGN. Andrew Phillips also has given me his version of the ABC's of UNIX.

Unfortunately, we are having a few problems with book reviews, hopefully these will be sorted out soon and we should have some for upcoming issues of AUUGN.

Finally, on other news, the AUUG Member handbook has been printed and you should have received a copy.

Jagoda Crawford

P.S. Don't forget the summer conferences!

AUUGN Correspondence

All correspondence regarding the AUUGN should be addressed to:-

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

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

AUUGN Book Reviews

The AUUGN Book Review Editor is Dave Newton. David has no network access at present, so please contact the AUUGN editor for more details. A number of books are available for review, please keep an eye on aus.auug for books available.

Contributions

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

Volume 14 No 2	Friday 26th March
Volume 14 No 3	Friday 28th May
Volume 14 No 4	Friday 30th July
Volume 14 No 5	Friday 24th September
Volume 14 No 6	Friday 26th November

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

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

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

James Sainsbury
1 Meranti St
Crestmead Q4132
(07) 834 2833 (wk)
(07) 200 9007
31 January 1993

The AUUGN Editor
PO Box 366
Kensington NSW 2033

Dear editor,

I should like to offer a correction to Doc Strange's column on pages 75-79 of AUUGN 13(6) if it has not already been made.

The infelicity of `_exit(2)` over `exit(3)` after a failed `execve(2)` is not only sanctioned by the manual pages for `vfork(2)` of SunOS 3.2 through SunOS 4.1 but actually mandated^{1,2}.

The library function `exit(3)` flushes all stdio stream buffers and closes all open file descriptors before calling `_exit(2)`.

In the case of `vfork(2)` the parent exchanges its address space with the child's then minimal address space and blocks until the child calls either `execve(2)` or `_exit(2)` when the parent reclaims its original address space (hopefully unaltered)³.

If a child's attempt to `execve(2)` an image fails and then calls `exit(3)` the parent's buffers will be flushed and file descriptors will be closed by the child and the parent will reclaim an altered address space.

The `vfork(2)` manual page actually offers the advice that it is also an error to call `exit(3)` after a `fork(2)` as buffered data would be flushed twice (subtle).

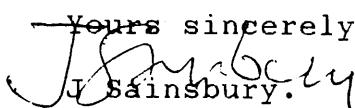
On a different tack one might have been lead to believe from Doc Strange's column that SunOS never had any bugs — maybe not. I can recollect strange messages coming from `csh(1)` in SunOS 3.5 mentioning `vfork(2)` and a definite recollection of a Sun-3/50 kernel panic after a pointer infidelity in a user program had caused one of the above `vfork(2)` messages. I definitely remember under SunOS 3.5 "cat /etc/termcap > /dev/printer" causing a panic.

Perhaps the poor soul that wrote the offending code was trying to work around a SunOS bug that did then exist but which he/she did not understand or correctly localise but from the `vfork(2)` evidence the she/he did at least RTM (I would be the last person to suggest a certain columnist do the same :)).

¹ p147 UNIX Interface Reference Manual, SunOS 3.2 (1986)

² p878 SunOS Reference Manual, SunOS 4.1 (1990)

³ pp138-140 *The Design and Implementation of the 4.3BSD UNIX Operating System* — Leffler et al., Addison-Wesley 1989.

Yours sincerely

J Sainsbury

AUUG Institutional Members as at 12/02/1993

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

AUUG Institutional Members as at 12/02/1993

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

AUUG President's Report

Has UNIX's soul been sold?

The decision by Novell to acquire USL is an interesting one and raises a myriad of issues. I've read numerous commentaries on the matter, and depending on the background of the writer, the responses are varied.

These responses fall into various categories. I've summarized these below, and added a few comments from my own perspective.

Competition for NT

USL already has an alliance with Novell. The two companies put together Univel last year to integrate SVR4.2 and Netware, and create *Unixware*. The product is now becoming available through existing Novell distribution channels.

The new arrangement gives a great deal of marketing clout to Univel – it is not just a joint venture which Novell has an interest in: it is now mainstream Novell business. The name Univel may well disappear, but it certainly makes UnixWare a much more serious contender for the desktop, particularly when the availability of NT keeps slipping.

Novell clearly has Microsoft in its sights, and, since Novell has the lion's share of the LAN market, Microsoft certainly has due reason to be concerned.

Unity of the UNIX market

As a result of AT&T's acquisition of Sun shares in 1988 (thereby appearing to favour Sun in the time-to-market stakes), the UNIX industry aligned itself around 2 well defined camps – the Open Software Foundation, and Unix International. This fragmentation has set UNIX back considerably.

Now that AT&T will be out of the picture altogether, it may provide the catalyst for a reconciliation of the two camps. A certain amount of reconciliation has already occurred, with OSF and USL agreeing to a common operating system interface, and cross licensing technologies.

It is certainly in Novell's interests to have a strong common UNIX, to present a more united front against Microsoft. It might just work too.

Loss of independence for UNIX

Concern has been expressed that Novell will manipulate the direction of SVR4 to suit its own needs (and who could blame them, given their \$350M outlay). The majority of computer manufacturers use USL technology in their operating systems, so this is a valid concern.

Both USL and Novell have attempted to put cold water on this problem by saying that SVR4's future will not be affected adversely by the Novell takeover.

Whatever happens, there will be a temptation to portray SVR4 as a proprietary operating system, whose future will be subject to the whims of a networking software company.

Role of Unix International

An issue related to the above is the future role of Unix International. UI has served a useful role in providing requirements to USL, by means of special interest committees, for the on-going development of SVR4. Whilst USL have been under no real obligation to implement these recommendations, it appears that most of the the UI recommendations are being implemented by USL, or are planned to be.

How UI will operate under Novell ownership is not clear. There are 2 possibilities:

- a. The role of UI will be strengthened, at least as far as producing requirements to Novell for SVR4 development. This will alleviate the fears outlined above.
- b. The role of UI will be weakened because, with the removal of AT&T from the scene, there may be a newly-found unity amongst UNIX vendors (see below).

My personal hope is that the former is the case, and furthermore that IBM, HP, and DEC contribute to producing these requirements.

OSF and UI relationship

As alluded to above, USL's new owners may bring a unity to the politics of the UNIX industry. Novell is a member of OSF, and this will certainly help bring OSF and USL (as technology providers) closer together.

A second order effect of this may be to introduce a truce between those old political sparring partners, OSF and UI. Most of the reasons for their *political* sparring will be removed, particularly if UI changes its focus, as outlined above, to be a specifications producing organization only, cooperating closely with organizations such as X/Open.

My personal hope is that such a truce occurs.

OSF is a beneficiary

Some commentators have claimed that OSF/1 is now the only independent open systems platform (and there is some truth in this). DEC have certainly highlighted this view, given that they are the computer manufacturer most committed to OSF/1.

OSF may well benefit in some respects. However it is unlikely that the new ownership of USL will cause many existing SVR4 licensees to suddenly switch operating system technologies.

OSF will be affected adversely

Another view being aired is that Netware will be strengthened in the marketplace, which will provide more competition for DCE. Currently, despite the fact that no single manufacturer is actually yet shipping DCE products, it is a widely accepted view that DCE will become the dominant 'middleware' technology. Certainly USL have recognized this, and will be licensing DCE themselves.

Sun will be a loser

Sun have reason to be concerned, since the new arrangement means that they have a much stronger competitor in the desktop market. Moreover, since Sun licenses its operating system technology from this competitor, Sun's concerns are compounded, particularly in relation to the potential delay at getting its products to market.

This is similar to the 1988 situation which gave birth to the formation of UI and OSF. Will we see Sun turning to OSF?? Unlikely! But you can bet there will be high level discussions between Sun and Novell.

It all highlights the need for an organization like UI to be a 'broker'.

IBM is a beneficiary

IBM is an onlooker at this point in time, and has put its eggs into the Taligent basket. This combined IBM/Apple project will produce yet another operating system candidate for the desktop, but is quite some time off.

IBM has its sights set on Microsoft, and will surely benefit from a divided desktop marketplace, where Microsoft and Novell are slogging it out – the well known 'divide and conquer' approach (there are those who claim that IBM has used this approach by actively backing OSF...).

AT&T is a beneficiary

By establishing this equity relationship with Novell, AT&T has effectively extended its sphere of operation into the LAN market area, where Novell is the dominant provider. In the converging world of communications and computing, AT&T is now very well placed, with activity in wide area communications, local area networks, and computer nodes (NCR).

It is interesting to compare AT&T and IBM, who are similarly sized companies. IBM's woes have been well publicized, and are generally attributable to IBM's focus on market segments where growth is flat (at best). By contrast AT&T is now positioned to take advantage of the high growth areas of the IT industry.

What do the UNIX creators think?

Dennis Ritchie's only response was to quote the following from Genesis 25, 31-34:

And Jacob said, Sell me this day thy birthright. And Esau said, Behold I am at this point to die, and what profit shall this birthright do to me? And Jacob said, Swear to me this day; and he swore unto him: and he sold his birthright unto Jacob. Thus Jacob gave Esau bread and pottage of lentils; and he did eat and drink, and rose up, and went his way: thus Esau despised his birthright.

Putting it all together

On balance, the Novell takeover of USL will have a positive effect on the Open Systems world. Currently UNIX is the only basis for Open Systems. Microsoft's implicit aim is to mostly ignore accepted industry standards, and to force its own interface standards on the world by sheer marketing muscle. Anything which counteracts this must be regarded as a good thing.

P. McCrea

The ABCs of Unix

The following has been supplied by Andrew Phillips, in response to the "The ABC's of UNIX" published in the last issue of AUUGN.

A is for ar, storing things in one place,
B is for banner, using far too much space.

C is for curses, making all screens the same,
D is for dump, ignore if you're game.

E is for echo, which repeats quite precisely,
F is for file, which classifies nicely.

G is for grep, regularly finds without fail,
H is for head, the inverse of tail.

I is for init, from which all are begotten,
J is for join, which is sadly forgotten.

K is for kill, which reduces the load,
L is for lint, which cleans all your code.

M is for make, which relies on the time,
N is for nroff, making words look sublime.

O is for od, for a different, odd view,
P is for patch, to make everything new.

Q is for (SIG)QUIT, a way to dump core,
R is for rm, you know what that's for.

S is for strings, for those nosy folk,
T is for tee, two paths at a stroke.

U is for UUCP, connecting hundreds of nodes,
V is for vi, which has different modes.

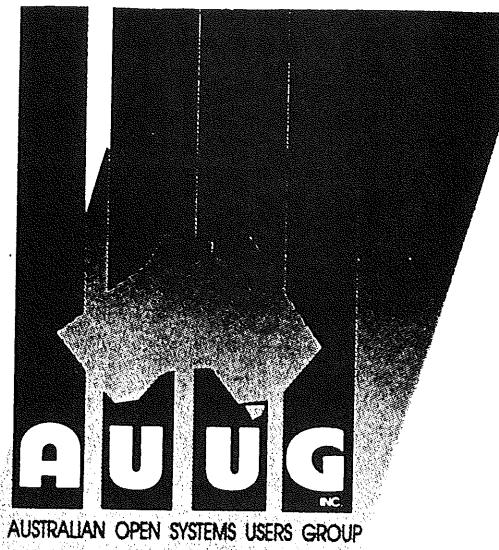
W is for wc, which counts words and lines,
X is for xargs, for building command-lines.

Y is for yes, producing output never-ending.
Z is for zoo, to make it cheaper when sending.

Andrew.
andrew@teslab.lab.oz.au, (02) 487 3267

AUUG '93

Darling Harbour, Sydney, Australia, September 27-30



1993 Preliminary Announcement and Call for Papers

AUUG, Inc., forum for UNIX® Open Systems Users Presents:

"Results through Open Systems."

Over the past several years we have heard about 'What are Open Systems', and 'Maintaining Control with Open Systems'. Now it's time to hear about the results which have been achieved. Rapid expansion, the challenge of integration, global networking, and security are all issues of importance and concern to users around the world. AUUG '93 solicits papers on all aspects of UNIX and open systems, and particularly on successful applications and implementations of open systems technology to age-old and newly emerging problems.

Events:

AUUG '93 will be a four day conference, commencing September 27, 1993. The first day will be devoted to tutorial presentations, followed by three days papers, work-in-progress sessions and BOFs.

Tutorials:

Provisions for two full-day tutorials and up to eight half-day tutorials have been made. These sessions, typically in a lecture format, are targeted to educate the audience and arm them with innovative "how to" lessons. Please submit tutorial abstracts, along with preference for a half- or full-day slot to address below.

Papers:

AUUG '93 provides dual Technical and Management tracks for the presentations.

To share your innovative implementations, applications, and similar areas submit your abstract for the technical track. We are also interested in your experiences, case studies, strategic issues, and the like. If your topic better fits these areas submit your abstract for the Management track.

The above should not, of course, discourage papers which are appropriate for both audiences at once.

Vendor product announcements will be automatically rejected unless specifically submitted for the special advertising stream.

Prize for the Best Student Paper:

A cash prize of \$500 will be awarded for the best paper submitted by a full-time student at an accredited tertiary education institution. In addition, the ten 'runners-up' will be rewarded with free registration.

Work-in-Progress and Advertising Sessions:

These brief 15 minute sessions are designed to report on current work with fundamental aspects highlighted. New to the AUUG conference are the Advertising sessions. These are devoted to new products only. Product specification sheets should be submitted with your abstract.

Birds-of-a-Feather Sessions (BOFs):

Are you interested in discussing particular problem areas, sharing arcane on favourite programs, using the internet, or other controversial topics? During the lunch hour and at the end of each presentation day, one hour time slots for BOFs will be available. We distinguish two types of BOF; general interest and vendor sponsored. Please contact the Programme Committee if you would like to organise a Birds-of-a-Feather Session. There may be some facilities charge to vendor sponsored events.

Speaker Incentives:

Presenters of papers are afforded free conference registration. Tutorial presenters will receive 25% of the profit for their session and a free conference registration.

Form of Submissions:

Please indicate whether your submission is relevant to the technical or management audiences, or both. In either case, submissions are required to be in the form of an abstract and an outline. Please provide sufficient detail to allow the committee to make a reasoned decision about the final paper; of course a full paper is also perfectly acceptable. A submission should be from 2-5 pages and include:

1. Author name(s), postal addresses, telephone numbers, FAX and e-mail addresses.
2. A biographical sketch not to exceed 100 words.
3. Abstract: 100 words
4. Outline: 1-4 pages giving details of the approach or algorithms pursued. Shorter outlines will not give the programme committee enough information to judge your work fairly, and, in most cases, this means your paper will be rejected. Longer outlines and full papers simply cannot be read by the committee in the time available. However, you may append a full paper to your outline; this is sometimes useful during evaluation.
5. References to any relevant literature
6. Audio-visual requirements
35 mm slides are preferred, however, overheads will be accepted.
Hand written or typewriter generated overheads will not be accepted.

Acceptance:

Authors whose submissions are accepted will receive instructions on the preparation of final papers for inclusion in the conference proceedings, and the format requirements for slides.

Programme Committee:

Piers Lauder - Sydney University (Chair)
Liz Fraumann - AUUG
Ian Hoyle - BHP Research Labs
Hugh Irvine - connect.com
Rolf Jester - Digital Equipment Corporation
Bob Kummerfeld - Sydney University
Phil McCrea - Softway P/L
Andrew McRae - Megadata P/L
Greg Rose - Australian Computing and Communications Institute

Relevant Dates:

Abstract and outlines due: April 6, 1993
Notifications to authors: April 26, 1993
Final Papers due: July 26, 1993

Addresses:

Please submit one hard copy and one electronic copy (if possible) to the addresses below:

e-mail: auug93@cs.su.oz.au

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

AUUG '93 Programme
P.O. Box 366
Kensington, NSW 2033

Tutorial abstracts to: ggr@acci.com.au

Please be sure to include your complete contact information (phone, fax, postal code and electronic mail addresses) in all correspondence.

UNIX is a registered trademark of UNIX System Laboratories in the United States and other countries.

SESSPOOLE

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

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

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

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

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

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

Wednesday, 17 March 1993

Thursday, 29 April 1993

Tuesday, 8 June 1993

Wednesday, 21 July 1993

Hope we'll see you there!

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



Moving to OPEN Systems?

Softway is Australia's largest UNIX systems software house.

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

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

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

79 Myrtle Street
Chippendale NSW 2008

PO Box 305
Strawberry Hills NSW 2012

WAUG and Perth News

Adrian Booth, who often reviews WAUG's meetings for our newsletter YAUN, has kindly agreed that AUUGN may print his reviews. This frees me to write about other things.

At the moment there's more going on than meetings and newsletters. WAUG is about to decide whether to become a formal chapter of AUUG, rather than the affiliated but separate group it is now. A number of members of WAUG, including myself, think this is a good idea, because AUUG's Perth members need a local AUUG chapter and WAUG is in an ideal position to fulfill that need. I'm told the AUUG committee thinks so too.

WAUG's March meeting will include a Special General Meeting at which our membership will vote on the issue. The vote is necessary because becoming an AUUG chapter means winding up WAUG as it currently exists; it means that WAUG membership will become AUUG membership, which costs more; and we already know that not everyone wants to do it (perhaps some people are turned off by the idea of associating with non-Eagles-supporters:-).

At last night's WAUG committee meeting (on 27 January) we passed a motion resolving to hold the Special General Meeting. This motion establishes the timetable and procedures for holding the vote, and defines another motion on which the Special General Meeting will vote. Although last night's motion was eventually passed 8 to 1, the opposition to it was strong enough to make the committee meeting pretty tortuous. (Don't ask me why there was opposition to asking the members to vote!)

Today I'm feeling most relieved that the motion was passed, so that the members get to vote at last, because we've been discussing this issue a long time. I believe it was Glenn Huxtable who originally suggested, about two years ago, that WAUG become an AUUG chapter. The issue has finally come to a head now that AUUG is actively establishing local chapters. I'm glad that the folks in Canberra (COSUG) decided to get on with it, so that AUUG's chapter policy had to be decided, otherwise WAUG might still be discussing it.

I really can't say what the outcome will be, but I hope WAUG does become our local AUUG chapter. I would like to have the local contact of WAUG combined with the national strength of AUUG. I would like to be in one group instead of two. And I'm sure WAUG's members would enjoy the extra benefits that only a nationwide group can provide.

I'll let you know what happens.

To another topic. Many members of WAUG, as well as Perth AUUG members, are looking forward to the 1993 Perth AUUG Summer Technical Conference on 16 April. Adrian Booth is organising this and seems to be doing a fine job. There will be "imported" speakers, including Chris Schoettle and Greg Rose, as well as local ones. There will also be tutorials.

Another thing that I guess is worth a mention is that WAUG's AGM will be on the third Wednesday in May, in conjunction with the usual technical meeting. It's usually in April, but we may decide not to have a meeting in April because it would be so close to the AUUG conference.

If you're interested in joining WAUG (the Western Australian Unix systems Group) or contributing to our newsletter YAUN (Yet Another Unix Newsletter), our postal address is PO Box 877, WEST PERTH WA 6005. And for those already living in the 21st century we now have an email address: waug@uniwa.uwa.edu.au. There's also yaun@uniwa.uwa.edu.au, for newsletter-related mail.

If you'd like to speak at a WAUG meeting or have an idea for a speaker, please contact our Meeting Organiser, Mark Baker, at waug-meetings@uniwa.uwa.edu.au or on (09) 420 6813.

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

WAUG Meeting Reviews

December

A Quick Xmax Romp

Paul Curtis, Engineering Manager, telinit 2

The December meeting saw a good turnout despite (or because of) the fact that it was going to be a technical meeting and not a Christmas social function like the ones we have had in past years.

Paul gave an overview of X Windows before dazzling us all with the specifications of Labtam's new X terminals.

X Windows comes from MIT, and is derived from the W windowing system. X Windows first came out in 1984. Many revisions have followed since then. The current revision is X11R5 (X Windows version 11, release 5).

X Windows has become a *de-facto* industry standard which is supported by the *X Consortium*, which includes companies like DEC, HP and Sun.

Paul contrasted the "old" timesharing model of one CPU with n terminals with the "new" client/server model all the glossy magazines are promoting as the latest in thing: essentially 1 user who has n CPUs available to him/her.

The three main goals of MIT in the development of X Windows were:

- Provide windows on bitmapped terminals in a portable manner that still allowed high performance
- Allow interconnectivity between terminals and machines
- Provide *mechanism, not policy*

Paul detailed four main methods for connecting to X Windows:

- Through a PC using a software package such as PC/XView or XVision
- Through a PC using a special hardware card
- Using an X terminal
- Using a workstation with X Windows capability and software - ranging from 386/486-based machines running UNIX through to "real" workstations

Paul debunked the myth that PC access to X Windows is the cheapest route, discussing aspects such as the limitations of the ISA bus and the comparatively poor quality of monitor technology typically used on PCs.

In fact, an excellent option is to buy a Labtam X terminal from Paul. These have very impressive specs - up to over 115,000 Xstones. They achieve this performance using the Intel i960 (80960) CPU, which HP have apparently recently adopted for use in their X terminals.

Labtam have now moved from the older KB version of the i960 to the new CA series, which features a 32 bit parallel architecture and executes 2 - 3 instructions per clock cycle. It has a burst-mode bus, on-chip register and instruction caches, and four on-chip DMA channels (allowing up to 59Mbyte/sec transfer rates), and can access memory at 132Mbytes/sec.

The same architecture should be or is capable of 250,000 Xstones.

Paul gave a brief overview of getting X terminals running. The main prerequisites are ethernet and TCP/IP. It is very important to know which version of X you are running and to consult the appropriate documentation for your version.

Other hints Paul briefly mentioned:

Running a local window manager within the X terminal means that there is more memory available for your applications.

Use *xdm* if you have it. Run *strings(1)* on the binary to check for any hard-coded directory names, and edit the *xdm config* file (usually */usr/lib/X11/xdm/config*) to alter these as required. *xdm* logs errors to an *xdm-errors* file, which you can peruse if you are having problems.

After Paul finished his talk, Glenn announced that Sam Pascoe had become the proud father of a baby girl, and would be putting some cash on the bar. In a fit of generosity, Glenn matched Sam's amount as a celebration of his and Janet's recent wedding.[†] Paul then announced that he had intended to do the buying, and would add his contribution to Sam's and Glenn's. A very long (and noisy) crowd stayed later than usual to make sure that these weren't wasted.

January

WAIS - Wide Area Information Systems

Todd Hooper, Random Access

Todd overawed the audience before he even started his talk with his projection equipment: an Apple PowerBook with an external colour LCD screen sitting on the overhead projector. This equipment was used to display high-quality colour slides during the talk.

Todd's talk concentrated on the origins, use and implementation of WAIS.

The impression I got is that WAIS aims to provide easier and more efficient access to the vast amounts of information available on the Internet. The main goals of the WAIS project are:

- Create an open architecture of information servers and clients
- Derive and standardise a computer-to-computer protocol that enables users to find and question servers
- Use current technology
- Provide information services that allow a workstation to act as an information librarian, keeping its information up-to-date automatically

WAIS originated at Thinking Machines Corp[†], a supercomputer manufacturer. Among other applications, Dow Jones are trialling a Connection Machine-based WAIS. Apple are also involved through their *Rosebud* project.

WAIS clients can be Macs, X terminals, PCs, or even ASCII terminals. The WAIS server is usually a Connection Machine or a UNIX platform. The clients and servers communicate over local or wide-area (e.g.: Internet, AARNet) networks, or over serial lines via modem. Gopher clients can also access WAIS by tunnelling.

WAIS is based on ANSI standard Z39.50 - no-one seems to know what this standard is, but it defines an interface to a remote information retrieval service.

Clients queries can specify information sources (e.g. *The New York Times* or the King James Bible) and keywords that define the area of interest. The client constructs a query locally and passes it to the server. The server performs a search of the appropriate sources and sends matching information back. This information is sorted by its "relevance" to the query based on word- and pattern-matching heuristics.

As opposed to existing database technology - which is highly interactive (each with a different interface); doesn't take advantage of client CPU power; provides little relevance feedback, and has poor connectivity - WAIS uses short, bursty communication; provides a consistent user interface; uses the power of the local workstation, and uses a standardised protocol.

The future of WAIS includes commercial WAIS databases, better client and server software, client software on more platforms, improves search facilities, and broader integration of sound and vision into WAIS databases.

Todd's talk - while a bit fast for someone who doesn't know shorthand to take notes - was an exciting glimpse into the future of information services, and one which the audience obviously appreciated.

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

[†]A full transcript of their later conversation after Glenn's unilateral decision is available for purchase, unless Glenn and Janet have already paid the extortion demands. :-)

[†]The motto and goal of Thinking Machines is "to make a machine that would be proud of us".

Open System Publications

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

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

For example:

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

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

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

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

Following is a list of prices† provided by UniForum.

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

† Prices in US dollars

(*) please specify issues

ACSnet Survey

1.1 Introduction

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

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

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

Please send replies to:

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

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

Technical enquiries to:

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

Thank you

=====

1.2 Contact Details

Name: _____
Address: _____

Phone: _____
Fax: _____
E-Mail: _____

1.3 Site Details

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

New Connections

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

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

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

- A13. Once connected, are you willing to provide additional connections? Yes No
- A14. Additional Comments:

Existing Sites

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

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

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

AUUG BOOK CLUB & PRENTICE HALL AUSTRALIA

20% DISCOUNT TO AUUG MEMBERS

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

Adams/ Writing UNIX Device Drivers

A detailed discussion of device driver architectures, template-based implementation methodology, functional tools and sample device drivers. A reference for experienced programmers.

ISBN: 0139638695 Paper 1993 RRP \$59.95*

Perlman/ UNIX for Software Developers

A step-by-step tutorial to the tools and techniques popular on UNIX systems. Filled with templates for programs and shell scripts; diagrams and worksheets throughout.

ISBN: 0139329978 Paper 1993 RRP \$44.95*

*Deduct 20% from listed retail price

Name: _____ Organisation: _____

Address: _____
(Street address only) Telephone: _____

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

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

Please charge my: Bankcard Visa MasterCard

Credit Card No:

Expiry Date: _____ Signature: _____

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



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

A.C.N. 000 383 406



Prentice Hall Pty. Ltd.

7 Grosvenor Place, Brookvale NSW 2100.

Tel: (02) 939 1333 Fax: (02) 905 7934

A Paramount Communications Company

Report on Usenix, San Diego 25th-29th January 1993

Greg Rose

Australian Computing and Communications Institute

I arrived in San Diego from New York on Saturday. The main reasons for me to arrive early were firstly, because the Usenix Board of Directors meeting was scheduled for Sunday, and secondly, given the choice between New York or San Diego in winter, well...

1. Sunday... Board Meeting.

Attendance at Usenix's "big" conference is declining, and this is seen to be a worrying thing, but it is most likely to be because the associated smaller conferences are doing extremely well. For example, the Large Installation System Administration conference, and the C++ conference, are both about half the size of the main conference. The program for this particular conference is seen to be extremely strong, especially with a number of great talks from Australia.

Elizabeth Zwicky reported about SAGE (the System Administrators' Guild), Local Technical Groups (Regional groups), Special Technical Groups (SIGs), and International Affiliate Groups (eg. Sage-AU). More about Sage later; for now suffice it to say that Usenix is happy that Sage-AU is up and running, and wants to cooperate.

Usenix had planned an Application Development Symposium, but that was cancelled. It was supposed to be a joint venture with UniForum Canada, and the logistics got too hard, but there was a reasonable amount of interest. "Symposium" was probably too high-falootin a word for this thing. This is something I think AUUG could do well.

Finally, after a number of years of being shunned, a change of management at UniForum is now amenable to getting friendly with Usenix again. Initially, this cooperation will probably take the forms of jointly funding the POSIX standards watchdog function, and hosting a major Usenix event (possibly the next LISA) at the same time and place as the UniForum in San Francisco, in early 1994.

Personally, I think these previous points show that AUUG was right to try and avoid splitting into technical versus commercial subgroups. UniForum has bet their farm on Unix, and really are not in a position to move towards other "open" systems. Usenix on the other hand have diversified quite successfully, but rely probably too much on the moral and technical high ground. Now they seem to need each other again. (End personal opinion.)

Mick Farmer reported on the new structure of EurOpen. Like most things in Europe, this group (formerly the European Unix Users Group) is undergoing rapid and disruptive change. EurOpen was structured as an umbrella organisation, but also undertook to organise their own conferences to some extent. The Secretariat was moved from Owles Hall in England to Brussels, Belgium, and a bunch of employees were terminated at the same time. A couple of member countries refused to pay their dues, and EurOpen was looking at bankruptcy this year. So, a quick restructuring resulted in "EurOpen Lite" which seems to merely coordinate and distribute information between the member country's groups (and no longer competes with them). It is too soon to say whether this will succeed or not, but it is a valiant effort to save the otherwise doomed organisation. AUUG seems better all the time...

Usenix's Board were all surprised to hear that 32% of their members were from overseas (which doesn't include Canada). Peter Collinson (U.K.) and I, who regularly attend these meetings, had a bit of a giggle about this, as we have certainly known it for some time.

When the board meeting moved into closed session (i.e. they threw out the hangers-on like me) I went around to the registration area and picked up my books etc. It was within the first few minutes that I realised what the really hot topic of this Usenix conference would be.

Background: Unix System Laboratories is sueing the University of California at Berkeley, and Berkeley Software Design, Inc., over some sort of alleged infringement of licenses or copyrights or look and feel, or something equally intangible. In the latest round of this war, USL required a list of all people who have ever worked for a Unix Source code licensee, or who have ever seen source code of Unix or of the Berkeley NET-2 (re-written) release, or who have read anything about internal design or data structures

of Unix (ever read the Bach Book, or the Lions Commentary?). The reason for this request is that USL asserts that any code such a person writes for an operating system, or in which the same algorithms are used (e.g. linear search, used extensively in Unix) is really the property of USL. Their deposition used the words "mentally contaminated" to describe such people. The court is currently deciding the status of this suit. (End of Background.)

Rick Adams, who works for BSDI and UUNET, has made a bunch of badges which say "MENTALLY CONTAMINATED" in large red letters, and I happened to have a pocket full that I was intending to return to Australia. Virtually everyone who saw the badge said that they were contaminated, and so wanted such a badge. I ran out within minutes. (Don't worry, I got more.) Peter Salus was wearing the UKUUG windcheater with the famous "/* you are not expected to understand this */" comment, and pointed out that anybody who looked at him was also forever contaminated. (In fact, if you just read that, you are now contaminated too... fortunately most of this audience is in Australia and we appear to be sensible enough that we can ignore this stupidity. The judge here (the U.S.) is expected to rule on this point after about two weeks of deliberations, and no one can guess which way it will go.)

The "launch" of the week was held at 18:00 and featured non-alcoholic drinks free, a cash bar, and some quite good (and sometimes very spicy) Mexican munchies.

The conference T-shirt features a standard sort of San Diego advertising picture, with a caricature of Rob Kolstad (Conference Convener) surfing on an oversized keyboard. Tomorrow night there is a group getting together to sew pink tutus around him (sorry, an "in" joke -- it was proposed on the network that there should be a competition about Rob and tutus).

2. Monday... Tutorial day 1.

Not much to say about this day. There are about ten full day tutorials each day, and I attended the one about OSF DCE. The standard of the tutorials is pretty high, even if the information is something you would rather not hear. DCE, despite its total lack of real availability at this point in time, is already an important standard. Fortunately I was already familiar with all seventeen of the possible synchronisation primitives which are all supported by DCE, so I floated around meeting people for the morning and only sat in on the DCE tutorial for the last half. Dinner was a rather nice Mexican meal from somewhere north of UCSD.

3. Tuesday... Tutorial day 2.

Relatively little to report on this day, as I decided that I needed to fill in the gaps in my knowledge of Kerberos, and the tutorial by Dan Geer and Jon Rochlis was excellent. Tuesday evening was the board meeting for USENIX/SAGE which I attended representing Sage/AU.

By the end of this day, my conference badge was getting rather heavy. As well as the two ribbons I deserved (Invited Speaker and Conference Pilot) I had added a "Newcomer", as it was such a pretty yellow. I had a dinosaur sticker, the "MENTALLY CONTAMINATED" warning mentioned above, and a similar pin in the form of a New Hampshire number plate, saying "NET2", and with an insertion mark and AT&T inserted in the motto, vis "Live (AT&T) Free or Die". At the end of each ribbon was a button. One was the "Robbie Kolstad fan club", another a Henry Spencer badge quoting Marshall Rose "OSI Committees lack Adult Supervision", and yet another lawsuit badge, "NET2" circled by the words "Want To Be Sued? Ask AT&T How!". This badge was getting pretty heavy. The solution to the problem appears on the next day...

4. Wednesday... Conference Day 1.

The keynote talk was "Pen Based Computing and its Impact", by Robert Carr of GO Corporation, writers of the PenPoint Operating System. This talk was quite interesting, until the last few minutes, but I'm not sure that it deserved the keynote position. I can believe that the interface to the computer through a pen might be more productive for many people, particularly people moving around a lot. The last five minutes of the talk, however, were mind blowing. All the features are usable in Japanese, including the handwriting recognition. Watching a video of Japanese characters being written and then redrawn accurately after being recognised was impressive.

There were two minor problems with the talk. The speaker claimed that PenPoint was the first operating

system to support Unicode, at which Dave Presotto and Phil Winterbottom, the implementors of Unicode in Plan-9, sort of choked, and they called the Hobbit chip from AT&T a "brand new high performance RISC microprocessor", while it is in reality the same old CRISP chip revisited.

Immediately afterward, Rob Pike gave a talk about use of Unicode in Plan-9. I can't reproduce the title here, as it included Japanese and Hebrew versions of "Hello world".

Many of the papers on this day were not of great interest to me, and because of the weight of my badge, I decided to start a new business (profit free) producing funny name badges. Because Rob Kolstad is the current scapegoat, name badges like Greg Kolstad, Ken and Dennis Kolstad, Kirk Kolstad, and even Rob (Pike) Kolstad appeared.

Dinner was an excellent pseudo-Italian meal while Addison Wesley negotiated a contract with me for my new book (that's a plug).

This evening, there was an Open Board Meeting for Sage, in which it was announced that there was an intent to affiliate between Sage/US and Sage/AU. The committee for Sage/US is Steve Simmons (President), Pat Parsegian (Secretary), Peg Schafer (Treasurer), Carol Kubicki, Pat Wilson, and Paul Moriarty. There was an observation that there was an enormous momentum behind the formation of Sage, that the first year was extremely successful, and that there was a great level of excitement about the future of Sage.

Sage has a number of working groups and discussion groups, which are displaying varying degrees of success in achieving their goals, or even specifying their goals. There is now a sub-committee which will review these working groups.

What do you get for your membership in Sage? An investment in the future, and in the working groups. A chance to be heard as a coherent group, without being drowned out by the other voices. Gradually, Sage will take over the Lisa conferences.

5. Thursday... Conference Day 2.

There were a number of interesting papers given, including a talk from Plan 9 about the I/O system. Some people from Bell Labs had been, shall we say aggressive, to some of the earlier presenters, so they came in for some bashing of their own from Peter Honeyman and me.

Dan Klein gave an excellent invited talk about languages for specifying specialised things, and the major example was the specification language for Blazons (medieval shield designs), with parallels to PostScript.

Immediately after lunch there was a highly acclaimed talk about "A History of Unix", and despite lots of people in the audience who had been closer to the events than the humble author, there were few disagreements over facts.

The conference reception was different to previous Usenix events, in fact it was much more AUUG-like, with tables and entertainment. Unfortunately, the impromptu comedy foursome did a pretty poor job of making jokes about relevant issues, and even got Ken Thompson's name wrong (Ritchie Thomas!) in one of the jokes. The jokes that weren't Usenix specific were pretty smutty and sexist, and didn't go over well. I think it will be the last time Usenix tries that idea.

After the reception, and with the pressure of my talk over, I proceeded to get (more) drunk, culminating in the Single Malt Scotch Working Group meeting, which went till well after midnight. That's why the typing is not terribly coherent.

This year, Usenix has introduced an annual "Keepers of the Flame" award, and the first one was presented to the Computer Science Research Group at the University of California at Berkeley. The seven members of the CSRG get rather nice glass sculptures, and corporate contributors get plaques. Then there was a huge list of major contributors who will get calligraphed certificates of appreciation, and Robert Elz was deservedly on this list. Then there was a list of about 160 other significant contributors, with a number of Australians on it. This is a nice idea.

6. Friday... Conference Day 3.

Andrew McRae opened the day, and his talk was well received by those who attended. I was in the parallel invited talks track, where some of the best papers from the filesystems workshop were represented. Peter Honeyman from the University of Michigan was the host and presented one of the papers, about Alex, the NFS FTP filesystem.

On one of the previous days, Margo Selzer presented a joint paper about the results of the final implementation of the Log Structured File System. This has been one of the great hopes on the horizon for improving the performance of file systems, but this paper basically canned it, pointing out that the overheads of garbage collection eventually soaked up most of the benefits for common scenarios.

As an obvious corollary to the promising initial results, this same conference had a number of papers about novel applications of the log-structured filesystems, which were no longer such wonderful things to do...

At lunch time I got back to work and attended a meeting of Usenix's tutorial program committee, to get ideas for tutorials for the coming AUUG conference. (I am tut coordinator for AUUG'93.) I have some good ideas, if people want to present a tutorial but don't know what to talk about.

As I mentioned earlier, there had been some amount of on-going Rob Kolstad bashing, particularly with reference to pink tutus. At the beginning of the after lunch Invited Talks session, which was a representation of the best papers from the last Lisa conference, this was finally laid to rest (perhaps). Steve Simmons began by introducing Rob (whose badge by now said simply "Rob *") and explaining the origin of the whole tutu business. He then said that Rob would be presented with a pink tutu which he could ceremoniously burn, and end the matter. What he didn't mention was that the tutu in question was being worn by Ed Gould (190 cm tall, 140 kg wide, and hairier than me). This raised quite a laugh.

At the close of the conference two equal awards were presented, for the best presentation. One award went to Margo Seltzer, Keith Bostic, Kirk McKusick, and Carl Staelin for "An Implementation of a Log Structured File System for UNIX", despite the depressingly negative result. The second went to Stephen Uhler for "PhoneStation, Moving the Telephone onto the Virtual Desktop".

After the conference was over I went to dinner with some excessively rowdy people from Bell Labs, and then to a party. This was a hard week for me. No, really, I didn't leave the hotel grounds except for two dinners. I only got to the hot tub once...

7. Overview of Content.

Given three parallel tracks, and the fact that there was never a clear distinction in the type of the content, obviously it was impossible for me to see all the papers. I'll mention here those that I thought were worthy of specific comment. I will suggest to the newsletter editor that a full table of contents be printed, and you can purchase copies of the proceedings through AUUG.

I'll repeat, for the record, the observation that it was an extremely good technical program.

Pen based Computing and its Impact
Robert Carr

See discussion above.

Hello World (even Usenix proceedings shortened the title)
Rob Pike

Describes the use of a variant of Unicode as the base character set for the operating system, and some of the issues that arose during the implementation.

DUEL - a Very High Level Debugging Language
Michael Golan and David R Hansen

The authors present a small language which is oriented towards evaluating expressions about high level AUUGN

data structures. It is interfaced to a common debugger (dbx?) and finds its chief use for writing things like "show me the elements that are out of order in this array". I find the language interesting and useful, but the application inappropriate.

PhoneStation, Moving the Telephone onto the Virtual Desktop

Stephen Uhler

You had to be there, but reading the paper is also very rewarding, particularly if you have Sun SparcStations.

Jgraph - a Filter for Plotting Graphs in Postscript

James Plank

I only caught the last part of the talk, but this looked like useful stuff well implemented.

Wafe - An X Toolkit Based Frontend for Application Programs in Various Programming Languages

Gustaf Neumann & Stefan Nusser

The X application builder wars are being fought between Tk and Wafe. Stuff about Tk appeared a couple of Usenixes ago, and if you do any X application programming you need to know about both of these, although you will probably end up settling on one.

The Design and Implementation of the Inversion File System

Michael Olsen

Inversion is a file system, accessible through NFS, that is implemented within the PostGres database (hence the name), and has all sorts of interesting properties regarding uncorruptability, consistency and recovery, not to mention some interesting semantic properties (Query-language constructs in pathnames!).

File Systems in User Space

Paul Eggart and D Stott Parker

How (not) to hack useful but irregular and unpredictable semantics into your file system, by totally subverting shared libraries. Read it, the idea is very interesting, but the design...

The Organisation of Networks in Plan 9

Dave Presotto and Phil Winterbottom

Worth reading. Watch for things by Phil Winterbottom.

An Implementation of a Log Structured File System for UNIX

Margo Seltzer, Keith Bostic, Kirk McKusick, and Carl Staelin

As usual, a right scholarly piece of work that analyses the performance of the much touted Log Structured File System in the real world. Unfortunately, it seems perhaps to have been a dead end. There was a paper a year or so ago that used a different method to obtain faster write performance, that perhaps now need more examination.

The Nachos Instructional Operating System

Wayne A Christopher, Steven J Procter and Thomas E Anderson

I didn't see the presentation, but the paper won the best paper award.

There were lots of other papers that I hope I am not slighting by leaving them out of this list. There seemed to be something for everyone at this Usenix.

!AUUGN

The following article has been reprinted from AUUGN Volume 1, Number 1 (dated October 1978). It is interesting to see that the same sort of problems were faced by computer installations then and now.

UNIX in a Hostile Environment

or

Mouse Watching by a Cat

by

Peter Ivanov

As supervisor of the largest UNIX system in Australia, I read with some amusement the section on UNIX security in the July UK Newsletter and decided to share with you some reminiscences about "UNIX cracking" from my colleagues and myself. The incidents described in this account are NOT fictitious, although some may seem so.

Firstly, however, I must say that Ian and Mike from UKC really only touched the surface of the problem and unfortunately showed admirable restraint in NOT resorting to "inelegant expedients" which in my experience can make a system about as stable as a teacup in a typhoon.

Our system in Computer Science at UNSW (see equipment summary) currently supports more than 550 student users, a small proportion of whom would very inelegantly stab the system in the back given half a chance. Whether through malice, incompetance or chance all users are dangerous to varying extents and a system cannot be called "secure" unless it at least resists (if not defeats) all attempts to bring it undone! Thus security, in my book, encompasses a number of aspects, some of which are

- a) Protection against depletion of system resources (such as disc space, proc slots etc),
- b) Protection of individual users information (files) from corruption or observation by other users, and
- c) Protection of privileged or proprietary software from those users not granted access.

Obviously when a system is cracked in such a way as to give the "cracker" super-user status that is the end of all security but if any aspect mentioned above is cracked, the results could be just as serious.

Now to some story telling.

We obtained our first UNIX system (level V) in 1975 and the first few tales date from this period. From the very first days "pseudo login" programs appeared, NOT in order to steal names and passwords for our little 11/40 system I hasten to add, but to crack the Cyber-Kronos system with which we shared the terminals. Soon it was quicker to see a second year student to get more money put in your Cyber account than to see the computing centre. It is obviously very difficult to defeat a well written "login" program and about all one can do is try and break its grip on the terminal.

Soon the "computniks" tired of "Cyber cracking" and turned their attention to UNIX. A super-user accidentally left the source mounted "readable by others" for about 30 minutes. In this time user file space soared (copies of source in various disguises) and a bug was discovered in "login" where password length was not checked properly and enabled a password of specific length to be entered followed by its known encryption. It took two days to clean up all the set-uid-root shells and spare source AND ALL IN 30 MINUTES!!!!

Another old favourite usable with shells which search in the order "x, /bin/x, /usr/bin/x" was to leave a dummy command (eg "ls") in any writeable directory (even your own) and wait for a super-user to blunder past. A simple "chmod, chown, unlink, exec" sequence worked wonders. As with fishing most of the fun was in selecting the correct bait and tackle.

A variation of the above method works well on sloppily maintained systems where "writeable-by-others" commands are sometimes available for even the shortest periods of time. By over-writing the command with one which does "that little extra" you once more wait for a super-user to execute it for you. The shell problem is easily fixed by changing the search order for uid zero, but its variant is more difficult and will be discussed later.

Inevitably, holes in existing code are always popping up. Our local classic was the "lpr-lpd" combination. Our "/dev" directory used to be fairly rigidly protected and in order for lpr and lpd to access user files and "/dev" they were set-uid-root. All was well until people discovered the remove (-r) after printing flag on lpr. I leave you to contemplate what could happen and assure you that it did.

Looking over a super-users shoulder can give exhaustive encryption programs a very good head start. Early on we discovered that passwords should be at least 10 characters long and, if possible, totally meaningless. Fortunately, Australia abounds in 10-30 character aboriginal place names that few would dare to pronounce.

In late 1977 our prayers for a larger PDP for teaching purposes were answered and I was given the rare opportunity of supervising building modifications, cabling, installation, maintenance, software development and making the afternoon tea for the workers, all of which, believe it or not, affect security.

The reason why building modifications and installation are important was summed up beautifully by a salesman of "secure systems" who said

"This system is guaranteed secure as long as it is
not removed from the concrete box...."

Several people I know could, given access to the front panel, crack any machine on campus in less than 30 seconds. To lay hands on our 11/70 one must pass through four lockable doors, the last of which has a unique key and is always locked.

Terminal laboratories should be located nearby and be laid out in a systematic manner so that during brief, irregular and frequent visits, particularly out of normal hours, budding computniks may be identified by sight and login name. They may then be watched and, when they have progressed sufficiently, asked around for a cup of tea and given something useful to do in return for "certain favours". This way they get to further their skills and we get cheap programmers.

Local software developments have resulted in a system as secure as humanly possible. AUSAM, described elsewhere, has implemented resource limits (procs, disc space, page limits etc) so well that I can recall only once running out of disc space, caused by a bug in a super-user program.

Other software changes are:

- a) "Bug" programs to watch computniks and warn of their presence.
- b) Programs to scan file systems setting modes and owners, and reporting on "funny" files (those with names containing unprintable characters or starting with '.', or having set-uid-root modes).

- c) Alteration of "init" to fork a "login" instead of a super-user shell in single user mode.
- d) Alteration of "login" to cope with the "no password file" situation.
- e) Changes to a vast array of programs (work still in hand) to create files mode 600 or 700 so that users are protected by default. This is a partial solution to the shell variant mentioned earlier but unfortunately one must still depend on super-users never being clumsy.
- f) Fixing "sgtty" to disallow calls setting modes on a tty not owned by the user. This practice was being used to acquire terminals and access to other users accounts by setting incorrect baud rates or parity and forcing the unsuspecting victim to leave because he thought the terminal had stopped working.
- g) Modifying "passwd" so it asks for the new password without echo so that users passwords are not visible on a "ps".

Finally some random points:

- a) We only have one super-user, root, and refrain from using this login name on any terminals except those over which we have absolute control, in or near the 11/70 room.
- b) Our shell searches in the order "/etc/x, /bin/x, /usr/bin/x, x" for uid zero and placing all super-user needed commands in "/etc" actually makes ones life easier. Also placing "su" in "/etc" completely removes any worries about "using the wrong one" when super-user status is required since "/etc/su" must be used.
- c) Periodically I run off a complete "ls -ali" of the mounted system and take it home for some sunday morning reading, along with lists of all set-uid files and copies of "my computniks" latest creations.
- d) To combat "login" programs a "grep login" of the whole system will usually obtain the desired results unless unusual measures have been taken to disguise the programs presence.
- e) When confronted with a user who has obviously been acting the fool (eg sending billions of nulls to some poor bugger's terminal or stealing other user's login names) he should be immediately "excommunicated". That is all his files should be made inaccessible and his initial shell should be changed to give a curt message to "see the system supervisor" before exiting. Nothing hurts a computnik like no computing. When he comes grovelling simply tell him what he did wrong (not what he is accused of doing wrong, note), promise that if it happens again the removal will be final, and give him back his fun. Naturally watch him very closely for the next few months.

At this point I was going to say something trite about our system never having been cracked but alas I cannot. During a normal "ps -agl" last week, the first year computniks were discovered running a setuid shell made up to look like a "getty". They were sprung in the act and under the threat of excommunication revealed that we too are sloppy. They had found a writeable command thanks to a poorly written run file, had compiled a special version of the command and had waited for a super-user to execute it. They were rather peeved that they only had about 30 minutes to explore before being caught but swore that they had done nothing nasty in that time.

I suppose the only thing I can say in defence is that these gentlemen have recently cracked several other machines on campus, but we asked them around for that cup of tea months ago.

Tomorrow ...

Tomorrow, when you walk into your office, your company will acquire another business. Tomorrow a division of your organisation will be sold off to another company. Tomorrow the public sector department you work in will be merged with a different one, and some units moved to a completely separate department.

Tomorrow the Government will make a decision that changes the rules of the game in your industry, facilitating new entrants. A major overseas enterprise will start to compete with you. You will want to enter one or more new overseas markets.

A new business

Tomorrow the realities of your industry will force you to radically rethink the business you are in; you will be facing new competitors, new partners, new customers, new suppliers, new channels of distribution. The way your organisation adds value will change.

Tomorrow your competitors will start using information technology as part of the service they sell, not just to count the money. [Diag.1] A hot new technology will become available that you really would like to exploit in the process of getting your widgets to market. [Diag.2]

Tomorrow your customers, whether consumers or businesses, will desert the traditional product for a new way of meeting their need. They will no longer be satisfied with what they always bought before, and you and your competitors will be working hard to add new kinds of value to products and services. That value will often be directly derived from knowledge or information, or created through the application of information tools. The proportion of information-content in your products' total value will be higher. And the knowledge workers who create that value will be requiring ever more sophisticated tools.

No, we don't know precisely what tomorrow will be like. But we do know that it will be different. The old days of fairly static industries, predictable competitors, established value-chains are gone. Post-recession, those enterprises that want to remain relevant to their customers will have to keep pace with change.

End of traditional management structures

The management gurus are all telling us that changes like these are causing fundamental shifts in the way organisations work and are structured. Prescriptions vary, but some of the repeated themes are flexibility, networks, small teams, flat structures, integrated value-chains, greater inter-company links. Static, hierarchical models are breaking down. [Diag.3. Ref. 9]

† This paper was presented at AUUG'92

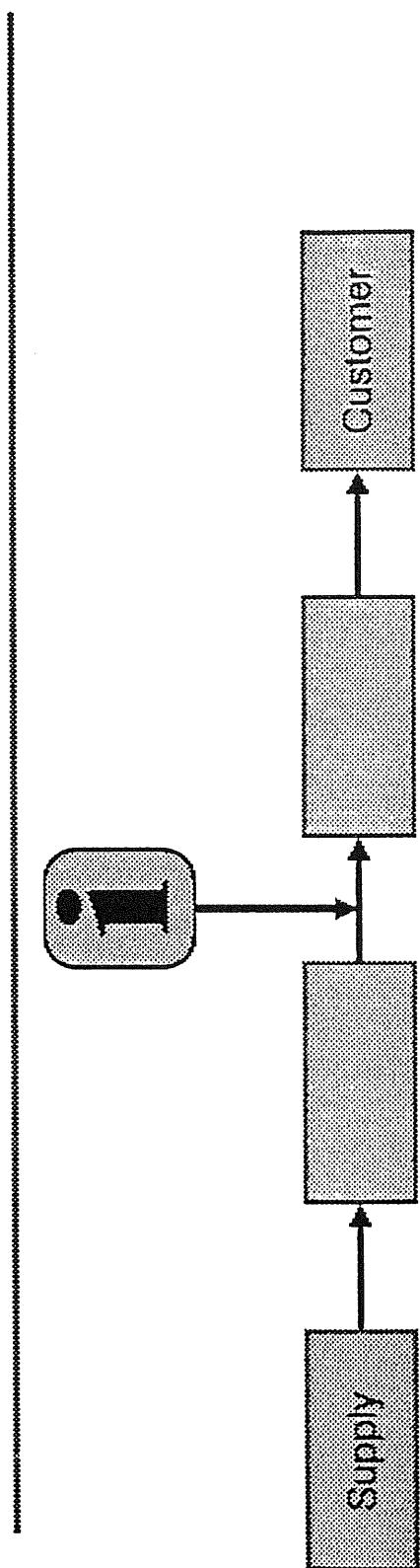
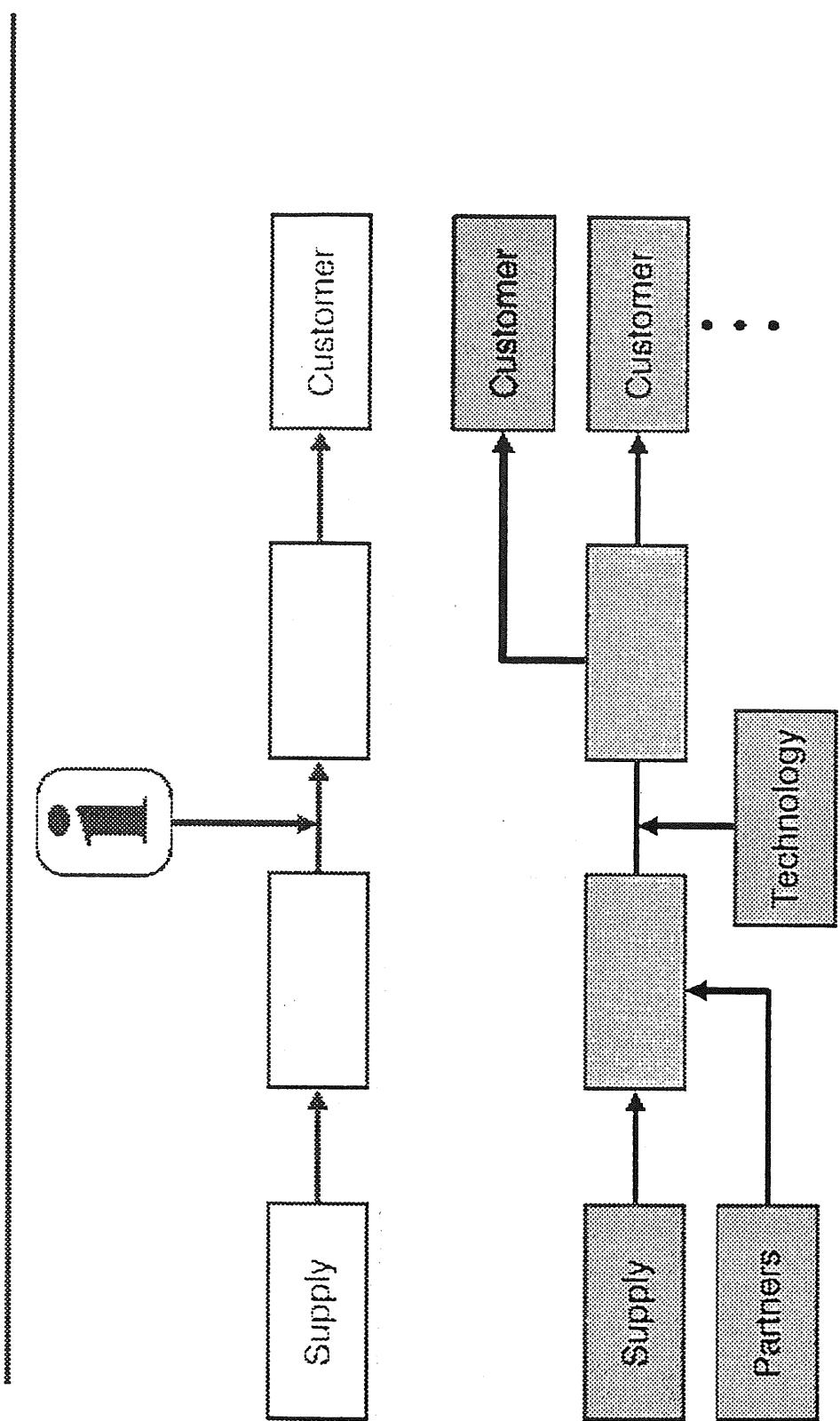
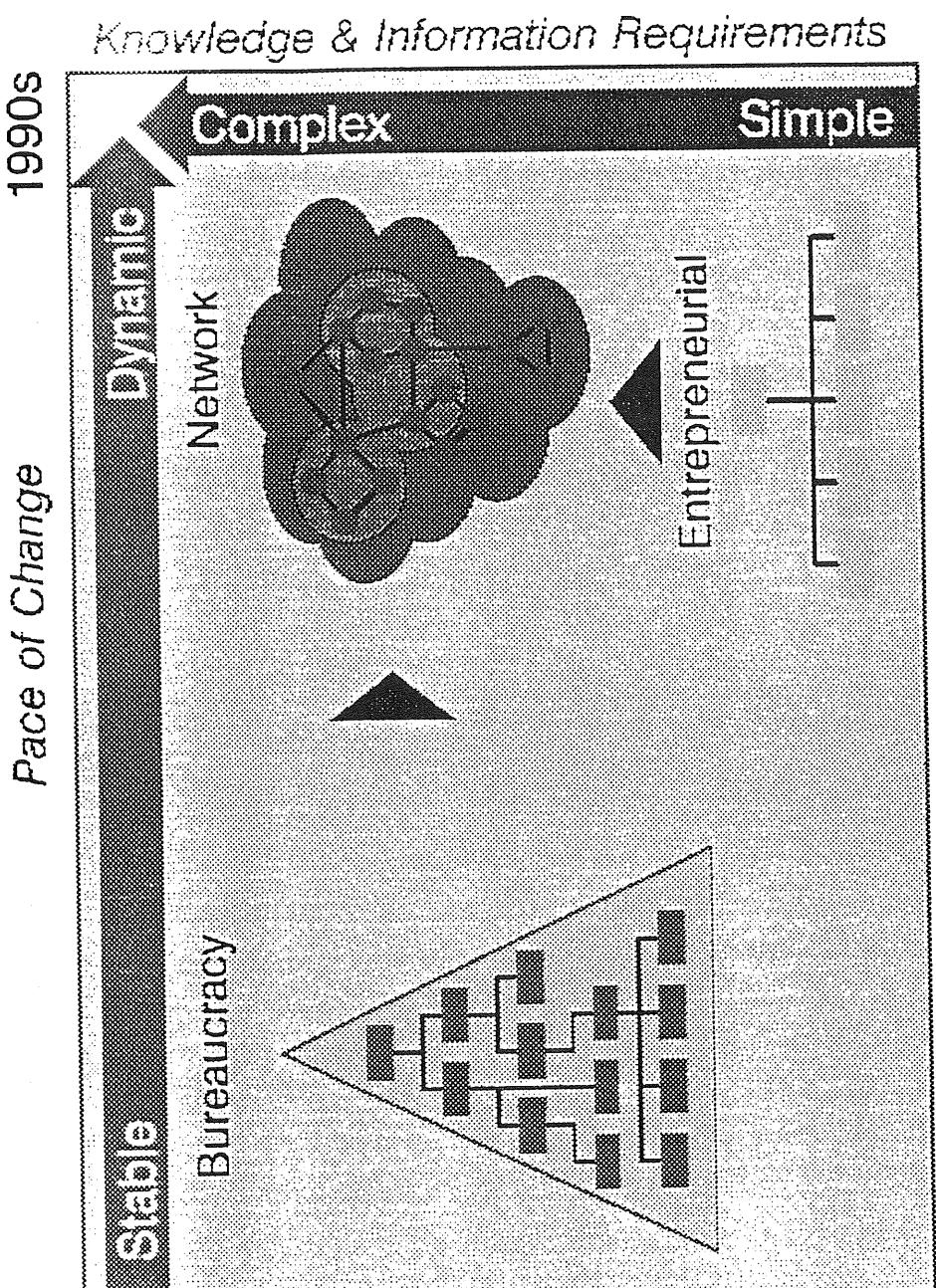


DIAGRAM 1



Organisational Forms and Environmental Demands



Source: Charles Savage - "Fifth Generation Management"

DIAGRAM 3

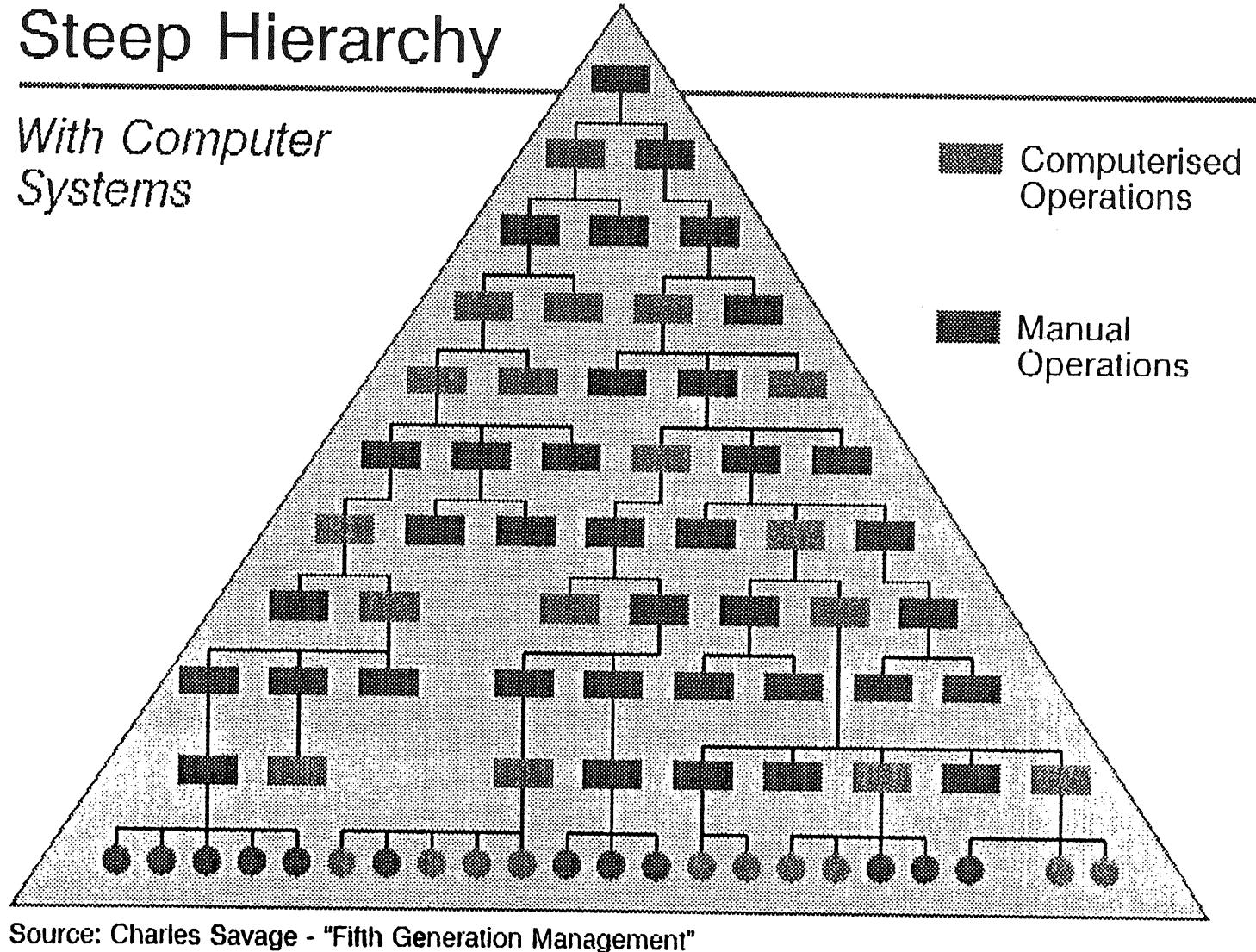


DIAGRAM 4

Information Technology Lifecycles

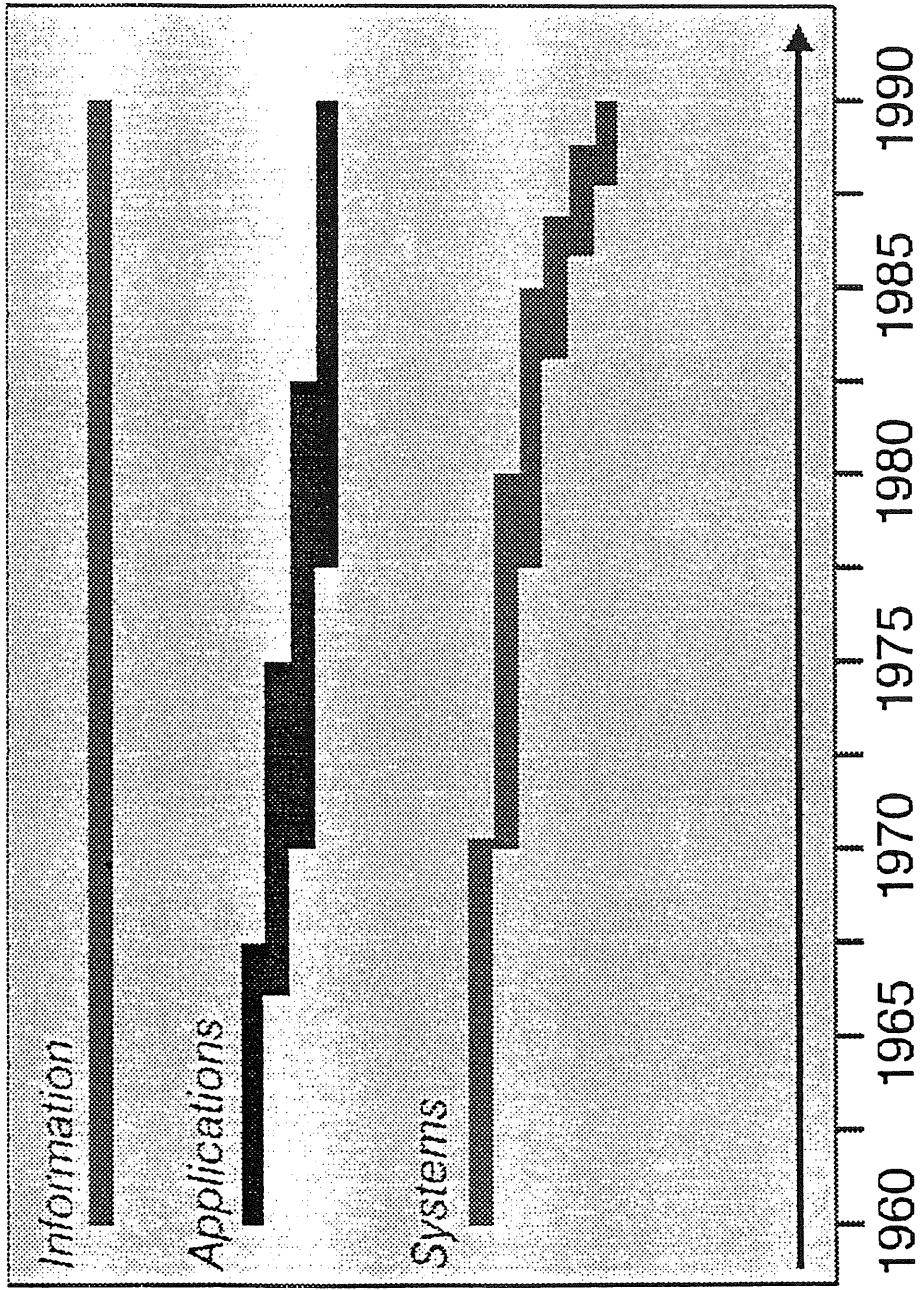


DIAGRAM 5

End of the simple information hierarchy

All these changes also have implications for information systems management. Static, hierarchical information systems are increasingly irrelevant to the new business environment. [Diag.4]

The information systems infrastructure of an organisation now has to cope with constant change. It has to accommodate the arrival of a new merged company with a different brand of computer. It has to be able to adapt quickly to links with new suppliers, customers, distributors, all of whom have different computer systems. It has to be able to serve a new organisation structure, and one that will keep on changing as the enterprise continually adapts itself to the evolving market.

The new market segments your company enters, or the new clients that your public sector department serves will bring with them new business needs that will create previously unanticipated information requirements. If you don't meet them quickly your competitors will.

The end-users of the information systems will want access to new information tools as their tasks change. They will want to be able to take advantage of the latest pieces of hardware for their desk-tops and the latest pieces of software. They won't want to be restricted to particular products or to vendors who may not have what they want.

The users of the information system will be moving around a lot more - from job to job within the organisation or outside, from one ad-hoc team to another, and physically moving around too. They will have to quickly learn to make effective use of the changing information tools that they find.

Information Technology Life Cycles

Meanwhile, the rate at which new technology is fed to the market accelerates to a pace that makes it hard for us to digest. [Diag.5]

Standardisation of products at every level means that radically new generations of products follow upon one another in a small number of years or even less than a year. We may well find compelling reasons to want to apply some of that technology - for cost saving or for meeting a new business opportunity. But our applications, and to an even greater extent, our data have a much longer life span. Those life spans derive from earlier times when assumptions about the structure of information models, fundamental applications and the nature of the business were static for many years.

Business impact of static systems

If a particular piece of hardware or software cannot fit into the structure, it may not just be an inconvenience any more. It may no longer mean just a cost saving missed. It may result in the company missing out on a major business opportunity.

There are two possible solutions.

The Soviet model

One answer would be for everyone to use the same brand of computer system, or the same brand of operating system, the same language or database. [Diag.6] That is the Soviet monopolistic model, the very antithesis of all the flexibility and change that we are seeing around us. Like the Soviet economic model, it must collapse.

Standards based flexibility

In fact there is only one answer. That is to move in a planned way to an information systems architecture in which each piece is built to a well-defined set of interface standards. [Diag.7] Put simply, the pieces fit because they all have standard plugs and sockets. Hardware, operating systems, languages, databases, network products, system management tools, human interfaces, and increasingly the application components themselves, have to conform to industry-wide standards.

I believe that users have no choice but to start building open systems. They will be creating long-term architectures based on standards, frameworks into which today's and tomorrow's as-yet uninvented technology will fit.

This is, of course, not the whole answer to solving all business and IT problems. But it is a necessary pre-requisite. The alternative is to face continued re-investment. The alternative is to accept the fact that the information system will always be a drag on the organisation's flexibility. More importantly, it would mean missing major business opportunities, because of information systems inertia.

Vendor-neutral

The standards that users adopt have to be independent of vendors, independent of consortia and particular trade-marked products. [Diag.8] Users have to be able to rely on the standards to be stable despite the competition-driven technological changes that will continue. Standards have to allow such technological change to continue, by virtue of being interface standards rather than specifications of what is inside the product. In fact, being external specifications, such standards will positively encourage competition and rapid product improvement.

It can be done now

The good news is that we are now at the point where users are able to begin building open systems. The knowledge of the open systems discipline is beginning to spread. System vendors have for some time now been supplying systems that conform to an ever-growing suite of universally agreed international standards, starting at the operating system interface. At every level of software, the body of standards is growing and user organisations are in fact now in a position to specify such suites of standards for their purchasing decisions. And despite the noisy debates in the media between competing vendors and consortia, all vendors do in fact conform to the same formal standards.

The Soviet Model of Information Systems

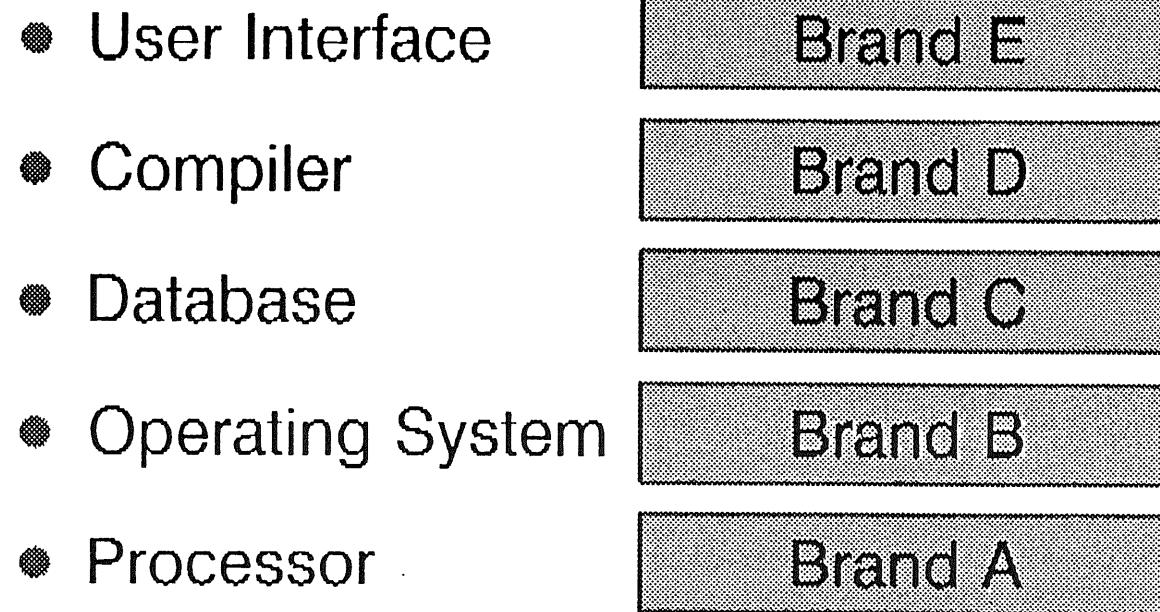


DIAGRAM 6

Standards-based Model

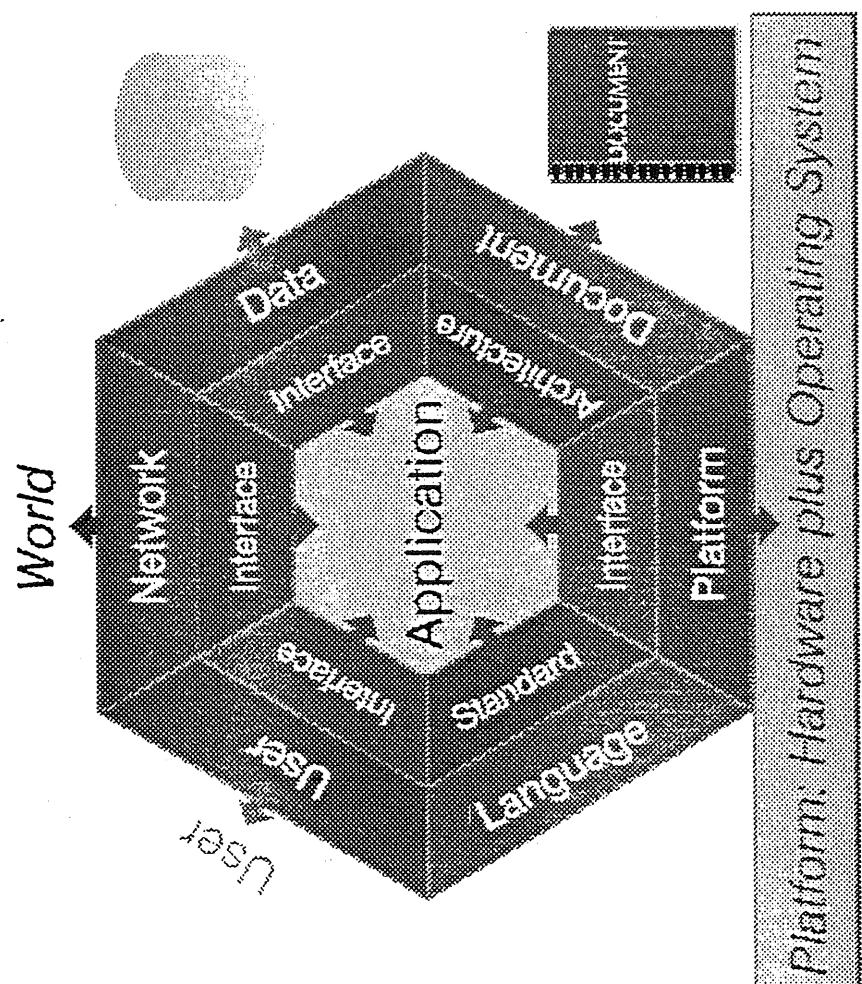


DIAGRAM 7

Interface Standards

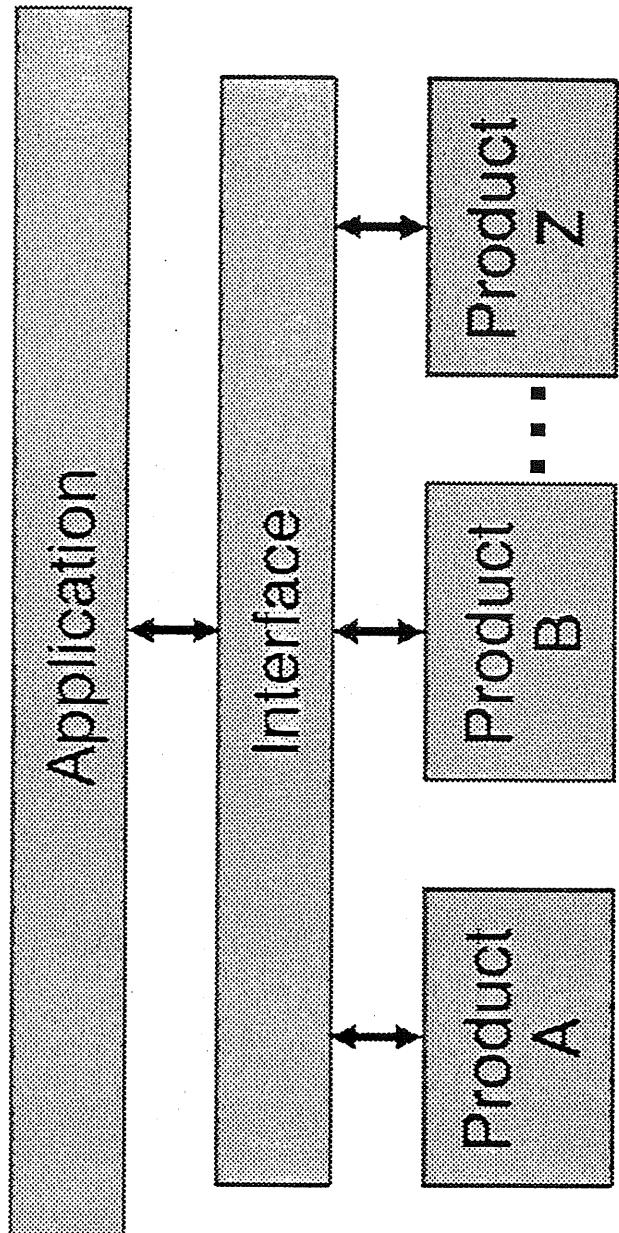


DIAGRAM 8

Organisations that are adopting standards

Some users are indeed beginning to adopt standards for the sort of reasons that I have outlined. The DMR Group of Information Systems consultants conducted a major study called "Strategies for Open Systems in Australia" [Ref. 3] a year or two ago, co-sponsored by AUUG. Amongst a wealth of other still highly valuable information about planning for open systems, DMR published data on the characteristics of those organisations that said they had a formal policy of adopting standards-based open systems.

The findings are described in detail in their report, but if I may put it in a simple way, the business driving forces found to be associated with open systems adoption were the ones that characterize the "clever" organizations, namely:

- shortening the delivery time of products / services,
- reducing the time to market for new products / services,
- expansion into overseas markets,
- product quality improvement or control,
- improvement in research/development capabilities.

There is a high degree of correlation between the extent to which organizations are innovative and open systems adopters.

These are the early adopters of standards. I suggest that all other will follow sooner or later.

Example: Unilever

Let's look at some examples. The international magazine Information Week, on March 2nd, 1992, carried a cover story under the heading, "How Unilever is gaining control of a global network of vastly disparate systems." [Ref. 10] Unilever is a giant international organisation, with 500 companies in 75 countries, employing 350,000 people and generating \$US 41 billion in annual revenue. They continue to acquire new companies, in various different areas of business, so they are not just big and complex, but are changing continually. My "tomorrow" scenario at the start wasn't imaginary. It is here and now.

Unilever of course have a large and complex multi-vendor array of IT assets. According to the article, they will continue to buy from multiple vendors. But they have now adopted a world-wide standards framework towards which all their information systems must evolve. This is no "big bang" approach to open systems adoption, but a conscious, rational policy that says that each new investment in a system must conform to their application environment specification, and all new applications, no matter what they run on now, must conform to standards and be portable. The Australian subsidiary of Unilever of course is well advanced in the move towards the standards-based approach, and has begun to deploy standards-compliant UNIX (tm) systems as part of its evolution.

The vision of this giant business is to be able to move confidently and quickly to exploit new business opportunities without technological restriction.

Example: Nippon Telegraph and Telecommunications Corporation (NTT)

NTT is the world's largest telecommunications provider. For NTT, in a very real sense, information systems not only support the business but are the business. An organisation like that has no choice but to position itself to exploit the best available technology from many vendors. Hence the development of NTT's Multi-vendor Integration Architecture (MIA). [Diag.9] MIA uses the principles we discussed earlier and allows for multiple products to co-exist through the use of rigorously defined interfaces. It specifies an Application Program Interface (API) between application programs and system software; a Systems Interconnection Interface (SII) for communications protocols; and a Human Interface (HUI) for display and workstation operation.

Example: U.S. Government

The largest IT spender in the world, the United States government, has also recognised the absolute necessity of building a standards framework for its IT purchases. Thus the National Institute for Standards and Technology (NIST) has published a number of mandatory Federal Information Processing Standards such as FIPS 151-1, dealing with a portable operating system. More importantly, NIST has produced a specification for an Open Systems Environment (OSE/1), to guide U.S. government agencies in their purchases and application development.

This open systems environment or Application Portability Profile (APP) specifies a consistent set of standards that government agencies should implement. At the operating system level, for example, it uses the IEEE POSIX 1003.1 interface standard, and the soon to be ratified 1003.2 shell and utilities standard. It recognises, however, that POSIX.1 and .2 are not enough. It therefore includes standards for security, user interface, programming languages, development tools, data management, data interchange, graphics, networking and management.

Wherever a formal standard or workable draft exists it adopts that. In areas where there is as yet no formal standard it uses a de facto standard or even an actual product as an interim solution.

Example: Australian Government

The Australian Government, through the Information Exchange Steering Committee (IESC), is currently developing a similar framework for all government information systems. It will be called the Commonwealth Open Systems Environment (COSE), and is modelled on the useful work done by NIST.

NTT Multi-vendor Integration Architecture (MIA)

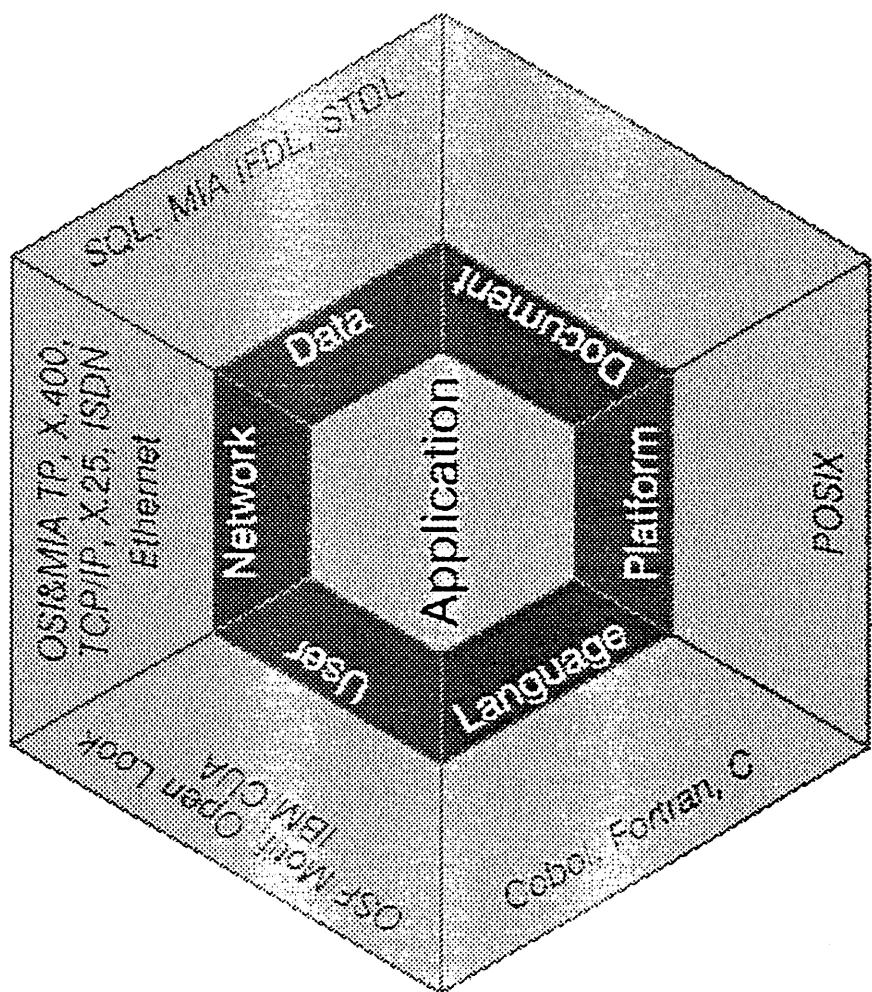


DIAGRAM 9

Example: Australian Government

Cont'd...

As a vendor representative, and as someone concerned about our industry, I welcome the leadership being shown by the Australian Government in this respect. Adoption of similar consistent standards framework by other Australian organisations, public or private, can only be to the good of our economy. If Australian firms wish to meet the needs of overseas customers, add value that customers will pay for, then adoption of international standards becomes an urgent necessity. Those customers, distributors and suppliers are part of a value chain that is rapidly becoming an information chain. Only consistent standards across whole industries will enable us to be a productive participant in the new economy. That is a major reason why the work of the Australian Government's IESC is so important.

Another positive recent development is the announcement earlier this year by the Queensland Government's Information Policy Board that it had adopted an open systems architecture. Future procurements will require evidence of conformance to the NIST profile and X/Open's Common Application Environment (CAE) as defined in the current X/Open Portability Guide (XPG). It is expected that X/Open branding will be used as evidence of conformance.

This move by a State Government is a healthy one since it adopts a practical standard agreed to by all parties regardless of which consortia they belong to. The XPG goes beyond the minimum standard of POSIX 1003.1 and 1003.2, in that it specifies a sufficient set of interfaces for a complete and workable system. XPG compliant applications are portable to all X/Open branded platforms.

Costs

Having looked at the over-riding reasons for implementing standards, let us now examine the costs and savings that result.

Contrary to conventional wisdom, building an open system may not be cheap. You are building an infrastructure that will allow your organisation to withstand all kinds of change, one that will allow you to profitably exploit the opportunities that such change presents. Building such an architecture requires knowledge, qualified professional staff, a disciplined approach and a long-term business focus. None of that comes cheaply, in dollars, people or time.

A significant part of the investment occurs during the very early stages of investigation, planning and design of the architecture. And the result is an intangible - a document that prescribes how the information system will evolve. It solves no short-term business problem and delivers no new application. But without that detailed strategy the organisation would never have an open standards-based system.

Costs Cont'd...

Adherence to the standards profile also increases the cost of the procurement process, at least in the earlier stages. Purchase decisions can no longer be made simply on short-term opportunistic grounds. The standards required have to be communicated to potential suppliers, and some level of verification and detailed investigation of compliance has to be undertaken if the standards commitment is indeed serious. That adds some delay and expensive skilled professional time. Of course it is worth it.

There will be a significant cost in training. IS staff have not only to learn some new products such as UNIX. What is much more difficult, they have to learn about the industry standards so that developers can stick to what is standard across all vendors.

All UNIX systems have extensions beyond the universal standard to support compatibility with historical versions. That is important for people who have applications developed on those historical UNIX versions. But a user who is adhering to standards must avoid getting painted into a corner again through the use of such features. After all, the whole point of adopting a standard at the operating system level is to avoid getting locked into one product.

And that is just the operating system level. The same thing has to be done at every other level of the information system architecture.

There will be costs in the promulgation, maintenance and enforcement of standards throughout your IS organisation. Most major IS users have established in-house standards and recognise the costs and benefits. But we are here talking about a much more comprehensive and more complex array of standards.

Since almost all major organisations have large investments in existing proprietary systems, there are definite costs in properly integrating newly acquired standards-compliant products with the established systems. The systems vendor should be able to help a lot here, but inevitably there is a requirement for integration products and services.

Not insignificant is the possible cost to the user of the new form of relationship with vendors. The move to a standards-based open systems architecture implies a shift in the user-vendor relationship, a shift in the balance of power towards the user, but also a shift in responsibility to the user. While in earlier generations one could simply adopt a vendor's product-sets and architectures in total, the onus for developing the architecture now at least partly becomes the responsibility of the user. Suppliers and consultants can help, but they have to be paid for.

Risks?

Some people have expressed the concern that adopting a standards approach might limit a user organisation, preventing it from adopting a proprietary solution that could offer a competitive advantage. I do not believe that there is a serious risk of this happening in practice. Firstly, the market for standards-compliant products is so large and dynamic that it is reasonable to expect most if not all useful innovations to be readily available within a framework of international standards. Secondly, no standards framework needs to preclude any user from consciously adopting individual solutions that go outside that framework in carefully defined ways. There is a future maintenance and re-investment cost in taking that path, but a careful consideration of benefits might sometimes indicate that it is valid. All Systems Development Managers would be familiar with this issue in the form of going beyond good standard programming practice for a possible performance gain. It is a matter for commercial risk judgement and management backed by technical advice.

Cost Savings

Let's now get to the cost savings. You will have gathered by now that I think there is much more to it than cost savings. I agree with Dr Pamela Gray [Ref. 4], who in her book "Open Systems: a business strategy for the 1990s" draws the conclusion that "an information system strategy based on 'open systems' is the only logical way forward in the 1990s for most organisations."

But there are indeed cost savings in a well managed standards-compliant information system. [Diag.10]

Hardware savings

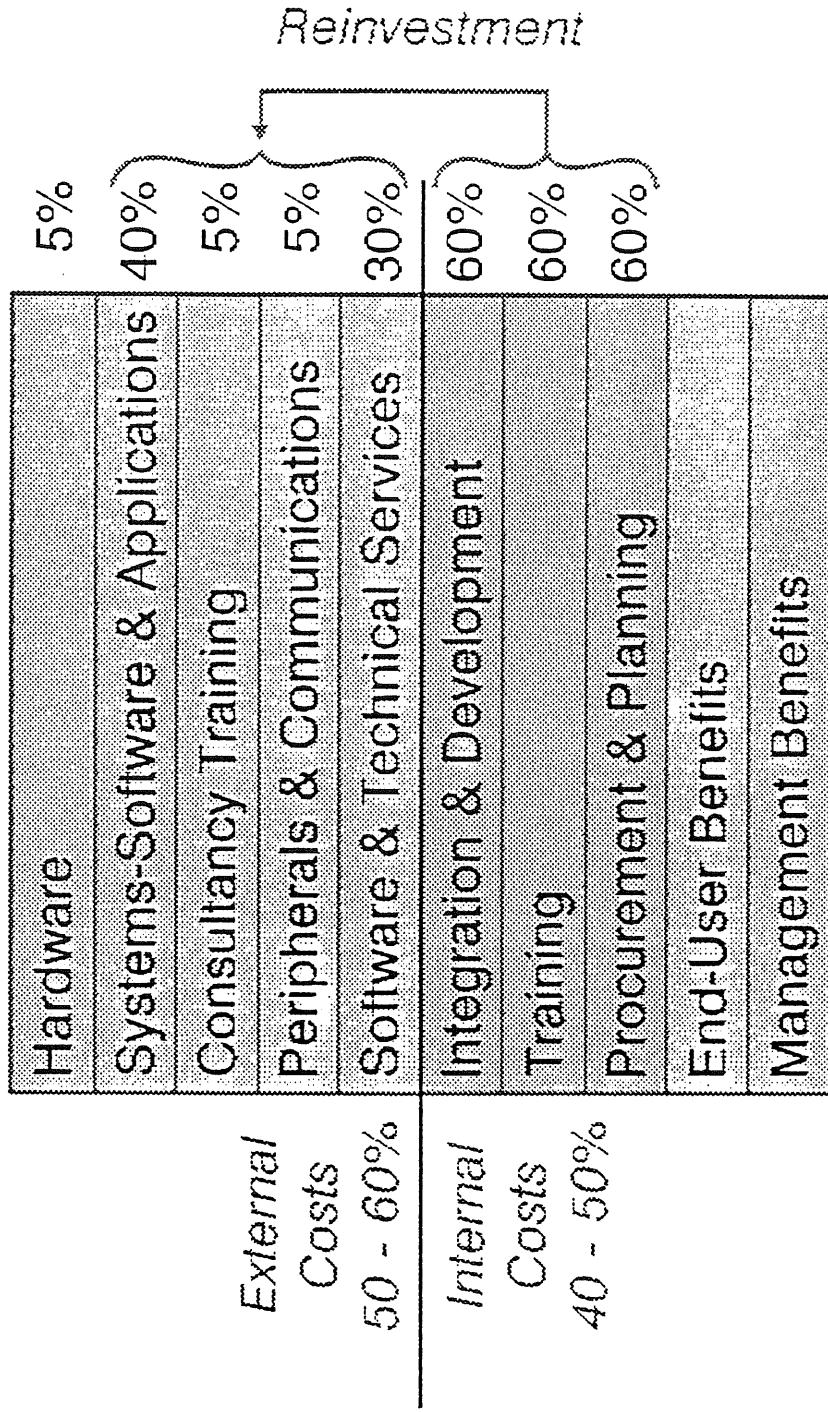
Hardware is the area that first springs to our minds. No matter how well we all know that it represents but a fraction of IS costs, we still all, myself included, tend to look to the tangible "iron" as the place to save money. In fact there can be savings here. Since all UNIX systems conform to the relevant standards, we can oversimplify for a moment and compare UNIX systems with non-UNIX systems. According to the 1992 Mid-Range Systems Report published by Ideas International [Ref. 5], non-UNIX systems tend on average to cost 48% more over 5 years life, counting the hardware, system software and the associated support charges.

We have to be wary of simple averages like that, however. Some UNIX systems are more expensive than some non-UNIX systems, and the most expensive non-UNIX systems of all are in my opinion those that are destined not to survive very long in any case. Nevertheless, it seems to be clear that we can save some money on the systems platforms.

In theory, standards-compliant products, whether hardware or software are more expensive to engineer since more discipline and testing has to go into them. But the use of common components, including common software components lead to cost savings for the manufacturer. The market volume and the competitive environment lead to reduced prices.

Cost Impact of Standards

IT Budget *(Cost Saving)*



Source: Wartworth Management Consultants

DIAGRAM 10

Application savings

Today all UNIX systems conform to the international industry operating system standards. As soon as a majority of application providers take advantage of that fact, then they and users will benefit from greatly reduced porting and maintenance costs. Today, many applications in the UNIX world have been written for one or another of the historical UNIX flavours and then ported to others. Software houses are now realising that users are demanding standards-compliant applications. After all, what's the use of a standard power socket if all your appliances have non-standard plugs?

Of course there is a cost for the software vendor to re-engineer the application to conform strictly to POSIX, XPG, ANSI languages, SQL etc. A disciplined approach is never cheap. But the benefits are enormous. Porting new versions can become a largely automated process. A common source pool means reduced maintenance and support costs. Meanwhile, a huge market of standards-compliant platforms open up. As with hardware, the cost savings, the volumes and the competition will drive prices down and quality up.

Savings through new technology

A thorough standards policy is based on a layered model, where each layer of technology is isolated from others through well-defined interfaces. A direct consequence is that it becomes possible to adopt new components quite readily without upsetting the rest of the system. The hardware layer might change, or the operating system kernel, leaving the majority of the investment intact. Thus new technology can be brought in when a cost saving, performance or other advantage can be gained that way.

Savings in procurement

Although I said earlier that there is a start-up cost associated with the transition to standards, once the standards are well understood in the user organisation, the procurement process can accrue savings as well. The common base-line functions inherent in the industry standards can be defined and evaluated quickly and reliably against a known yardstick. We waste less time on comparing common functions in products. More time can be spent on comparing features that are outside of the scope of standards, and on ensuring that the best value for money is obtained. In the case of public sector agencies, who have to be vendor-independent and seen to be so, this will be particularly significant. Tender evaluations in a non-standard environment involve an expensive process of digesting information about and evaluating totally disparate systems. And we haven't even mentioned the outrageous costs of periodically converting between different non-standard systems.

Application Development and Maintenance

Once a user organisation has established its standards profile, trained its people and become experienced in the effective use of standards, then there will be clear savings in in-house application development. It will take time to get there, and in the interim the cost may well go up due to the learning process. Eventually, however, the advantage of re-usable software and relatively easy global deployment on multiple systems will outweigh the initial difficulty.

Application maintenance is even more important, often estimated at costing 80% of the total systems development budget. A large proportion of that tends to be spent on migration to new hardware or new operating systems. If a truly rigorous standards policy is enforced, that can be sharply reduced. But I emphasise, the standards policy must be rigorous and enforced.

The key to this is that applications must conform to standards, not just the platforms. It sounds obvious, but in fact users have had problems with portability. They have bought standards-compliant UNIX systems. The applications they have bought or developed, however, were developed using some of the proprietary extensions in those UNIX systems. Now there is nothing sinister in those extensions. They are necessary to support continuity with the historical UNIX versions. Vendors have to provide that compatibility so that customers can continue to run their old applications. But if developers continue to use facilities that are not in POSIX and XPG, then they are locking themselves in to specific products. The cost savings from portability, therefore, are dependent on a thorough application of the strict standard.

Training and services

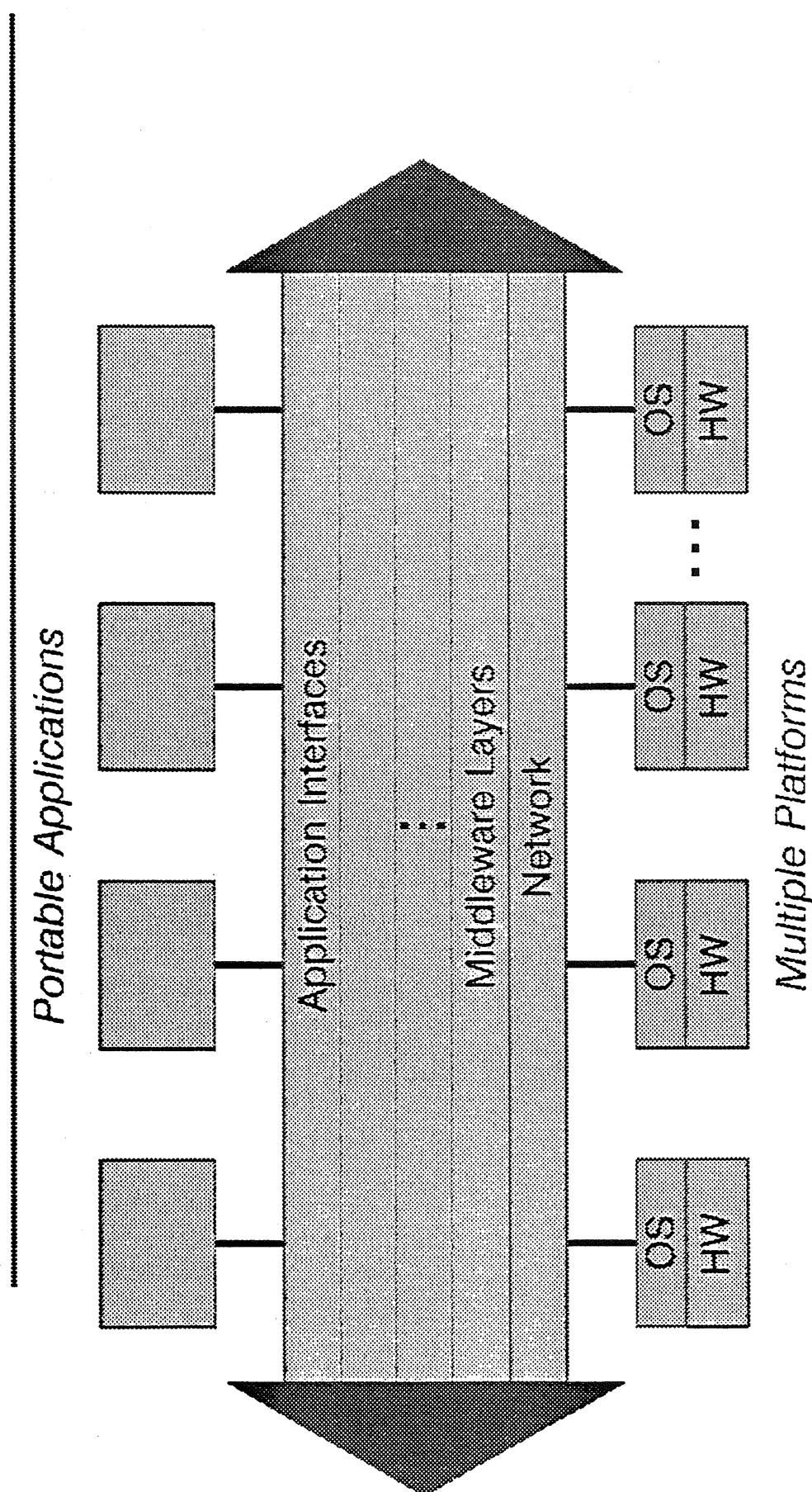
In the longer term, significant savings will be realised in the area of training and outside services. This is because you will be drawing on a pool of practitioners all trained and experienced in a standard environment, both for your potential employees and contractors.

Risk Reduction

Open standards offer the greatest risk reduction for both users and suppliers. [Diag.11] Independent bodies like ISO maintain the standards and control their evolution over time. Where conflicts exist regarding desired changes in a standard, resolution is arrived at through an open, independent consensus process. Both parties can thus plan their investments confidently.

The benefits

The cost savings we have examined are obviously going to be significant, after a period of investment. But I firmly believe that the more important benefits are well beyond the cost issue.



The Benefits Cont'd...

Let me draw an analogy to the Australian economy. It is certainly true, as businessmen and politicians remind us, that firms in Australia have to operate cost-effectively to compete. But there is only so much you can achieve with cost-cutting. You can keep cutting costs and you can still go out of business. What Australian enterprises have to do is create value that people whether here or overseas, want to pay for. That means being in touch with the fast changing needs of those customers, with the hectic pace of global change, competition and technology. It means being better at meeting those needs.

So we are back with my opening scenario. The economy is no longer static. Information systems, if they are to be part of the solution rather than part of the problem must anticipate change and enable us to exploit the opportunity that it brings. There is only one way to do that - standards.

References and Further Reading

- [1] Australian Standard 3976-1:1991. Sydney: Standards Australia, 1991.
- [2] Commonwealth Government, "Open Systems Newsletter", quarterly. Write to editor, Open Systems Newsletter, Department of Finance, room A329, Treasury Building, Parkes ACT 2600.
- [3] DMR Group, "Strategies for Open Systems In Australia" (4 vols.). Sydney: DMR Group, 1991.
- [4] Pamela Gray, "Open Systems: a business strategy for the 1990s". London: McGraw-Hill, 1991.
- [5] Ideas International, Mid-Range Systems Report. Sydney: Ideas International, 1992.
- [6] Jim Isaak, "Standards: a healthy investment", a White Paper. Maynard MA: Digital Equipment Corporation, 1992.
- [7] National Institute of Standards and Technology, publication 500-187, Application Portability Profile, Open Systems Environment profile (OSE/1).
- [8] The Multivendor Integration Architecture: Overview. Tokyo: Nippon Telephone and Telegraph Corporation, 1991.
- [9] Charles Savage, "Fifth Generation Management". Maynard MA: Digital Press, 1990.
- [10] "Uniting the World: how giant Unilever is gaining control of a global network of vastly disparate systems." InformationWeek, March 2, 1992.
- [11] X/Open, "X/Open Portability Guide", 3rd edition. Reading: X/Open Company Ltd.

Trademarks

UNIX is a trademark of UNIX System Laboratories.

Graphic User Interfaces Made Easy? †

A Tcl/Tk Tutorial

Robert Biddle
Computer Science
Victoria University of Wellington
`robert.biddle@comp.vuw.ac.nz`

Tcl is a small programming language designed to be embedded into other programs to enable extensibility and customisation. Tk is an X window system toolkit providing graphic user interface facilities in connection with Tcl. Both were developed by John Ousterhout at the University of California at Berkeley, and are available free. Tcl and Tk together offer an easy route to graphic user interface programming in the X environment. This document is a practical introduction to using Tk and Tcl, concentrating on how they can be used to quickly develop flexible graphic user interface facilities. Some simple examples are worked through, and suggestions made as to how to proceed further.

1 Introduction

Tcl (Tool Command language) is small easy-to-use language designed to be embedded into other systems to enable simple yet flexible extensibility and customisation. Tk is an X window system toolkit providing simple graphic user interface and event driven facilities in connection with Tcl. Both Tcl and Tk were developed by John Ousterhout at the University of California at Berkeley, and, like the X window system, are available free for a wide range of Unix based systems.

This tutorial is a practical introduction to Tk and Tcl, and will focus on how they can be used together to very quickly develop flexible graphic user interface facilities. This means the main concern will be Tk, and Tcl will be regarded as a supporting framework. Tk and Tcl are easy to learn, but no practical introduction exists: this document is an attempt to fill that gap. It will be useful to read through this document once, and then to go through the examples again and explore further at an X display. Full details about the language and the toolkit may be found in several documents described at the end of this tutorial.

Tcl is a simple programming language oriented mostly to character string processing. It has one type only, the string, and one data structure only, the array — arrays may be indexed by arbitrary strings. The control structures are conventional, `if`, `while`, and so on, and a simple procedure is also available with value parameters, local variables, and a return value. There is a range of built-in procedures that allow easy manipulation of strings, and lists of strings, including pattern matching and substitution. There is also a range of procedures to facilitate interaction with input/output and the external system in general.

The Tcl language is implemented by an interpreter, so that instructions in the language are acted upon as they are recognised. The interpreter is accessed by a subroutine call, and instructions are passed as parameters and results of the interpretation are then returned. In this way Tcl capability may easily be added to another program, allowing access to the programmable flexibility of Tcl. Moreover, associated facilities are also available by subroutine call that allow *new* procedures to be added to the language recognised by the interpreter. These new Tcl procedures are implemented by procedures specified by the host program. In this way, a host program may be extended by Tcl capabilities, and Tcl capabilities may themselves be extended and customised by the host program.

† This paper was presented at UniForum NZ '92

The Tk toolkit is a set of procedures that extend Tcl in this way with facilities for graphics and event-driven interaction using the X window system.

While Tcl was designed to be embedded in other application programs, it is itself sufficient for many string based programming applications. Moreover, with the Tk procedures, it is itself sufficient for many straightforward graphic user interface applications. For such applications, a very simple program — *Wish* — is available that takes lines of input and passes them to a Tcl interpreter. The Wish program may be used interactively, and so is an excellent way to learn about Tcl and Tk; Wish may also be used to interpret script files of Tcl and Tk statements.

2 Getting Started

In the classic book on the C programming language, Brian Kernighan claims “the first program to write is the same for all languages: print the words *hello, world*”. In Tcl, the program is:

```
puts stdout "hello, world"
```

To run this program, you can start Wish, and at its prompt (*wish:*) simply type the line above, and hit return. Alternatively, you could create a file with that line, *hello.tcl*, and then have Wish use that file as a script:

```
% wish -f hello.tcl
```

Either way, the words *hello, world* should appear on the standard output.

In Wish, the graphics extensions of Tk are integrated into the Tcl language: they appear simply as extra procedures that can be evaluated in Tcl — with the graphics and event interaction happening separately. When you start Wish interactively at an X display, the presence of the Tk extensions is shown by a square graphics frame appearing as a separate window. This frame is flexible, and will be controlled as a result of Tk procedures read by Wish, and evaluated by the Tcl interpreter. To explore this, try the following Tcl/Tk program:

```
# hello.tcl -- Hello, World in Tcl with Tk Widgets
label .title -text "Tcl/Tk Greeting Program" -relief raised
label .response
button .greet -text "Greet" -command {.response configure -text "hello, world"}
pack append . .title top
pack append . .response top
pack append . .greet top
```

This program creates three graphics “widgets” within the frame: two “labels” and a “button”. Labels simply display some text; buttons also display text, but have a “command” associated with them that is executed if mouse button 1 is pressed within the widget. Pressing the “Greet” button should result in the “hello, world” message appearing in the middle of the frame.

The **label** and **button** procedures create widgets with the name given; configuration settings may follow the name. Complex widgets may be constructed that contain other widgets within them, and for this reason a widget must be specified indicating where in the hierarchy it belongs with a *pathname*. The pathname **.greet** indicates the button is in the



Figure 1: The display from `hello.tcl` after the “Greet” button has been pressed.

original “root” frame (.); if there had been an intermediate frame with pathname `.dialog`, the button within would have pathname `.dialog.greet` showing the relationship.

The `pack` procedure is a “geometry manager”: in accordance with certain guidelines, it arranges for widgets to be mapped together. This arrangement is flexible, and will change automatically with the packing and unpacking of widgets, as well as with any significant changes in widget configuration settings. The `append` directive indicates that a parent and child widget follow, and that the child widget should be displayed within the parent. Widgets are arranged with respect to a flexible “cavity”, and the `top` option indicates that the new widget is to be displayed at the top of the cavity.

The state of a widget may be inspected or changed by calling a procedure with the same name as the widget itself. The first argument states the operation required, and the remaining arguments are used for the operation. Each “class” of widget has particular operations, but all have the `configure` operation, which takes the remaining arguments as configuration settings: the `button` action command above sets the text displayed by `.response`, for example. After typing the above program in interactively, the following could be typed:

```
.response configure -text "how do you do?"
```

This will immediately change the text showing in the middle widget; of course, pressing the “Greet” button will change it back. If the `configure` operation is not followed by any arguments, a list of all the current configuration settings is returned. Exactly what is included in a widget configuration depends on the class of widget, but most include things like size, colour, border width, relief effects, and so on.

As mentioned earlier, the `button` widget class includes a configuration setting `-command` that specifies what Tcl command to execute if the mouse button 1 is pressed within the widget. Such event handling is available with more flexibility via the `bind` procedure. For example, it is also possible to associate a command with mouse button 2 being pressed:

```
bind .greet <ButtonPress-2> {.response configure -text "bye for now"}
```

Now pressing mouse button 2 within the button widget will result in the text `bye for now` being displayed; and pressing mouse button 1 will result in `hello, world` being displayed again. For any widget, the `bind` procedure is capable of binding any Tcl command with a great variety of events.

The major Tk procedures have now all been introduced: `widget create` (each procedure being the name of a widget class, like `label`); the display management procedure `pack`; `widget specific procedures` (each being the name of a widget itself, like `.response`) with specific operations (like `configure`); and the event handling procedure `bind`, which associates any event involving a widget with some Tcl code to be executed.

These few Tk procedures, the availability of a useful set of widget classes, and the flexibility of Tcl programming, are sufficient to easily create a wide variety of programs with useful visual and event driven interfaces. The next steps in learning how to create such programs involve learning more about particular widget classes, a bit more about the Tcl language, more about interacting with Tcl and Tk programs, and more about the important display management procedure pack.

3 The Tk Widget Set

Because all the Tk facilities are extensions to Tcl, and because Tcl is so easily extended, there is little reason to regard any particular set of widgets as "intrinsic". However, the distribution of Tcl and Tk includes a very useful set of widgets, and it seems reasonable to assume that they, at least, will be available in any Tcl/Tk programming situation. These widgets are described briefly in the following table:

Widget Type	Brief Description
<code>label</code>	displays a flexibly sized single line of configurable <code>text</code>
<code>message</code>	displays a block of configurable <code>text</code> ; flexibly sized to display the entire text in a rectangular block with configurable <code>aspect</code> ratio
<code>button</code>	displays a flexibly sized single line of configurable <code>text</code> ; configurable <code>command</code> is executed on press of mouse button 1
<code>checkbutton</code>	displays a small square-shaped box; operations to change and inspect state and appearance of the box from empty to colour filled; configurable <code>command</code> is executed on press of mouse button 1
<code>radiobutton</code>	displays a small diamond-shaped box; operations to change and inspect state and appearance of the box from empty to colour filled; configurable <code>variable</code> is associated with button state, so states of radiobuttons are arranged mutually exclusive; configurable <code>command</code> is executed on press of mouse button 1
<code>scale</code>	displays a bar of configurable size with a slider that may be dragged along the bar with the mouse button pressed, with a numeric value displayed alongside, changing with slider motion; the numeric value may be changed with the <code>set</code> operation, and inspected with the <code>get</code> operation
<code>entry</code>	displays a box of configurable <code>length</code> ; various operations allow text to be inserted, deleted, inspected, scrolled, and selected
<code>menu</code>	displays a vertical menu of items displaying lines of text; various operations to insert, delete, and select items, and to associate settings and commands with item selection
<code>menubutton</code>	displays a flexible single line of configurable <code>text</code> ; configurable associated <code>menu</code> is displayed ready for item selection on press of mouse button 1
<code>listbox</code>	displays a box of configurable size, with a vertical sequence of associated lines of text; operations to insert, delete, and select lines, and to show various sequences of the lines within the box
<code>scrollbar</code>	displays a bar of configurable size with arrows and slider; various operations allow position of slider to be changed and inspected; useful in controlling <code>listbox</code> widgets

Widget Type	Brief Description
frame	holds other widgets within, allowing them to be treated as a unit for display management
toplevel	holds other widgets within, and is displayed as a separately managed window; useful for temporary dialogs

4 The Tcl Language

This tutorial is concerned mostly with Tk, and is regarding Tcl as the framework for Tk. The examples presented here show some of the features of the Tcl language, but by no means all of them. There are different variations on the commands shown, and many commands are not introduced at all. But most of the language features are simply useful procedures, and do not require learning special syntax or difficult concepts.

The list and string orientation of the Tcl language is reflected in the language itself: statements and expressions are in fact simply lists of strings. Variable names may be evaluated by prefixing the name with a dollar-sign (e.g. \$inputline). Strings separated by spaces may be regarded as a whole by surrounding them by double quotes ("one string"). Lists of strings may be evaluated, with the first string used as a procedure name and the rest as arguments, by enclosing the list with square brackets (e.g. [refresh table \$inputline]). This evaluation is done automatically if a list appears as a complete statement, either on a line alone, or separated by a semicolon (;). So in fact, all Tcl "commands" are really procedures: even the proc command is a procedure call to establish another procedure, with the argument list and body being strings.

While evaluation of variables and procedure calls as arguments must be explicit, the evaluation should not be done too soon. Evaluation of variables and procedure calls may be deferred by enclosing strings within braces; this illustrates the strong string nature of Tcl, as it is the method used to nest control structure:

```
if $fetch then { puts stdout [lookup $key] } else { puts stdout $line }
```

To a C programmer, the style looks reasonably familiar, but the braces are there to prevent premature evaluation of the arguments to if.

5 Interacting with Tcl and Tk Programs

Some classes of Tk widgets, the buttons, have a configurable command setting that associates a Tcl command to be executed when some event concerning the widget occurs, like the mouse button being pressed within the widget.

However, many widgets do not have an association between an event and an action that may be so simply specified. Moreover, even buttons may sometimes need something more sophisticated. Using the bind command with buttons to associate different commands with different mouse buttons has been shown earlier, for example.

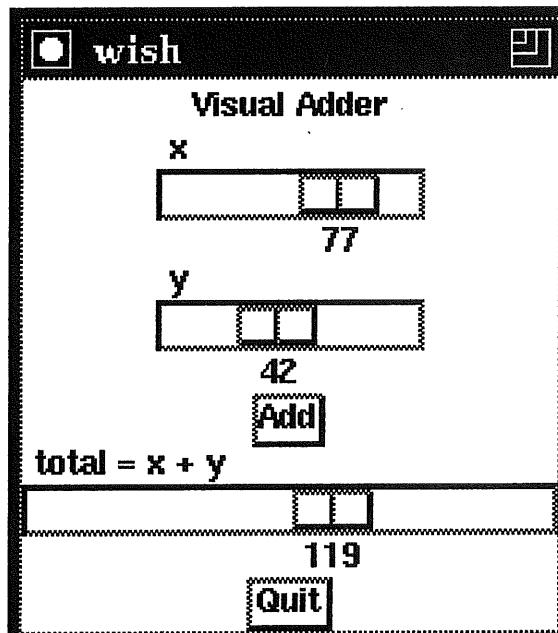


Figure 2: The display from `adder.tcl` showing the value of the bottom scale to be the sum of the values of the two top scales. This can be deferred until the button is pressed, or continuous with motion of either of the top two scales.

Consider the following program, where the settings from scales are only used when a button is pressed:

```
# adder.tcl -- Program that Adds Two Numbers Visually
label .title -text "Visual Adder"
scale .x -to 100 -length 100 -label "x" -orient horizontal
scale .y -to 100 -length 100 -label "y" -orient horizontal
scale .total -to 200 -length 200 -label "total = x + y" -orient horizontal
button .add -text "Add" -command settotal
button .quit -text "Quit" -command exit

proc settotal {} {
    .total set [expr "[.x get] + [.y get]"]
}

pack append . .title top
pack append . .x top; pack append . .y top
pack append . .add top
pack append . .total top
pack append . .quit top
```

In this program, the user may set the values of two numbers using the two top scales. When the “Add” button is pressed, the setting of the bottom scale is set to the sum of the two above.

Now consider the same program, but with the addition of the following two statements:

```
bind .x <Button1-Motion> settotal
bind .y <Button1-Motion> settotal
```

Here any motion of cursor while the mouse button is depressed on the scale slider causes the `settotal` procedure to be called. The display is the same, but now the bottom scale shows the sum of the upper scales — *as they are themselves being set*. This means that with Tk, dynamic input can be reflected dynamically: a powerful method of visual feedback. The flexibility of Tk here is extraordinary — it is even possible to use a dynamic action to cause continuous re-configuration of the widget itself involved in the action. Consider this sequence:

```
# self configuring widget...
scale .self
pack append . .self top
bind .self <B1-Motion> {.self configure -width [.self get]}
```

The `bind` command also recognises a wide variety of other events: button press for each mouse button, possibly modified by other keys or repeated, button release, keyboard entry, and cursor motion of various kinds.

The `entry` widget class requires careful attention in keyboard event handling. Not only must characters be inserted as typed, but simple line editing facilities should be catered for. The following program needs to allow a file name to be typed into an `entry` widget:

```
# checkread.tcl -- Check If a File is Readable
label .title -text "Check File Readability"
entry .file -relief sunken
button .checknow -text "Check" -command docheck
label .report
button .quit -text "Quit" -command exit

bind .file <Any-KeyPress> { .file insert cursor %A}
bind .file <Delete>      { set p [expr {[$.file index cursor] - 1}]
                           if {$p >= 0} {.file delete $p} }

proc docheck {} {
    if [file readable [.file get]] then {
        .report configure -text "File IS Readable"
    } else {
        .report configure -text "File IS NOT Readable"
    }
}
pack append . .title top
pack append . .file top
pack append . .checknow top
pack append . .report top
pack append . .quit top
```

This program allows a filename to be entered, and when the button is pressed it checks to see if the file is readable. The first `bind` directive concerns all ordinary characters; the second concerns the delete key. The `bind` procedure provides special values preceded by a percent sign (%) that hold various values specific to each event. When an ordinary character key is pressed, that character (%A) is inserted into the widget text string, before the current cursor using the `insert`. When the delete key is pressed, the current position is checked

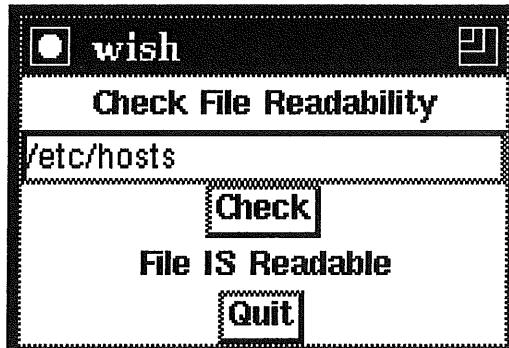


Figure 3: The display from `checkread.tcl` showing the result of pressing the button to check whether a file is readable.

decremented, and if that remains a valid position, the character at the current position is deleted via the `delete` operation. In fact, there are many other events one might wish bound to a text entry widget, from re-positioning with the mouse, to a full cut and paste capability. The best way to deal with complicated lists of bindings you frequently need is to write a procedure that sets up the bindings for any widget you pass as an argument.

The program above also shows Tcl interacting with the Unix system. The `file` procedure provides various information about files, depending on its argument — in this case `readable`. The `open` procedure opens files for input or output, returning a file descriptor string for use with `read` and `puts`.

As well as allowing input and output with the wider system, Tcl allows external programs to be executed:

```
# setbeep.tcl -- Set X Beep Preferences, via xset (1)
label .title -text "Set Beep Preferences"
scale .volume -label "Volume: %" -to 100 -orient horizontal
scale .pitch -label "Pitch: Hz" -to 20000 -orient horizontal
scale .duration -label "Duration: msec" -from 10 -to 1000 -orient horizontal
button .set -text "Set" -command setbeep
button .quit -text "Quit" -command exit

proc setbeep {} {
    exec xset b [.volume get] [.pitch get] [.duration get]
    puts stdout \007 nonewline
}

.volume set 100; .pitch set 5000; .duration set 100

pack append . .title top
pack append . .volume top; pack append . .pitch top; pack append . .duration top
pack append . .set top
pack append . .quit top
```

In this program, three scales may be used to set the preferred sound of the X display's "beep". When the user has set the scales, pressing the "Set" button results in a call to the Tcl `exec` procedure. This takes a sequence of arguments describing an external command, and the arguments to the command. (In fact, redirection and command pipelines are also

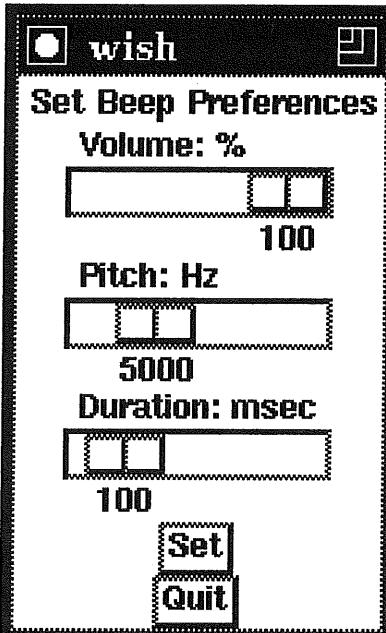


Figure 4: The display from `setbeep.tcl` showing the scales set to change the user preferences to set the X display's beep sound.

allowed.) In this particular case, the X command `xset` is used with the scale values to change the sound of the beep.

Sometimes, it doesn't seem sufficient that graphic display programs act only in response to direct user action. Consider the following program:

```
# users.tcl -- Show How Many Users are Logged In
button .checknow -text "Check Number of Users" -command docheck
label .number
button .quit -text "Quit" -command exit

proc docheck {} {
    .number configure -text "[exec who | wc -l]"
}
pack append . .checknow top
pack append . .number top
pack append . .quit top
```

Here, the number of users on the system is displayed whenever the user presses the "Check Number" button. Notice that this uses the `exec` procedure to execute the Unix `who` command, and pipe the output into the Unix `wc` program to count the lines — the resulting output is returned and used to set the number displayed.

If the user wants the number to be updated *automatically*, without having to press the button, the Tk `after` procedure may be used:

```
proc keepchecking {} {
    docheck
    after 60000 keepchecking
}
keepchecking
```

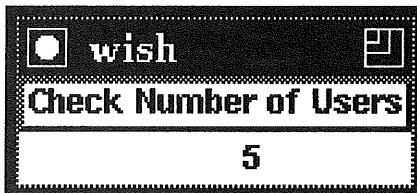


Figure 5: The display from `users.tcl` showing the number of users logged in.

This addition to the above program will cause the number of users to be found out every minute (60,000 milliseconds), and the display updated.

6 Display Management with `pack`

Display management using `pack` involves three main tasks: arranging for a widget to be displayed, arranging layout of the displayed widgets, and arranging for widgets to not be displayed.

Specifying a widget *not* be displayed anymore is easy:

```
pack unpack .title
```

This results in the widget no longer being displayed. Note that the widget and any configuration settings still remains, and it may be displayed again if so wished. If a widget is no longer necessary at all, the `destroy` procedure may be used:

```
destroy .title
```

After this, the widget no longer exists at all, though a new one with the same name may be created later, and newly configured. Destroying widgets that contain other widgets within them results in the destruction of the inner widgets as well, and all widgets may be destroyed by:

```
destroy .
```

The Wish program interprets Tcl line by line, until end-of-file, but remains in execution until all widgets are destroyed — or the `exit` procedure is called.

Specifying display layout is not so straightforward, because the `pack` procedure involves managing flexible arrangements of a changing set of changing widgets. It is very difficult to perform these tasks without involving very complex specifications, and `pack` attempts remain understandable by limiting the specifications possible.

Essentially, `pack` requires that a display be either a vertical or a horizontal arrangement. This may seem annoyingly restrictive, but there are sources of flexibility within the model. Most significantly, `pack` may be used to arrange widgets within a `frame` widget, and then may be used again to arrange `frame` widgets themselves at a higher level. In this way, the total arrangement might be, for example, a vertical arrangement of horizontal arrangements with vertical components.

```

# setaccess.tcl -- Set File Access via chmod (1)
label .title -text "Set File Access"
entry .file -relief sunken
frame .mode
button .set -text "Set" -command {setmode}
button .quit -text "Quit" -command exit

proc setmode {} {
    global modeflag rflag wflag xflag
    exec chmod $modeflag=$rflag$wflag$xflag [.file get]
}

frame .mode.class; frame .mode.access

radiobutton .mode.class.user -text "owner"      -command {set modeflag "u"}
radiobutton .mode.class.group -text "group"      -command {set modeflag "g"}
radiobutton .mode.class.other -text "other"      -command {set modeflag "o"}
radiobutton .mode.class.all  -text "every"       -command {set modeflag "a"}
checkbutton .mode.access.read -text "readable"   -command {set rflag "r"}
checkbutton .mode.access.write -text "writable"  -command {set wflag "w"}
checkbutton .mode.access.exec -text "executable" -command {set xflag "x"}

pack append .mode.class .mode.class.user {top fillx}
pack append .mode.class .mode.class.group {top fillx}
pack append .mode.class .mode.class.other {top fillx}
pack append .mode.class .mode.class.all  {top fillx}

pack append .mode.access .mode.access.read {top fillx}
pack append .mode.access .mode.access.write {top fillx}
pack append .mode.access .mode.access.exec {top fillx}

pack append .mode .mode.class left; pack append .mode .mode.access left

pack append . .title top; pack append . .file top; pack append . .mode top
pack append . .set {top fillx}
pack append . .quit top

.mode.class.user invoke; .mode.access.read invoke; .mode.access.write invoke

```

The above program arranges a display for a program to change the access mode of a file. There is a vertical sequence consisting of a title, an entry box for the file name to be typed, the mode settings, and a button to set the mode and a "quit" button. The mode settings, however, consist of a horizontal arrangement to specify the class of user accessing, and the mode allowed; vertical arrangements of four and three choices, respectively. This shows, therefore, how more complex graphic layouts may be organised with pack. This program contains all the necessary instructions to actually set the file mode with the `exec` call to the Unix command `chmod` in the procedure `setmode`. For a working program, all that must be added are bind commands to allow the user to type in the filename — as was shown in an earlier example. So this program also shows how useful graphic user interface programs may be developed quite easily using Tk and Tcl. (Though note there are many different ways

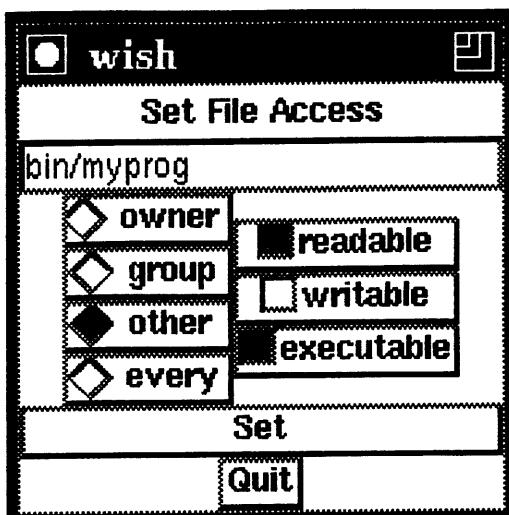


Figure 6: The display from `setaccess.tcl` showing the result of packing small vertical frames within the middle horizontal frame within the vertical outer frame.

which this interface might be designed.)

As well as allowing the linear arrangements to be nested, `pack` also allows some control over the exact placement of widgets within the linear arrangement. In the example above, each of the widgets is packed at the top of the cavity below a vertical arrangement, or at the left of the cavity beside a horizontal arrangement. It is also possible to build up the arrangement in similar ways specifying `bottom` and `right`. Moreover, other options may specify that a widget be padded with some space horizontally (`padx`) or vertically (`pady`), or that it is to fill the flexible space horizontally (`fillx`), or vertically (`filly`). In the above program, for example, the mode buttons and the set button use `fillx` to align all the widgets within their vertical frames.

All the `pack` calls discussed so far have specified `append`. The `pack append` construct is fine for building up widget arrangements, putting them in place as they are needed. There are two other options used in connection with displaying widgets, `before` and `after`, that allow a widget to be placed at a specific point in an existing arrangement. While `append` must be followed by a parent and a child widget, both `before` and `after` involve specification of two sibling widgets, one already mapped, and another to be mapped either before or after the first. While `pack append` is usually sufficient for static displays of widgets, `pack after` and `pack before` are useful when displays are being organised in a complex order, or require re-organisation as a necessary part of the display. For an example of how this can be useful, see the VUW CR system performance viewer, described in a later figure.

7 Program Development with Tcl/Tk

All of the programs and program fragments presented here have been purely Tcl and Tk — with occasional `exec` calls to Unix. Quite useful programs can be constructed in this way. In particular, Tcl/Tk programs can be ideal to construct helpful graphic user interfaces to existing Unix commands and programs, as shown above. Such interface programs can use the various widgets of Tk to make it easy for users to specify various settings and options which otherwise require learning of difficult command syntax. Moreover, the list and string processing capabilities of Tcl, together with the communication procedures, allow Unix commands to be run from Tcl, and for their output to be retrieved and analysed.

All this is especially useful for commands and programs that are not used frequently, and that have complicated syntax: the file protection mode changer `chmod`, and the X preferences command `xset` have been briefly discussed, and the file archiver `tar` and file locator program `find` are also in this category — and there are many others.

Programs that just use Tcl and Tk can simply use the Wish program itself as a command interpreter, and like Shell scripts can be made executable with a first line specifying this:

```
#!/usr/local/bin/wish -f
#This file may now be made executable and will be interpreted by Wish
#(make sure the wish pathname above is correct for your system)
.label .title -text "My Wonderful Tcl/Tk Program"
...
```

When such programs are still in development, it is often useful to use Wish interactively, and use command `source myprog.tcl` to include the program source. In this way the program will be run, but you may still give commands interactively to Wish to explore the program, printing values, reconfiguring widgets, as necessary. The `source` directive is also useful for including sets of useful procedures so they are available for new programs.

Both as an aid to exploring program behaviour, especially event driven behaviour, as well as debugging, the Tcl command `trace variable` is also very useful. It can be used to associate a given variable with a command to be executed every time the variable is accessed. For the `chmod` program shown earlier, the following setup would print helpful lines every time the variable `modeflag` is set (`w`) or inspected (`r`):

```
proc mydebug {name arrayindex usage} {
    puts stdout "Variable $name: Usage: $usage"
    puts stdout "Value is now [expr $$name]"
}
trace variable modeflag rw mydebug
```

Where programs use `exec` to invoke external Unix programs, the Tcl command `catch` is useful for *handling* errors or unexpected results. Normally any failure of an external command results in an error message and call backtrace being printed on standard output. The `catch` command will successfully invoke *any* command, and return the completion code.

Some advanced features of Tk have not been introduced here, but are worth at least mentioning. For instance, it is useful to know that widgets created using Tk can work together with X “resource” facilities to integrate configuration of Tk programs with other X clients. There is also a Tk command `wm` that allows direct communication with the window manager. And lastly, Tk presents an extremely easy-to-use method for arranging communication between Tk applications: the simple `send` procedure arranges for a Tcl command to be executed by the interpreter of a *different* Tcl/Tk application, even if it is running on a different machine. This suggests that a well designed set of applications could offer interesting capabilities by working together: an opportunity that seems worth pursuing.

And while the approach discussed above does allow graphic users interfaces to be constructed very easily, it is important to remember that Tcl and Tk do offer much more flexibility if necessary. New programs can be written that use the ability to work *with* Tcl so that the program can use the Tk facilities for all user interaction from the original design. Whereas the approach described earlier involves using Tcl to *externally* drive existing programs, this approach would involve working with Tcl *internally*, with the host program driving the Tcl component. Further consideration of this approach is beyond this discussion,

but the documentation described later provides all the details necessary. A good way to start is to closely examine the C source to the Wish program itself.

8 The VUW Meters Widget Set

At the Computer Science department at Victoria University of Wellington, our work in management of distributed systems requires various graphical displays to allow easy comprehension and comparison of empirical data. Especially because we wish to explore various approaches in data display, the flexibility of Tcl and Tk appeared very useful. Accordingly, we have developed widget classes within the Tcl and Tk model specifically for display of empirical data; we call them "meters". Some examples of these are described in the table below:

Widget Type	Brief Description
bargraph	shows a varying length rectangular bar against a set scale; length, width, range, scale, orientation, and colour are configurable; operations set and get the current value
dial	shows a varying needle set against a scaled circular dial; radius, range, scale, dial portion, and colour are configurable; operations set and get current value; optional radii mark extrema from recent settings
stripchart	shows a moving sequence of varying height bars against a set scale, indicating the recent values set; length, width, range, scale, colour, and number in the sequence are configurable; operations set and get current value

All the widgets in this set may be used in a similar way to other Tk widgets, and so created, configured, and arranged from Tcl. They may also be set and inspected from Tcl, and so can be used when better empirical display would improve the graphic interface. Consider the following modification to the program presented earlier to show the number of system users:

```
# usersdial.tcl -- Show with a Dial How Many Users Logged In
button .checknow -text "Check Number of Users" -command docheck
dial .number -maxvalue 30 -numticks 30
button .quit -text "Quit" -command exit

proc docheck {} {
    .number set [exec who | wc -l]
}
pack append . .checknow top
pack append . .number top
pack append . .quit top

proc keepchecking {} {
    refresh
    after 60000 keepchecking
}
keepchecking
```

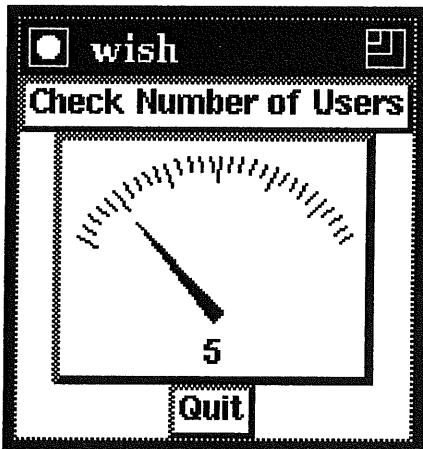


Figure 7: The display from `usersdial.tcl` showing the number of users logged in. Note that this is really more cluttered than necessary, because with `bind` the dial itself can play the role of a button.

Even simple programs like this are quite useful, and can be used to provide clear visual indicators of important varying empirical quantities: free space on critical disk file systems, for example, by using the Unix `df` command to provide the data.

Tcl is interpreted statement by statement, however, and that is a liability where efficient real-time monitoring of live data is required. In addition to the conventional Tcl/Tk use of these widgets, therefore, a lower level of interaction is also provided. As described in the introduction, Tcl is designed to be imbedded into programs to provide flexible and programmable facilities, and Wish is just a simple program that passes input lines to the Tcl interpreter. Like all the Tk widgets and related procedures, the VUW meters are specified by C data structures and functions that are associated with Tcl using the methods Tcl provides to extend itself to suit the host program.

The VUW meter widgets, however, themselves allow *other* functions to be specified to provide values to set the widget. The widgets include operations to `start` and `stop` this background activity, and a configurable `interval` setting that determines how often it takes place. In this way, we have access to the great flexibility of Tcl and Tk for widget arrangement and user interaction – where the interpreted nature of Tcl is an advantage and presents no impact on overall performance. But we also have access to the low levels to efficiently update values very frequently so we can display in real-time live data from various sources in the system.

9 Conclusion

Tcl and Tk are simple in design and structure, though of course they do require some exploration to become familiar with the way they work. However, the interpreted nature of Tcl allows instant feedback, and makes such exploration easy and fun. From a software engineering point of view, the whole approach has a very high power-to-weight ratio.

The nature of the Tcl approach offers great variety in application. In developing interfaces to existing programs and commands, the string orientation and easy access to Unix files and commands are valuable. In developing more sophisticated systems, the ability to embed Tcl in another program is very useful. And the design of Tcl and Tk as an *interpreted* language allows an extraordinary flexibility in what can be achieved, as well as providing the

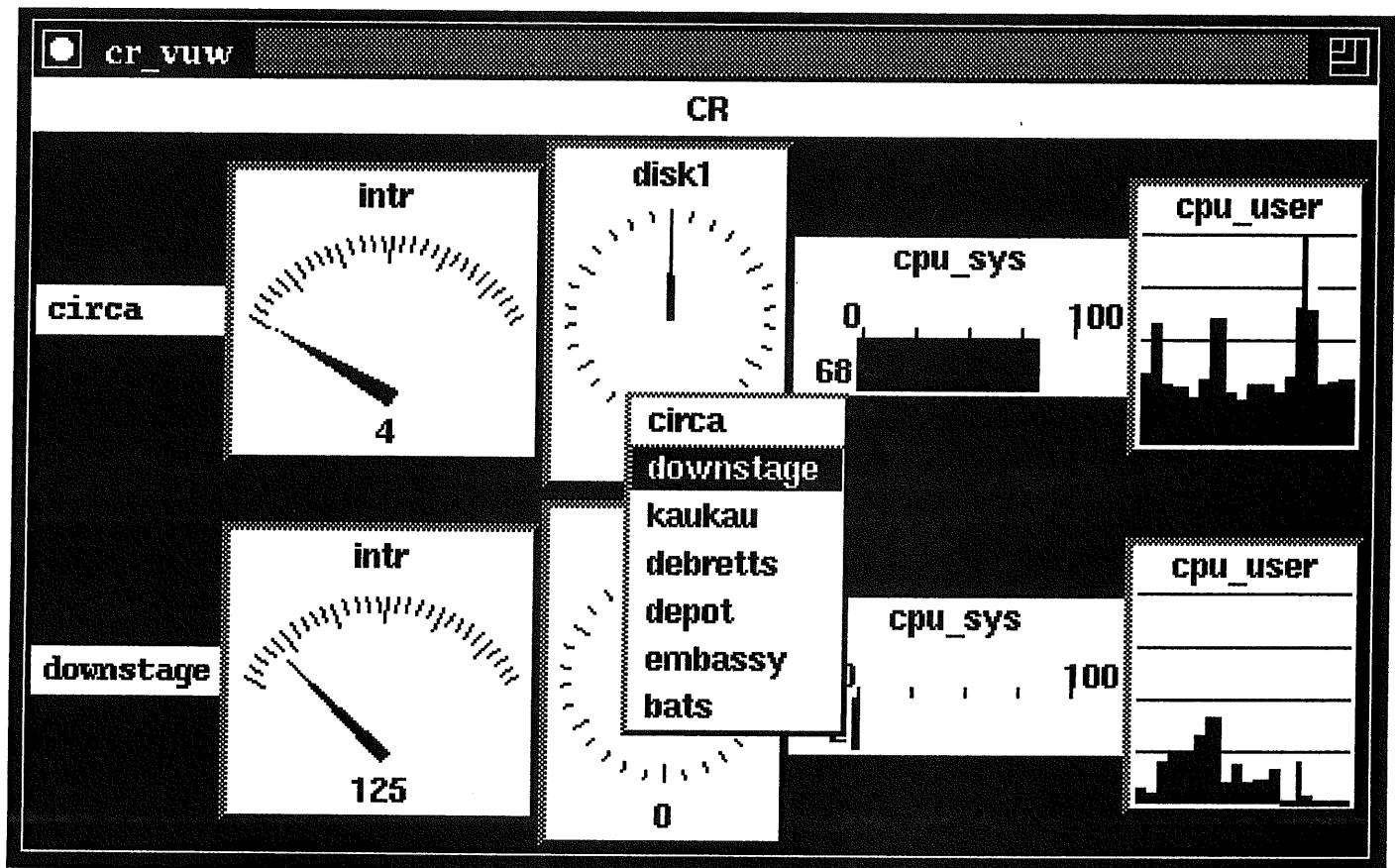


Figure 8: The display from the VUW CR (Control Room) distributed system performance viewer. Each row represents data from a particular machine, and each column a particular performance *metric*, allowing easy visual comparison. Any number of rows and columns may be displayed, and the machines and metrics are chosen by pop-up menu, and placed at the cursor position. The configuration of the meter for any metric may be changed while viewing. Live data is fed to the meters at a low level from other monitor processes.

opportunity to explore while developing.

The approach is still new, and much more experience is needed in using these tools in a wide range of situations. There may well be subtle problems with some of the simplifying assumptions made — though the design is still evolving. However, it does appear that Tcl and Tk at least open the door to easy programming of graphic user interfaces in the Unix and X window system environment.

10 Where to Learn More about Tcl and Tk

The original documents describing Tcl and Tk are by their creator, John Ousterhout:

Ousterhout, J. K., *Tcl: An Embeddable Command Language*, in Proceedings of the Winter Usenix Conference, 1990.

Ousterhout, J. K., *An X11 Toolkit Based on the Tcl Language*, in Proceedings of the Winter Usenix Conference, 1991.

These two papers provide a formal introduction to the aims and design philosophy behind Tcl and Tk, make comparisons with related work, and indicate future directions for development. They are not, however, suitable as detailed reference material for programming with Tcl and Tk.

For details about programming, the main reference source is the set of Unix manual section 3 entries supplied with the Tcl/Tk distribution. These are well written at the detail level, and reasonably complete. The definitive source of information on the Tcl language itself is the entry `tcl` (3). This is more than 30 pages long typeset, and explains the syntax of the language, as well as all the builtin procedures useful in string and list processing, and interaction with the system.

For the Tk commands available directly in Tcl, the documentation is in a set of entries such as `tk_label`, `tk_button`, `tk_pack`, and so on — `tk_` followed by a name in lower case. For the Tel facilities available to a host program written in C, see the entries like `tcl_CrtCommand`, `tcl_Eval`, and so on — `tcl_` followed by a capitalised name. For the Tk facilities available for a host program written in C, see the entries like `tk_ConfigWind`, `tk_GeomReq`, and so on — `tk_` followed by a capitalised name.

The Tcl/Tk distribution includes all of the above, as well as all source in C, and a small set of demonstration programs, and is available free. In New Zealand, the distribution is available from the Computer Science department of Victoria University of Wellington: if on the Internet, use *ftp* to connect to `ftphost.comp.vuw.ac.nz` and see the directory `/pub/tcl`; otherwise contact the department for advice.

A Usenet news group `comp.lang.tcl` has existed since the beginning of 1992, and is a good source of news and advice about Tcl and Tk.

This tutorial was written in May 1992, and there have been several releases of TCL/Tk since then, with many minor and some major changes. Probably the most important change has been the addition of the "place" command that allows detail control over geometry management. Also, a new "canvas" widget allows precise layout of drawings and widgets, and a new "text" widget allows display of multi-font formatted text. Both widgets are designed for complex interaction; the text widget includes "hypertext" capabilities, for example.

TCL/Tk has been installed without difficulty on most Unix/X systems, and is in use in many locations worldwide. A small number of commercial systems have been built that rely on the software, and more are in development. Ousterhout is still working on TCL/Tk, but regards the basic facilities as complete and is currently writing a book about the system. He is committed to TCL/Tk conforming to the Motif "look and feel", and some smaller changes are being made to reflect that commitment.

The current software and documentation is available for anonymous *ftp* from `barkley.berkeley.edu`, directory "`tcl`"; it's also from us at `ftphost.comp.vuw.ac.nz`, directory "`pub/languages/tcl`" (where our "meter" widgets are also available).

Robert.Biddle@comp.vuw.ac.nz
February 1993

An Update of UNIX-Related Standards Activities

by Stephen Walli

Report Editor <stephe@mks.com>
USENIX Standards Watchdog Committee

Report on POSIX.0: The Guide to Open Systems Environments

Kevin Lewis <klewis@gucci.enet.dec.com> reports on the July 13-17, 1992 meeting in Chicago, IL:

First, let me indulge in some self-aggrandizement before going into the details about the POSIX.0 work. A little over a year ago, I projected that the guide document would be going into formal ballot in July 1992. (I am the co-chair of the working group and have a stake in this!) As of the writing of this report, July 16, 1992, the POSIX.0 guide has been sent out for formal ballot by the IEEE. I must add that this would not have been possible without a core of dedicated people within the group, acting as section leaders.

As for the work of this group at the July meeting in Chicago, there were two major areas of activity. The first was the development of the Guide document rationale. The rationale captures the group's memory on those issues it felt were significant and which it felt might surface during the formal ballot process. The audiences for this document will be the section leaders to assist them during ballot resolution, and future members of the working group who might need to understand the thinking of the group on these issues.

This document was completed during the meeting and will be available to the group prior to the October meeting in Utrecht during which the group will be resolving ballots.

The other major area of activity were discussions around the group's coordination with other standards organizations at the ISO level. The group is specifically concerned with WG15, the WG15 Rapporteur Group for Coordination of Profile Activities (RGCRA), and SC22. We have formally requested that the U.S. Technical Advisory Group (TAG) to WG15 seek review and comment of the formal ballot draft by WG15 and SC22. In addition, we asked the TAG to notify the WG15 RGCRA that several members of POSIX.0 would like to participate in their Utrecht meeting in October. The formal ballot draft, along with a cover letter highlighting questions for consideration during the review, was passed to the TAG.

Finally, for those who are interested, the POSIX.0 Ballot Group composition is:

Class	Number	Percentage
Academic	3	3.49%
General Interest	23	26.74%
Producer	24	27.91%
User	36	41.86%
Total	86	100%

We've gotten over the first two hurdles of establishing a balanced ballot group and getting our document out on time. Stay tuned to find out what the response is....

Report on POSIX.2: Shell and Utilities

David Rowley <david@mks.com> reports on the July 13-17 meeting in Chicago, IL:

Summary

September the 16th, 1992 – that's the date people have been waiting for since the POSIX.2 working group was formed more than five years ago. It's the date the IEEE Standards Board is due to approve P1003.2 as an IEEE Full Use Standard. The standard includes both the "Dot 2 Classic" and "Dot 2a" components, previously balloted as separate standards. The IEEE Standard (based on the new Draft 12) is identical (at least from a technical standpoint) to ISO/IEC Draft International Standard 9945-2:1992.

NIST continues to work on a new FIPS (Federal Information Processing Standard) for POSIX.2, expected in draft form by early Fall 1992.

POSIX.2b work is progressing well, incorporating symbolic link support within a number of utilities and a new PAX archive format, and addressing a number of international concerns regarding locales.

Test assertion work continues. The POSIX.2 assertions have almost full coverage, and will go to ballot soon, perhaps as early as October. The POSIX.2a test assertion work is going well, though assertions for *vi* have not yet been attempted.

There is talk that the test assertion work will be renamed P2003.2 instead of the current P1003.3.2.

Background

A brief POSIX.2 project description:

The base utilities of the POSIX.2 standard deal with the basic shell programming language and a set of utilities required for the portability of shell scripts. It excludes most features that might be considered interactive. POSIX.2 also standardizes command line and function interfaces related to certain POSIX.2 utilities (e.g., `popen()`, regular expressions, etc.). This part of POSIX.2 , which was developed first, is sometimes known as "Dot 2 Classic."

The User Portability Utilities Option, or UPUO , is an option in the base standard (previously known as POSIX.2a). It standardizes commands, such as `vi`, that might not appear in shell scripts, but are important enough that users must learn them on any real system.

Some utilities have both interactive and non-interactive features. In such cases, the UPUO defines extensions from the base POSIX.2 utility. Features used both interactively and in scripts tend to be defined in the base utility.

POSIX.2b is a newly approved project which covers extensions and new requests from other groups, such as a new file format for PAX and extensions for symbolic links. It also includes resolution of items arising from comments by ISO Working Group 15. POSIX.2 is equivalent to the International Standards Organization's ISO DIS 9945-2 – the second volume of the proposed ISO three-volume POSIX standard.

POSIX.2 Status

The ISO Draft International Standard 9945-2 that was approved at the May meeting of Working Group 15 is due to be approved at the IEEE on September 16th. There are no apparent road-blocks to the IEEE Standards Board approving the standard, but of course there are very few sure things in life.

POSIX.2 Draft 12 comprises the standard set of utilities ("Dot 2 Classic") and now includes the User Portability Utilities Option (previously "Dot 2a, User Portability Extensions"). Hal Jespersen has done a fine job integrating the two separately balloted standards into one epic tree-killing tome, coordinating it with the ISO 9945-2:1992 Draft International Standard, POSIX.2 's ISO equivalent. The implementors of the world owe Hal a debt of thanks for ensuring that the ISO and IEEE standards can be technically identical.

FIPS and Certification

NIST continues to work towards a new FIPS (Federal Information Processing Standard) for POSIX.2 . Verifiable conformance to the standard is now the critical issue. Fortunately, it seems as though good progress is being made within the standards industry on coming up with a well-endorsed solution. X/Open has issued an RFQ (Request for Quotation) for an Integrator to put together a joint POSIX.2 and XPG4 Commands and Utilities verification suite. This work points towards there being a single validation suite for both the POSIX.2 and XPG4 implementations of the shell and utilities, again making life much easier for implementors and users alike. The XPG4 commands and utilities specification comprises a superset of the POSIX.2 utilities. The X/Open suite will allow verification of the XPG4 superset as well as the POSIX.2 subset.

NIST will likely point to this suite, once in place, as the yardstick for gauging conformance to the POSIX.2 FIPS.

The suite will likely be finished towards the end of 1993.

PAX File Format

The group continued to define the new PAX file format, but are now intent on verifying the sanity of using the ISO 1001 tape format as a base format. A posting to "comp.std.unix" requested feedback and input as to the appropriateness of ISO 1001 , along with a request for alternate proposals. The proposals will be discussed at the Utrecht meeting in October.

The group also modified the proposal for codeset representation of filenames, user names, etc. contained in the archive. The format that will be used is now specified as UTF (UCS Transformation Format). A slight problem with this exists because the UTF description is contained in Annex F of the ISO 10646 Unicode standard, and is only informative rather than normative. The group is therefore (a little) hesitant to point to it, but feels the space savings and the inherent seamless ability to upgrade to the full 32-bit codeset (UCS4) overcomes these objections.

Working Group 15 Requirements

The group also examined the Working Group 15 (ISO) requirements for the next revision, as outlined in Annex H of the ISO Draft International Standard 9945-2:1992. Most of the issues centered around the definition of locales, specifically codeset issues. A number of specific proposals are pending from the ISO member bodies, includ-

ing something similar to trigraphs for the *sh*, *awk*, etc., extensions to locale character class definitions, reincorporation of the substitute facility, relaxing of the restriction on NUL collating lower than all other characters, support for state-dependent characters sets (such as shift encoding), and a general character translation utility (perhaps X/Open's *iconv*).

These issues will be discussed further at the Utrecht meeting on October 22nd, 23rd (just before the next WG15 meeting).

Test Methods

POSIX.3.2 Test Method work is progressing well, with almost all of the assertions corresponding to the current draft of POSIX.2. The group expects to go to ballot sometime around October.

Work on the UPUO test methods also progressed, with only a few gaps remaining. The daunting *vi* command still strikes fear in some that would approach it, and has not yet been addressed. This will be worked on at the Utrecht meeting.

Report on POSIX.5: Ada Bindings to POSIX

Del Swanson <dswanson@email.sp.unisys.com> reports on the July 13-17, 1992 meeting in Chicago, IL:

The POSIX.5 group has been working to produce Ada language bindings to POSIX standards. As of June, 1992, the IEEE Standards Committee has approved the Ada binding to POSIX.1 as a standard, designated POSIX.5. It should be published as an IEEE standard by the end of the year. Congratulations all around to the working group, the ballot resolution committee, the ballotters, and all the supporting employers, spouses, lovers, etc.

At this time, it is not expected that this document will become an ISO standard, because of its format and derivation. POSIX.5 is a "thick" binding: it can be read by itself, since it duplicates the descriptions of all the functions, in addition to describing how they relate to the Ada language. And POSIX.5 is derived from the POSIX.1 C binding, since no Language Independent Specification (LIS) yet exists. ISO requires that language bindings be "thin," not duplicating any information present in the base document, and that they be bindings to an LIS.

TCOS-SS (the IEEE committee responsible for all POSIX standards) had previously agreed that POSIX.5 could be approved as an IEEE standard in its current form. It would not be submitted for ISO standardization. A new version of the standard (which will then be submitted for ISO stan-

dardization) will be produced after the LIS is approved, and after the revision of the Ada language, now expected to be finalized in 1994.

Meanwhile, there has been a reaction from the European community, and from members of ISO Working Group 9 (on Ada) that there should be an Ada binding of POSIX officially sanctioned by ISO. At the July POSIX meetings, therefore, we recommended to TCOS-SS that it suggests to ISO Working Group 15 (ISO POSIX) that POSIX.5 be approved as an ISO "Committee Document."

Now that the IEEE standard has been approved, it is incumbent upon the group to resolve interpretation questions. Officially, this involves the formation of an interpretation committee (on which nearly the entire group sits). The intent is to explain interfaces, elaborate semantic descriptions, and define the implications of problematic interface specifications. About ten interpretation requests have been received to this point. The TCOS approach is that this interpretation group adds nothing normative to the standard, even by logical extension. Any such specifications must be done by balloted revisions to the document.

The major current activity of the group is the development of bindings to the Real-Time Extensions standards being developed by the POSIX.4 group. The binding to POSIX.4 will be relatively straightforward. This is especially true since a draft thin binding to POSIX.4 has been prepared by one of our members at Florida State University with financing from the U.S. Army.

This draft has now been updated a couple of times by the group, and is ready to be massaged into IEEE format, with a few changes reflecting the latest POSIX.4 draft. This POSIX.20 draft 1 is planned to be circulated for mock ballot after the October meetings. Our goal is to have POSIX.20 approved as a standard hard on the heels of POSIX.4 LIS.

This schedule is somewhat of a change from our previous assumption that we would produce a unified binding to POSIX.4 and POSIX.4a (threads extensions). Our current direction is to proceed directly with balloting the binding to POSIX.4, and work concurrently on the binding to POSIX.4a. The advantages are that this reflects the document structure of the POSIX.4 group, that this approach will fill the needs of some users sooner, and that the approval of the POSIX.4a standard is likely to be significantly later than POSIX.4.

Meanwhile, we have also agreed to assist in the production of the POSIX.4 LIS. The new technical editor of this document has been a joint member

of the POSIX.4 and the POSIX.5 groups. The members of the POSIX.5 group are committed to help him and the POSIX.4 group to produce the LIS as quickly as possible.

The production of a binding to POSIX.4a is going to be significantly more complex, because of the interplay of two separate modes of intra-process concurrency, Ada tasks and POSIX threads. Complicating the issues is a difference of philosophy among members of the group, which is probably reflective of the community at large.

A key question that differentiates the philosophies: should operating system functions be visible in a binding if the language itself provides parallel functionality? Several other issues ensue. Should functions be visible that, if called directly, may interfere with the assumptions and operations of the language support library? Would it be acceptable to isolate such functions to emphasize their danger? Is it adequate (or acceptable) to assume that Ada compilers will allow calling such functions via language interface conventions?

One of the greatest technical challenges to the POSIX.4a binding is to determine the implications of interactions among processes in a multi-language environment. The feasibility of mapping tasks to threads is being demonstrated in prototype implementations. But some potential conflicts caused by the interactions of the two entities are becoming apparent.

We are assuming that these conflicts must be resolved, since at a minimum Ada programs will want to make use of libraries written in C, such as GUI and DBMS packages. We are starting to catalog such potential conflicts, which revolve around the creation and destruction of threads/tasks, parent-child relations of threads/tasks, and the handling of exceptional conditions. We have barely begun the resolution process.

Meanwhile, members of our group are involved with two efforts that are prototyping implementations of Ada bindings to the Real-Time Extensions (including threads). As it happens, this is not only valuable input to our effort, but a few problems have been found with the base document drafts that have been passed on to POSIX.4.

In preparation for the next meeting, we have volunteers to analyze issues with task/thread interactions, and to propose directions and bindings to synchronization and scheduling functions. We hope for significant progress on these issues, as well as completing preparations for the mock ballot.

Report on POSIX.7: System Administration

Bob Robillard <duke@cc.bellcore.com> reports on the July 13-17, 1992 meeting in Chicago, IL:

Overview of POSIX.7

Since this is the first snitch report on POSIX.7 in quite some time, I'll start with some background. (If you already know what POSIX.7's been up to for the past year or so, you can skip ahead some). POSIX.7 is one of the three POSIX "Base Standards" (POSIX.1 and POSIX.2 are the other two). It covers the kinds of commands typically found in section (8) of the man pages – things like *fsck* and *init*.

Early on, POSIX.7 decided to address distributed system administration, rather than just single machine administration, in the belief that networked computing is the way things are going. This has caused a great deal of trouble since distributed system administration is relatively new and perhaps less ready to be standardized than stand-alone administration. The hope, however, is that the final standard will be more useful.

In the last year, POSIX.7 broke its work into several pieces. Each area of system administration is getting its own document. The current POSIX.7 "sub-groups" are:

POSIX.7.1 – Printing Administration,
POSIX.7.2 – Software Installation and Management, and

POSIX.7.3 – User and Group Administration.

POSIX.7.1 – Print Administration

The Printing group is probably the furthest along, since they held a mock ballot in June. The base for their document is the Palladium print system which was originally developed as part of MIT's Project Athena. It is now included in the Open Software Foundation's DME project. The document specifies print commands, a programming interface, and a set of managed object definitions. (More on these later.)

Palladium is the reference implementation of the ISO Document Printing Application Standard (DPA), currently in international ballot as a Draft International Standard (DIS) under working group ISO/IEC JTC1/SC18 (the official name of the ISO document is: Information technology – Text and office systems – Document printing application (DPA) – Part 1: Abstract-service definition and procedures, September 1991). It's a client/server distributed system.

One of the reasons for the mock ballot was to determine whether Palladium was an acceptable choice for a base. Since *lpr* and *lp* (both the pre-SVR4 and SVR4 versions) are much more widely used, the group was concerned that their standard would be voted down on "not-existing-practice" grounds.

The people in the mock ballot okayed Palladium. Eleven (11) said Palladium would be okay, nine (9) said it would be okay if some changes were made (changes the POSIX.7.1 group then adopted) and only five (5) were against it. Astute readers will note that this was a small mock ballot group, but it was at least a well-rounded one with 6 University people, 10 from computer or operating system vendors (NeXT, IBM, Sequent, USL, Sun, OSF, Intergraph, Fujitsu), and 4 from user companies (US West, Bellcore, British Telecom, Boeing).

The No votes on Palladium were particularly strong, however, so the group is still concerned. If you have an opinion on this either way, please contact the author.

POSIX.7.2 – Software Management

The Software Management group is not far behind the printing group. The draft document is stabilizing and should go to mock ballot soon. It includes commands to install and upgrade software packages. It also includes managed object definitions, but no API.

The base of their standard is HP's *swinstall* tools and USL's *pkg* tools. Currently, the commands look more like the HP commands, but that is still in flux.

POSIX.7.3 – User and Group Management

The User and Group Management subgroup is still in the early stages. They have been gathering submissions for their base documents, and have been trying to determine a course of action.

There has been some debate about whether User/Group Management is a mature enough area to standardize, and the POSIX Project Management Committee (PMC) suggested that this group publish a Guide rather than a standard. You can expect these issues to be cleared up in the near future, and a solid direction to form soon.

Managed Objects?

All of the POSIX.7 documents are providing descriptions of "managed objects" for their area. I'm not an expert on this, but here's everything I know about it.

Managed objects are hot in distributed management. UI's Atlas, OSF's DME, HP's OpenView, the Object Management Group (OMG) — everyone who's anyone in the field is using them. I think they come from network management, where the "object" being "managed" was a physical thing (like a router, for example.)

The concept is that there's an object out there, and to do something, you send it messages. The Print document, for example, has the concept of a printer object. If you want to know what kind of paper a printer has, you send a message to the printer object and ask for its "media-supported" attribute. There are objects for print jobs, software packages, etc.

The idea is that these "managed objects" work well with distributed systems because you don't have to know where the printer is — the message sending mechanism deals with that. Also, they are an aid to interoperability, since all POSIX compliant software will have to support the same set of objects.

Road Blocks

Fair warning: I'm now going to get up on a soapbox.

The next step for the POSIX.7.1 document would seem to be to go to ballot. There are, however, two things standing in its way. First, all documents need to have test assertions written before entering ballot.

Test assertions are statements about what a function or command does, written in such a way that someone could easily write a shell script or program to check that an implementation actually does the correct thing. For example: "If *lpr* is given the name of a non-existent file, it returns the following error...." (There are formatting details about test assertions, but that's the basic idea.)

Although having these test assertions is clearly valuable, writing them is a tedious, time consuming process, and it is likely to delay ballot by several meetings. Also, since many details of the commands and functions are likely to change during ballot, many of the assertions will need to be thrown away.

Less clearly valuable is the Language Independent Specification (LIS) of the function calls, which also needs to be written before a draft goes to ballot. The functions have to be abstracted from C to an invented specification format which is free of programming language dependencies.

The idea of this is to remove any parts of the API that are implicitly dependent on C syntax, such as return values from functions, pointer parameters, or the use of structures. Only the functionality should remain.

The group then writes a companion "C thin-binding," which doesn't describe what the functions do, it just talks about how the functionality described in the LIS is implemented in C.

I believe the goal of the LIS is to make it easier for people interested in an Ada or Fortran version of POSIX to write the appropriate language binding for it. Again, this is tedious and time consuming, and will likely eat up several meetings of POSIX.7's limited resources.

Report on P1224: X.400 API

Steve Trus <trus@duke.ncsl.nist.gov> reports on the July 13-17, 1992 meeting in Chicago, IL :

Introduction

The Chicago meeting was productive for the P1224 working group, and we are very near the completion of the standardization of the P1224 and P1224.1 documents.

At the Chicago meeting the group:

- planned the next P1224 ballot resolution meeting,
- reviewed the JTC1 recommendations for the International Standardization of the IEEE X.400, POSIX.17 Directory Services, and Object Management APIs,
- planned future work for the P1224 group,
- presented the status of the IEEE balloting of P1224,
- presented the status of the IEEE balloting of P1224.1,
- resolved the ballot objections and reviewed the ballot comments for the P1224 and P1224.1 documents, and
- planned the recirculation of the P1224 and P1224.1 documents.

P1224 Next Ballot Resolution Meeting

The group will not meet with the other TCOS groups at the October meeting in Utrecht, NL. We agreed to meet November 16-20 at NIST (Gaithersburg, MD).

International Standardization of the APIs

JTC1 has recommended that the IEEE P1224 and P1003.17 working groups split each of the X.400, X.500 and Object Management API documents into four separate documents (Language Independent Specification, Test Methods for Language Independent Specification, C Language Binding, and Test Methods for C Language Binding). Additionally, JTC1 has recommended that the IEEE submit the X.400, X.500 and Object Management API documents to JTC1 for fast-track when they are approved IEEE standards, and that members of the P1224 and POSIX.17 working groups solicit international support for these IEEE standards in order to increase the likelihood of a successful fast-track. The P1224 group agreed to follow these JTC1 recommendations.

P1224 Working Group Status and Future Plans

The first recirculation of the P1224 document began on May 20 and it ended on June 19. The balloting pool consists of 73 members. The balloting for the P1224 document closed with 81% of the ballots returned and 78% of the eligible voters approved the document.

Plans for standardizing future X.400 related APIs were discussed. The X.400 API Association and X/Open will have stable base documents for a P7 and an EDI API by the end of 1992. Tentatively, we would like to begin converting these documents into IEEE standards at the January 1993 meeting.

P1224.1 Balloting Status

The P1224.1 balloting period began May 6 and it ended June 5. The balloting pool consists of 50 members. The balloting for the P1224.1 document closed with 77% of the ballots returned and 82% of the eligible voters approved the document.

P1224 and P1224.1 Ballot Resolution and Recirculation

The group spent three days resolving the ballot objections and reviewing the ballot comments for the P1224 and P1224.1 documents. The technical editor will incorporate the changes into the document.

A 10 day recirculation of the P1224 document was scheduled to begin October 4 and end October 14. A 30 day recirculation of the P1224 document was scheduled to begin October 10 and end November 9.

Summary

The progress of the P1224 working group is very good. We hope to have the P1224 and P1224.1 standards complete early 1993. The primary function of the November meeting will be P1224 and P1224.1 ballot resolution.

Report on The Proposed ROSE API

*David Cannon <D.Cannon@EXETER.AC.UK>
reports on the July 13-17, 1992 meeting in Chicago, IL.*

A project authorization request (PAR) has been submitted to the Distributed Systems Steering Committee (DSSC) for review, covering the Transparent Remote Operations Interface (TROI) proposal. The first ROSE meeting presented the details of the proposal, and a second meeting on Friday was a BOF on the technical content.

The presentation was led by J.J. Cinecoe and Dan Shia. J.J. Cinecoe anticipated the obvious question of "why do we need a Remote Operations Service Elements (ROSE) API?" He proposed that the work of the TROI group was essential for two reasons:

- it provides vendor OSI application portability,
- there is a requirement by large corporate users who want to write specialised applications which are needed to be portable across multiple platforms.

ACSE (Association Control Service Element) is too complex for a user-programmable platform; ROSE is perceived to better fill the need, and offers a "true" peer-to-peer relationship with another end-system.

The purpose of the proposed project is to generate an IEEE API for the classes of operations defined by ROSE. It is intended to co-locate meetings of the group with both the OSE Implementors Workshop (OIW) and the IEEE POSIX meetings. If both of these options are used the group would be meeting on average every six weeks. [ed. – *The OSE in OIW is "Open Systems Environment"*]. This is the NIST supported group, which used to meet as the OSI Implementors Workshop, and has recently had its scope expanded.]

A quick overview of ROSE vs. RPC:

- RPCs tend to be very proprietary.

RPCs bundle together:
interaction semantics
data transfer

- ROSE unbundles these and provides a variety of synchronous and asynchronous classes of operation.
- transfer of user-defined data streams.

ROSE can provide an equivalent service to RPC across different platforms.

Dan Shia described the Computing Environment on OSI (CEO), of which TROI is one component. The aim is to enable the construction of highly efficient distributed concurrent systems by providing a very thin API over the top of the protocol engine, and is based on OSI ACSE (Association Control Service Element), ROSE, and ASN.1 (Abstract Syntax Notation 1).

The proposal is based on experience gained from an implemented, working testbed.

Someone wanted to know what the difference was between this proposal and the POSIX.12 (Protocol Independent Interfaces) Simple Network Interface proposal. Dan suggested that the main differences were user-defined data presentation and remote operation facilities.

Dan outlined the problems involved in using full ASN.1, which is unparseable, and went on to describe ASN.C. This incorporates a simplified data definition language enabling the automatic creation of data objects, together with the ASN.1 data manipulation language and can be handled, for example, by a C language preprocessor.

The ASN.C specification allows data-object specification through statements which can be mapped on to functions or to extensions of the C language. These could ultimately call XOMcreate to generate the data objects. XOM is being standardized by the IEEE's P1224 working group, and is based on X/Open's Object Management API.

Someone asked whether XOM could do the work of encoding the ASN.1 rules for a particular data-object. Dan said it could for public objects, but wasn't very good at handling user-defined objects.

Dan went on to describe some RPC shortcomings, which include the inability to support:

- all ASN.1 types
- callbacks
- multicast
- peer-to-peer interactions

He also described some limitations of XAP and P1238, (the IEEE's FTAM working group,) including:

- over-complexity for applications writers
- non-integrated naming service
- non-integration of IPC and ITC (Inter-Thread Communication) support.

He concluded that this led to the inherent attraction of the CEO system, which amongst others provides for:

- RPC (blocking) support
- request/reply (non-blocking) support
- multiple underlying protocol stacks
- peer-to-peer operations

The relatively high-level approach offers a number of plusses, for example, a short training period, rapid OSI application development, the ability to port existing applications to the OSI environment, and suitability for development of server applications.

In summary TROI is an attractive proposition; the main problem from an IEEE viewpoint is that much of the elegance is dressed in extensions to languages – ASN.1, C, and any other required language binding – and languages are not the province of TCOS-SS. (TCOS-SS, the Technical Committee on Operating Systems – Standards Subcommittee, is the organizing group within the IEEE for the POSIX related standards efforts.) If they can be shown to be justifiable and useful the APIs could be worked under the TCOS umbrella, but there are some fears that the API work alone may not offer substantially more than P1003.12 and the XOM work.

Report on ANSI X3B11.1: WORM File Systems

Andrew Hume <andrew@research.att.com> reports on the May 11-15, 1992 meeting in Pasadena, CA:

Introduction

X3B11.1 is working on a standard for file interchange on random access optical media: a portable file system for WORMs or rewritable optical disks. TC15 is a committee within ECMA that works on file system standards. This report covers the last three X3B11.1 meetings in Santa Clara, California, Denver, Colorado, and Pasadena, California and two recent TC15 meetings in Denver, Colorado and Reading, England. In brief, we have an ECMA standard!

Pardon my laggardly snitching; I have been snowed under this year. In trying to meet the deadline for the June ECMA General Assembly meeting, I have attended 5 standards meetings in the first six months of 1992 (all but one was a full week) and I redacted new drafts for every one.

ECMA - 167

Editorially, ECMA-167 is arranged as five separate parts. Semantically, these form four independent standards. (Part 1 contains general references and definitions.)

Parts 1 and 2 describe a general scheme for recognising standards used to record the medium (is it ISO 9660, ECMA-167, or perhaps both?) and for recording boot blocks.

Parts 1 and 3 describe a volume structure standard, which includes support for volume labels, volume sets, volume partitions, and logical volumes (which may span multiple physical volumes).

Parts 1 and 4 describe how to record hierarchical file systems (assuming we have a suitable underlying volume structure scheme). The file system is approximately a POSIX (ISO 9945-1) file system augmented by extended attributes.

Parts 1 and 5 document the arcana of record-structured files. ECMA-167 has to support record-structured files, if only for backward compatibility with ISO 9660, and making it a distinct part allows other standards to easily use the same specification.

An important aspect of each of these parts is their interfaces. The input interface describes what the part needs in order to work. The output interface describes what the part allows you to specify (and perhaps use as input to another part). As an example, Part 5 (record structure) has a single input, the data space of a file, and two outputs, the identification of record formats and record display attributes.

International Activity

There is a lot of international interest in volume and file structure standards, particularly for removable optical media. That is why our committee has an ISO standard as its main goal, rather than an ANSI standard. That is also why we have bent over backwards to solicit input from, and work with, Europe (ECMA), Japan (JNC), and ISO (SC15).

We reached our first major milestone on June 25 when the ECMA General Assembly accepted our draft as ECMA-167 by a vote of 30 yes, 0 no, and 1 abstention. Regrettably, the General Assembly chose not to forward the standard for fast-track processing within ISO at this time; it will probably do so at its December meeting.

With the exception of France, we do not expect any problems when ISO SC15 processes ECMA-167 as a DIS. France's objections draw mainly from a French company's claim that adopting the standard will have dreadful performance impact on that company's products. We have discussions ongoing about this and other issues but our basic response is twofold:

- Our standard is an interchange standard. There is no intent that integrated applications adopt our format for their internal data format. They might, however, adopt our format for the import and export of data. As a concrete example, we do not anticipate that Epoch's Infinite file server product will change to use our format for their disk format (although they could). However, Epoch might interchange files into and from their server in our format.
- While we have spent considerable effort to minimize the number of seek's to access files and their data, the bottom line is that for good performance, you will have to have some kind of cached database that maps file or directory names to disk addresses. Optical media, particularly 12in media, is just too big and too slow (although a cache helps relatively fast magnetic disks as well). We decided that a portable high-performance cache was a contradiction in terms and too hard to specify in any case, and so we left it to each implementation to decide what, and how, to cache.

Future Activity

The work in ECMA and TC15 has one single focus, getting the standard into the ISO fast track process. From here on in, the process is purely political. Other than acting as a technical resource, I am pretty much a bystander now.

The process in X3B11.1 is, unfortunately, just as political. Because X3B11.1 is a sub-committee, our parent committee, X3B11, has to approve most of our official activities, and in particular, drafts for processing as ANSI standards and positions for the US TAG to SC15. Ordinarily, this is not a problem but recently, a couple of members of X3B11 starting throwing as many roadblocks in our way as possible. As ANSI has more procedures than probably any other standards organization in the world, this could mean considerable delay in

ANSI processing. As a result of all this hooey, X3B11.1 is changing its focus from technical issues to politics and has now scheduled its meetings concurrently with X3B11 so we can at least argue our own case and cut down on the amount of misrepresentations and falsehoods being made about our committee and its work. (I gave a well-received presentation on our work at the August X3B11 meeting and was present during discussions of X3B11.1's work; this was a real win.)

Just remember, the technical content of a standard is very important, but getting a draft through the standards process is just as important.

Electronic Distribution of Standards/Drafts

Since I became technical editor of X3B11.1, my drafts have been available electronically by both *ftp* and email (*netlib*) from *research.att.com*. (For *ftp*, login as *netlib*.) For details, get *index* from *research/memo*.

As far as our standard is concerned, there are three documents:

- The standard itself (121 pages including TOC and index). (Of course, it can't be the actual standard as ECMA owns the copyright on that but rather, the final draft of TC15; ECMA takes this draft and reformats it using a word-processor program and then publishes it.)
- A technical overview (12 pages). This gives a high level overview but has significant technical content.
- A programmer's guide (20 pages). A low level guide through the standard from a C programmer's point of view. It gives you enough details to design an implementation and do most of the implementation.

Finale

Finally, we have a standard and can now complete our implementations. Although there is considerable procedural work to do, the hard stuff is finished. The technical work has been quite interesting, as has been the role of technical editor. (Mind you, I am scarred for life; I can read standards quite easily now and find myself tsk'-ing at poorly written ones.) Writing a precise description of a nontrivial system is obviously hard, but you never appreciate how hard it is until you do it and then have a whole bunch of folks ballot on it.

If you wish to comment on the standard, get a copy electronically, or contact me or the X3B11 chair (Ed Beshore) for a copy. I will make sure any comments sent to me go to the right folks.

If you would like more details on X3B11.1's work, you should contact either me <*andrew@rsearch.att.com*>, 908-582-6262) or the committee chair, Ed Beshore <*edb@hpgrla.hp.com*>, 303-350-4826).

An Update of UNIX-Related Standards Activities

by Stephen Walli
Report Editor
<stephe@usenix.org>
USENIX Standards Watchdog Committee

Report on The IEEE Standards Board

Mary Lynne Nielsen <m.nielsen@ieee.org> reports on the June 1992 meeting:

The meeting was the occasion for the approval of two more POSIX standards and further activity concerning IT standards in general.

NesCom and RevCom Actions

Two TCOS standards were before the IEEE Standards Board Review Committee (RevCom) for approval as IEEE standards at this meeting – POSIX.5, the POSIX Ada Language Binding to IEEE Std 1003.1-1990 (ISO/IEC 9945-1: 1990), and POSIX.9, the POSIX FORTRAN 77 Language Binding to IEEE Std 1003.1-1990. Both were approved straightforwardly, which is a credit to their chairs for completing the difficult work involved in coordination.

One lesson to be learned from their experiences – RevCom requires that the names of negative ballotters be attached to each negative ballot when these objections are submitted with the RevCom package. It was a bit of a scramble for the chairs to come up with the appropriate documentation. Chairs should ensure that their records clearly reflect committee actions.

The IEEE Standards Board New Standards Committee (NesCom) also had a revised Project Authorization Request (PAR) for POSIX.5 on its agenda. (Seems they had never revised its original PAR, which said it was doing an Ada binding for all of POSIX!! Don't want to imagine the size of that document!) This PAR had been lost in the shuffle for awhile, but NesCom agreed to consider it at the same time as RevCom, in an exception to their rules. It was approved straightforwardly.

The unapproved PAR for POSIX.19 (the Fortran 90 binding to POSIX.1) remained unapproved, as the working group did not explain its relationship to the X3 Fortran committee in a satisfactory manner to NesCom. This will appear on the NesCom agenda again in September.

Congratulations all around to those folks involved in POSIX.5 and POSIX.9. Developing a consensus standard is a long and painstaking process, and everyone deserves a great deal of credit for finally getting there!

The most wide-ranging actions that affect TCOS, however, occurred in groups other NesCom and RevCom.

IT Funding

The Standards Board had created an ad-hoc committee in March to look at the issue of funding of Information Technology (IT) activities. The American National Standards Institute (ANSI) had made a proposal that the cost of involvement in international standards development in the IT area be covered by the individuals involved in those activities. This would mean that anyone involved in these standards would be charged a fee to cover the administrative costs that ANSI incurs as the secretariat to JTC1.

As the IEEE is a major developer of standards in this arena, the subject concerned the Board greatly, and in March an ad-hoc committee was appointed to review the issue. At the June meeting, the committee reported that it recommended that interim support be given to the JTC1 secretariat contingent to the IEEE receiving a seat on the committee that oversees this involvement. It further recommended that professional opinions be obtained as to the legal, financial, and tax implications of IEEE committees being assessed for the financial support of ANSI secretariats. The final report of this committee is expected in September.

One note: this subject was discussed in great detail at the TCOS Standards Executive Committee (SEC) meeting in July, and a motion was passed that recommended general support while encouraging involvement of IT standards developers in any final decision. This resolution has been forwarded to members of the Standards Board ad-hoc committee as a contribution.

Other board news involved reports on the JTC1 TAG (Technical Advisory Group, the US national member group to JTC1). The IEEE had voted "no" on the proposed merger of X3 and the JTC1 TAG, which had been proposed in several forms for the past six to nine months. The proposal for this

merger has now been dropped. Changes to the JTC1 TAG procedures were recommended from the meetings on this issue, however, and those are expected to be developed in the future.

The JTC1 TAG also authorized three simultaneous ballots in the IEEE and in the JTC1 TAG of P1224, P1224.1, and POSIX.17. This is a ground breaking process that should result in faster advancement of these standards into the international arena.

Finally, the IEEE Standards Board Procedures Committee (ProCom) took action on the ongoing requirement for approval letters from companies to include company acknowledgments in a standard. ProCom, after approving this process last December, voted in June not to include such company acknowledgments in standards.

ProCom felt that the policy of obtaining a letter of permission from each company still allowed the possibility that the person writing the letter was not the appropriate person to authorize the acknowledgment. In addition, there was no equitable way of acknowledging everyone associated with the standard by having some companies send in letters and some not. As such, ProCom felt that it was simpler not to include company acknowledgments at all.

The only problem with this, of course, is that ProCom announced this policy and began to implement it just six months ago. Many groups have begun to do all the leg work involved in getting letters signed by the appropriate personnel in their departments, and those letters have been coming into the IEEE. As such, ProCom made a somewhat awkward policy change, which only exacerbates the perception that "they're always changing the rules."

ProCom was well aware that this perception could exist, and discussed various ways to try to record their rationale for such changes. Nevertheless, they felt the implications of this policy were too unsettling to allow it to continue for a longer period of time.

By the way, this series of Board meetings was the first held outside of Regions 1-6. Region 9 hosted this meeting in San Juan, Puerto Rico. The Board was received enthusiastically, and the week was devoted to extra sessions and inclusions of special seminars. This was a result and a reflection of the IEEE's worldwide membership and was a large success.

Report on POSIX.6: Security Extensions

*Charisse Castignoli <charisse@Smallworks.com>
reports on the July 13-17, 1992 meeting in Chicago, IL*

The POSIX.6 group continued to work on new project authorization requests (PARs). Two PARs have been submitted to the Project Management Committee (PMC). They are:

- A Secure General Terminal Interface (GTI)
- Identification and Authentication

Other PARs, such as a portable interchange format, have not been submitted to the PMC due to the lack of resources to work on them.

In response to requests by individuals to assess whether or not POSIX.6 could go out as a trial use standard, Mike Ressler suggested we carefully analyze this approach. We need to go back and look at what the Trial Use definition from the IEEE is, and try to determine what the right approach is for POSIX in general, NOT just POSIX.6.

Monday:

The POSIX.6 ballot resolution committee continued to slog through the comments and objections. We work individually on our laptops, and then send a merged document back to Bellcore. Mike Ressler and his horde of great editors then patch together our individual sections and email out the updated sections.

Of course, you can imagine what happens when a laptop breaks down. The person depending upon it instantly becomes an order of magnitude less productive. This week's session began with the power supply failure of one of the laptops. No problem, says customer service, we'll ship you a new power supply overnight and have you up and running in no time. We should have started taking odds on whether the power supply or the end of the meeting would arrive first!

Despite our hardware limitations, the committee still struggled on....

Tuesday:

We spoke for a long time about multi-level directories and whether or not they should be in the standard. Multilevel directories, are a technique used to solve the problem of public directories (such as `/tmp` and `/usr/spool`). In a trusted system with more than one sensitivity level, a process at SECRET cannot view files created by processes at TOP SECRET. To solve this problem, the idea of creating non-visible subdirectories, one for each

level, was hatched. Processes without a privilege will only see files in their subdirectory. To this process, the pathname would look like `/tmp/mysecretfile`. But to a process with the multilevel privilege the pathname would look like `/tmp/SECRET/mysecretfile`.

Kevin Brady pointed out that there were very few existing applications that actually needed to view the resulting true multilevel directory. Most applications just want to create a file and are unaware of whether the underlying directory is multilevel or not.

For example, in current UNIX trusted systems, `vi` writes file to `/tmp`. `vi` is unaware that `/tmp` is a multilevel directory, however `expreserve`, the program that reclaims `vi` drafts from `/tmp`, is multi-level aware.

Our power supply didn't arrive today....

Wednesday:

One of the most controversial ballot resolution issues we face is that we do not have a consistent storage and allocation model for the data structures that the POSIX.6 interfaces manipulate. Some functions, such as the Mandatory Access Control (MAC) and Information Labels (IL) interfaces, lend themselves to persistent opaque data types. Others, such as the Access Control List (ACL) interfaces, require data types that are non-persistent.

It is amazing that almost two years after this issue was raised, after many hours of thought and great debates over countless beers, we reach the final hours of ballot resolution and still have not reached consensus. The resolution of the day is:

- MAC and IL are going to be persistent opaque,
- Audit, Discretionary Access Control (DAC), and PRIV are going to be non-persistent opaque

Still no power supply and it's the shipper's fault according to customer service.

Thursday:

The next controversial issue that was raised has its origins even further back in the history of POSIX.6. The discussions go all the way back to /usr/group meetings! This is the ACL feature called the mask. The mask was introduced as a mechanism to:

- map UNIX mode bits into an ACL,

- map `chmod()` calls to manipulations of an ACL

- provide backwards compatibility with the current uses of the mode word.

In order to achieve maximum compatibility (but not 100%), the ACL algorithm became incredibly complex as ACL entries became subject to restrictions and manipulations incurred by the mask. The algorithm became esoteric, to the point where this reviewer believes that no one without a PhD in computer security will be able to understand it.

In order to simplify the algorithm, the mask has been deleted. Now, mode bits are converted to ACL entries, and `chmod()` only affects the UNIX mode bits. A POSIX configuration option allows the application to select whether or not to receive an error when `chmod()` is executed on a file that has an ACL on.

No power supply – but it will be there tomorrow for sure for sure.

Friday:

Most groups are 80-95% complete on their pass through the objections and comments. ACLs, who had a few extra to begin with, still have the furthest to go. The committee would like to go out for re-ballot or re-distribution at the end of the Utrecht meeting.

UPS finally delivers Roland's new power supply just in time to pack up his laptop and get absolutely no use out of it whatsoever.

Report on POSIX.17 – Directory Services API

Mark Hazzard <markh@rsvl.unisys.com> reports on the meeting in Chicago July 13-17, 1992:

Summary

Draft 3.0 of POSIX.17 completed the first round of IEEE balloting in May. We met primarily as a ballot resolution team in Chicago, resolving 98% of all outstanding comments and objections. Since the Chicago meeting, we have finished Draft 3.0 ballot resolution, and published and re-circulated Draft 4.0 for ballot. We plan to get the ballot results in time to resolve comments at the Utrecht meeting in October. From there we plan to submit the balloted specification to the IEEE for final approval, publication, and forwarding to ISO for fast tracking (i.e. direct ISO ballot).

Our Project Authorization Request (PAR) has been split/recast into 4 separate PARs to:

1. separate the Directory Services API work (which is almost finished) from the POSIX name space issue which hasn't received much attention, and
2. separate the actual document into a format aligned with ISO expectations.

Introduction

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

XDS is an object oriented interface and requires a companion specification (XOM) for object management. XOM is a stand-alone specification with general applicability beyond the API to directory services. It will be used by IEEE P1224.1 (X.400 API) and possibly other POSIX groups, and is being standardized by POSIX/TCOS as P1224. A draft of P1224 is already in ballot.

Status

POSIX.17 was reviewed by the Project Management Committee for the Chicago meeting without problem.

Draft 3.0 of POSIX.17, which included all test methods and its Language Independent Specification (LIS), completed IEEE ballot prior to the Chicago. The group spent a majority of the meeting processing the results of that ballot. Over 200 comments/objections were processed, with all but four tentatively resolved. Actions were assigned to resolve the remaining four. Our technical editor did an incredible job in producing Draft 4.0 in time for a recirculation ballot, which closed October 5th.

POSIX.17 was one of three TCOS-SS projects recommended for fast track ballot to ISO during a special ad hoc meeting of the US TAG to JTC1. [Ed. - ISO is responsible for developing and approving of international standards. TCOS-SS (Technical Committee on Operating Systems - Standards Subcommittee) is the IEEE committee responsible for developing operating system standards, e.g. POSIX. Documents developed by the IEEE can be forwarded by an ANSI (hence U.S.) Technical Advisory Group (TAG) to the

Joint Technical Committee (JTC1) of ISO and the International Electrotechnical Committee (IEC) for consideration as ISO standards.]

In order to accommodate the ISO format, POSIX.17 needed to be split into four separate parts (documents). To that end, four PARs were submitted to the IEEE which requires a PAR for every document. This in effect revises our current work to reflect the ISO format requirements. These have been reviewed and accepted by the IEEE Review Committee (RevCom) and assigned the following project numbers under the general heading of "OSI APIs".

P1224.2 – Directory Services API – Language Independent specification

P1326.2 – Test Methods for P1224.2

P1327.2 – C Language Binding for P1224.2

P1328.2 – Test Methods for P1327.2

My understanding is that when P1224.2 and P1327.2 are approved by the IEEE, they will be proposed to ISO as Draft International Standards (DIS).

In Closing ...

The group is meeting in Utrecht in October, where we plan to process the results of our September recirculation ballot. If all goes to plan, we will submit our specification to the IEEE for acceptance as a standard before year's end. Based on this schedule, I would expect to see it published by the IEEE in the first half of 1993.

Report on POSIX Distributed Security Study Group

David Rogers reports on the July 13-17, 1992 meeting in Chicago, IL:

Background

In October 1991, as a result of the activities of the informal liaison group between the Security, System Administration, and Distributed Services working groups, a draft project authorization request PAR was circulated for discussion. This draft PAR proposed a working group to define a POSIX.0 model for security in a distributed system, and the definition of security interfaces in a distributed environment.

From the discussions on the draft PAR, a study group was proposed to investigate the subject more thoroughly and, if appropriate, produce a more clearly defined PAR.

As a consequence, a BOF was held at the January 1992 POSIX meeting and the formation of the Distributed Security Study Group under the auspices of the Distributed Services Steering Committee was approved by the POSIX Sponsor Executive Committee (SEC).

Current Status

Two full meetings of the study group have been held, with a core of about 10 people. The initial emphasis has been to define a framework or model, based on the POSIX.0 model, in which to place the required security functionality into context and to identify suitable APIs.

The first meeting entertained a set of presentations on the OSF's Distributed Computing Environment (DCE), the ECMA SESAME project, and Secureware's MAXSIX. We wanted to review input on existing or emerging practice and architectures.

Following this, an abstract approach to develop the model was tried, based upon the POSIX.0 model. One overheard comment was that "They don't even know what planet they are headed for." However, the meeting did agree to a suggestion that the ECMA Security Framework (described in ECMA TR/46) should be used as a starting point. Additionally the GSSAPI was identified as a potential candidate for a base imple-

mentation. Accordingly, liaison was initiated with the Internet Engineering Task force (IETF) on the status of the Generic Security Service API (GSSAPI) and potential need for extensions to it.

An initial draft paper mapping the ECMA Security Framework into a POSIX.0 model with POSIX.1 and POSIX.6 was produced between the April and July meetings. The ideas in this were reviewed during the July meeting, together with the overall structure and content of the proposed report to be produced by the study group. The POSIX Security Framework document is being further developed prior to the October meeting in Utrecht.

Liaison with Other Organizations

Shortly after the April meeting it was brought to the attention of the chair of the study group that X/Open were also proposing work of a similar nature and scope, including approaching the IETF regarding GSSAPI. (In fact the IETF working group chair received approaches from POSIX and X/Open within 2 hours of each other!) Several meetings have been held between members of the study group and X/Open representatives to ensure that the respective groups coordinate their activities and do not unnecessarily diverge or conflict.

Calendar of Events

1993

Jan 11-15 IEEE 1003, New Orleans, LA
25-29 * USENIX, San Diego, CA

Feb 22-24 Sun Open Sys. Expo, Chicago, IL

Mar 8 -12 Interop, Washington, D.C.
15-19 UniForum, San Francisco, CA
29-

Apr 1 * UNIX Applications Development
Toronto, Canada
19-21 * Mach III, Santa Fe, NM
19-23 IEEE 1003

May 3 - 7 EurOpen, Seville, Spain
20-22 UniForum NZ, New Zealand

Jun 5-11 DECUS, Atlanta, GA
21-25 * USENIX, Cincinnati, OH

Jul 12-16 IEEE 1003

Aug 1 ACM Siggraph, Anaheim, CA
2 - 3 * Mobile & Location Independent
Computing, Cambridge, MA
23-27 Interop, San Francisco, CA
INET '93, San Francisco, CA

Sept 20-22 *Micro-Kernels II, San Deigo, CA
23-24 * SEDMS IV, San Diego, CA

Oct 4-6 * UNIX Security Symposium IV
18-22 IEEE 1003

Nov 1- 5 * LISA VII

Autumn EurOpen/UniForum
Utrecht, Netherlands

Dec 4-10 DECUS, San Francisco, CA

1994

Jan 17-21 * USENIX, San Francisco, CA

Mar 23-25 UniForum, San Francisco, CA

Apr 18-22 EurOpen

May 7-13 DECUS, New Orleans, LA

Jun 6-10 * USENIX, Boston, MA

Sep 12-16 Interop, San Francisco, CA

Autumn EurOpen/UniForum
Utrecht, Netherlands

Nov 12-18 DECUS, Anaheim, CA

1995

Jan 16-20 * USENIX, New Orleans, LA

Feb 21-23 UniForum, Dallas, TX

May 1- 5 EurOpen
13-19 DECUS, New Orleans, LA

Jun 19-22 * USENIX, San Francisco, CA

Nov 2- 8 DECUS, San Francisco, CA

1996

Jan 22-26 * USENIX, San Diego, CA

Mar 12-14 UniForum, San Francisco, CA

May 18-24 DECUS, Orlando, FL

Nov 16-22 DECUS, Anaheim, CA

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

* = events sponsored by the USENIX Association.

ACE: Advanced Computing Environments
ACM: Association for Computing Machinery
AFUU: Association Francaise des Utilisateurs d'UNIX
AUUG: Australian UNIX Users Group

DECUS: Digital Equipment ComputerUsers Society
EurOpen: European Forum for Open Systems
GUUG: German UNIX Systems User Group
IEEE: Institute of Electrical and Electronics Engineers

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

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

USLE Column

Gillian MoggSmith
Marketing Manager
USLE
London
United Kingdom

E-mail gill@uel.uucp

Gill is Marketing Manager at UNIX Systems Laboratories Europe.

for futher information on this column, please contact Gill on gill@uel.uucp, Telephone +44 81 567 7711.

The European Open Market and Open Systems – much more than a marriage of convenience

The approach of 1992 and the Open European Market is both a challenge and an opportunity for the IT industry. The various European countries have traditionally served as catchment areas for their own proprietary OEM giants. Now the old, comfortable, proprietary way of life is under threat from two directions. The vanishing of internal borders will add considerable impetus to the competitive process between OEMs. While at the same time, the irresistible growth of open systems is already forcing a radical rethink upon all the players in the industry. The European computer industry has recognised that its future prosperity depends upon the adoption of common standards of computing across Europe.

An inevitable consequence of the growth of open systems is increased competition. By definition open systems encourages multi-sourcing. Because it relies on publicly defined standards and interfaces across a range of key areas, from operating system components to communications protocols, open systems speeds up the whole process of bringing new technology to the market. Instead of proprietary companies being in control of the pace of change and managing the introduction of new technology to the market, open systems acts as an enabler of change. The result has been continual improvements in price/performance ratios of hardware.

Already, this has had a dramatic impact on margins. Companies of all descriptions have had to cut down on their overheads. One of the most effective ways of doing this is to cut out unnecessary expense at the

product development stage. IT companies now have to make very careful 'build versus buy' decisions. If a particular piece of technology already has a proven track record, or looks promising, then it is frequently going to be more cost effective for one or more players in the market to team up with the provider of that technology rather than to try and compete with it by engineering an alternative solution.

What this means in practice is that while open systems has fuelled competition, it has also provided something of a solution to the pressures on margins. The result is a growth in the number of 'partnering' announcements made by companies who would normally be considered rivals. Some of these announcements are very short term and tactical, others have a more long term dimension to them. One recent example here is the announcement by UNIX System Laboratories (USL) that it had reached an agreement with the Open Software Foundation to make OSF's Distributed Computing Environment technology available for SVR4. Another example is USL's joint venture with Chorus systemes concerning Chorus' Microkernel technology.

The European market needs to be viewed trans-nationally across Europe. This type of market openness can only be achieved through the establishment of publicly agreed standards and published common interfaces. It is highly unlikely that a totally new, open operating platform will arise in the time scales available, and for this reason, the European computer industry is already adopting UNIX SVR4 as the de facto standard operating system.

Proof of this came with the recent announcement that a number of key players in the European computer industry are working as a consortium on a project named Ouverture - backed by the Commission of the European Community to the tune of ECU 14 million - to promote a unified Open Systems policy. The aim of the project is to utilise the best of European and US technology to develop the potential of the UNIX SVR4 operating system in the microkernel arena, swiftly and cost-effectively, by avoiding duplication of research across Europe.

A further aspect of the development of the European computer market is the perceived need for a single platform running from the desktop to the mainframe. Again, at present, only UNIX SVR4.2 is in a position to provide a pan-European open operating system, capable of running across the full spectrum of machines. It is

worth pointing out that some non trivial technical engineering issues needed to be resolved in order to attain this goal.

To be capable of running on PCs as well as on larger boxes, the operating system had to be repackaged with

all the basic functions put into the 'foundation set'. This produced the UNIX SVR4.2 operating system, the same basic module of which runs on everything from the mainframe to the PC. Other modules are available as 'add ons' to address such requirements as multi-user, server connectivity, systems development and so on.

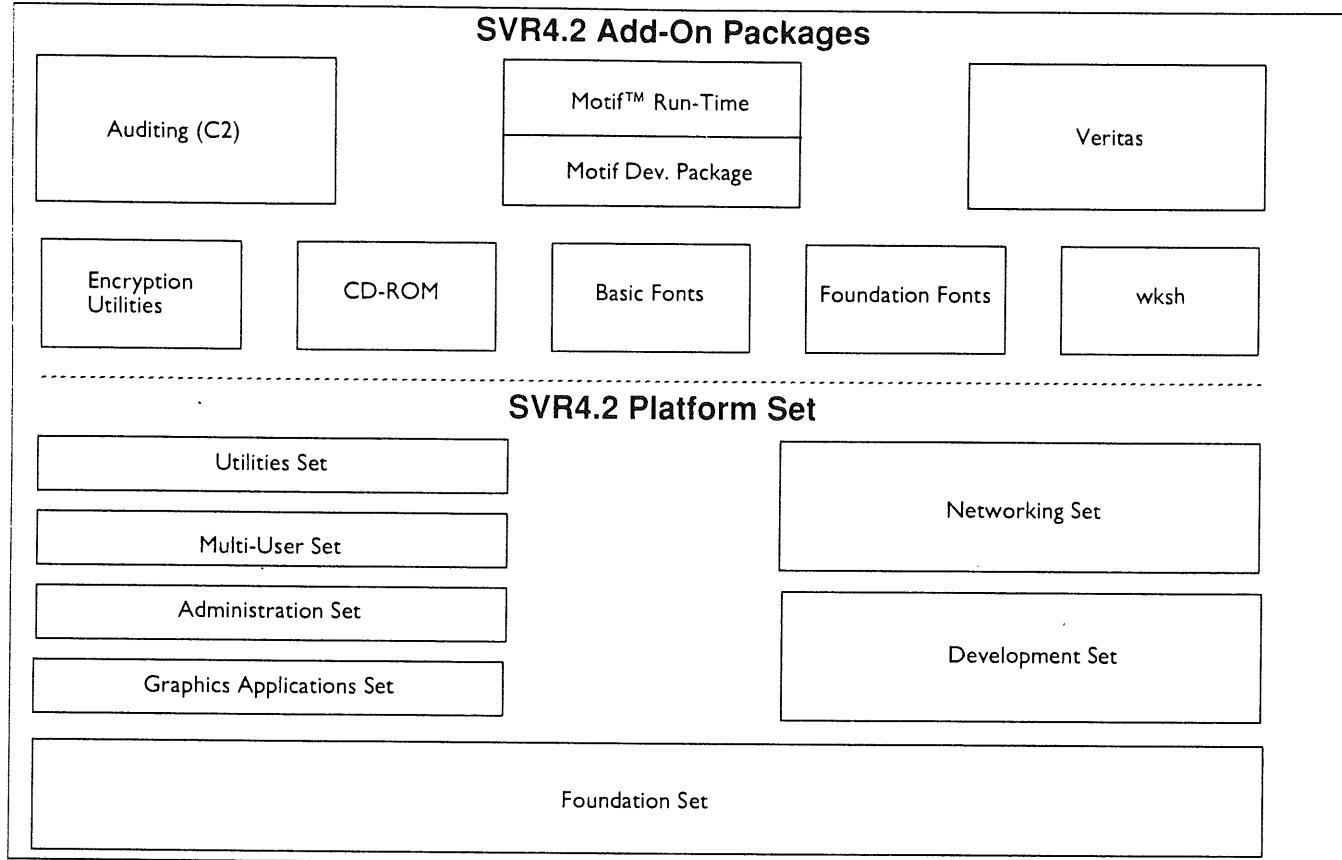
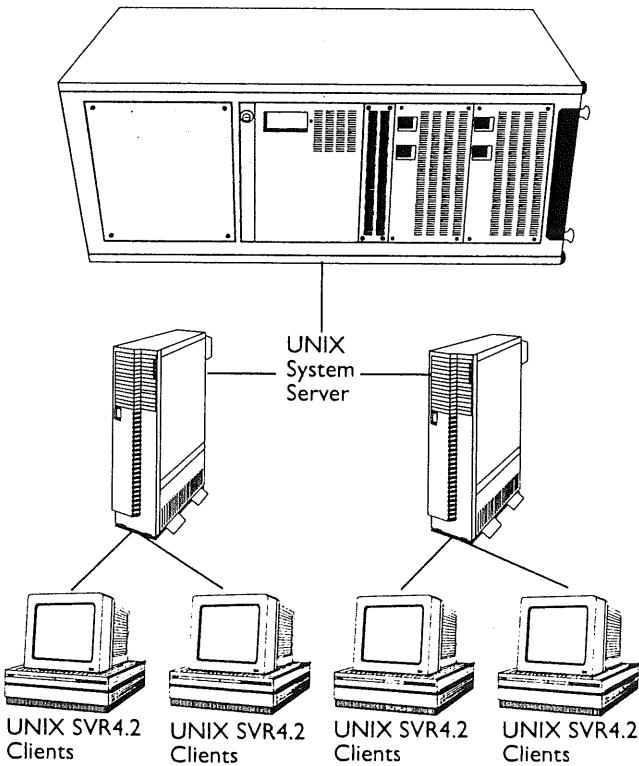


Figure 1: SVR4.2 Components

UNIX SVR4.2 is designed to address the current, common requirement across Europe for 'rightsizing'. This is a concept which means giving end users systems which are capable of precisely satisfying the processing requirements placed on them. This means 'downsizing' departmental applications from the corporate mainframe to UNIX boxes and it means giving 'power users' on the desktop the ability to 'upscale' their PCs so that they can run the more demanding applications and access information on an organisation-wide basis.

The current downsizing/rightsizing wave has driven the explosive growth of UNIX system implementations within high-end and mid-range server applications. UNIX provides the functionality required in a commercial environment, coupled with availability on a wide range of hardware platforms that includes every major hardware vendor. Today, information executives, from companies of all sizes, can benefit from the wide variety of hardware choices, client/server application support and connectivity provided by UNIX. All this is available within an environment of application portability and scalability, assuring protection of investment and easy migration paths for the future.



Expandable and scalable, UNIX SVR4.2's modular architecture is the natural bridge for linking desktops and LANs to mid-range servers and mainframes through a common UNIX SVR4 application and data environment. Imbued with the rich multi-user and multi-tasking heritage of UNIX SVR4, UNIX SVR4.2 answers the modern needs of departmental and inter-departmental computing.

Another key factor in favour of UNIX SVR4.2 is that USL has addressed the issue of volume shipment which is critical to any operating system aiming to support a Europe-wide open systems computing base.

Accordingly USL, as was announced some months ago, has formed a joint venture company with Novell, the world's leading network company. The new company, called Univel, will integrate UNIX SVR4.2 with Novell NetWare. This integrated product will then be distributed and supported through the established Novell reseller channel.

In the past, USL has produced source product and others have gone on to prepare and ship binary versions of the UNIX operating system from that source. With the earlier version, SVR4.0 the gap between the appearance of the source product and a binary product was roughly a year. It is now obvious that in the fast moving world of desktop PC technology, such a time gap is not feasible. A good deal of work has been done to move UNIX SVR4.2 closer to a binary OS product. As a result, the first binary of UNIX SVR4.2 is likely to be available within three months of the launch of UNIX SVR4.2.

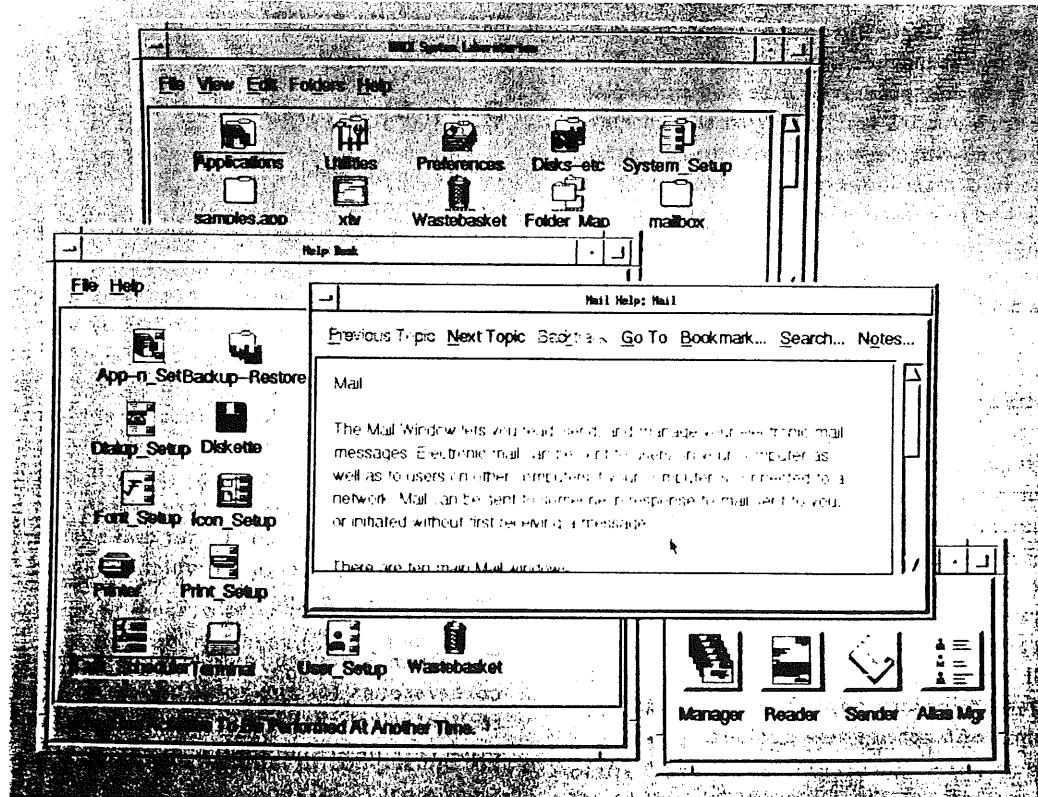
The Univel agreement means that USL is able to deploy a two-pronged approach to the channels problem,

drawing upon the established strengths of its traditional links with OEMs as well as on Novell's proven distribution capability. In other words, in addition to providing the new binary product to Novell to sell through its existing dealer channels, USL in its turn, will continue to be responsible for selling source product to OEMs as well as meeting their requirements for the binary OS provided by Univel.

As a final point, it is worth noting that the two channel approach, through the OEMs and through Novell's dealer channels, is very likely to be mutually reinforcing. As Novell succeeds in selling more and more of its UNIX SVR4.2 binary product across Europe, pressure from end users will grow for the OEMs to ensure that they too, provide UNIX SVR4.2 binary compatible operating systems.

Because UNIX SVR4.2 will come out under a variety of names - each OEM will tend to use their own brand name - USL has established a compliance branding program to guarantee UNIX SVR4.2 compatibility. USL will implement the program, providing the industry with compatibility requirements/specifications for conformance, based on UNIX SVR4.2. All vendors who meet these standards, via verification by USL, will receive the brand. End-users may purchase branded UNIX system products with assurances that the products will work together.

USL, Univel and the OEMs will be running an aggressive campaign aimed at attracting independent software vendors to ensure that UNIX SVR4.2 becomes the standard open systems advanced OS across Europe - and indeed world-wide - in the years to come.



OSF Column

Mark Laureys
Communication Manager European Operations
Open Software Foundation
15 Avenue des Pléiades
1200 Brussels
Belgium

Mark Laureys joined OSF's European Operations based in Brussels in June 1991. He is responsible for the European Communications and his main task is to provide members of the European press and consultants with information on OSF.

He was previously working in a similar function for a network integrating company called Telindus Networks, part of the Telinfo Group.

For further information on this column, please contact Mark Laureys: telephone +32 2 772 88 88

Europe, on the Brink of a Vast New Market

Today, we stand before perhaps the biggest single opportunity in business history. Advanced technology having brought us firmly into the computer age, information has replaced energy as the world's single most valuable resource. Thus capacity to harness that priceless resource will undoubtedly determine tomorrow's business.

At the same time, the Single European Act is about to open the world largest market. A market of over 320 million people- a market greater in size and potential than even the enormous US market. With the barriers down, the fastest growing segment of this huge market will certainly be information technology. Indeed, demand for data and information exchange, storage and processing is expected to grow with as much as 40% per year. Leaders in this market will by definition have access to the world market.

New Business Challenges

The stakes are high. Removal of national barriers in this new information area will not only open a large untapped market, it will also unleash competition from all over the globe. With powerful companies competing for a share in a much bigger pie. To succeed organisations must have the right experience, technological capability, products and the most important of all highly skilled people. They also must be flexible and forward-looking enough to meet rapidly changing demands of a totally new business environment.

The globalisation of the services industry for instance is bringing with it a far-reaching change in the way of doing business. Borders are disappearing, as much between different production specialities as between countries. Information systems and networks ensure that everyone can communicate world-wide. No one any longer accepts that his data cannot arrive directly onto his business partner or prospect's desk, nor is acceptable that applications can't run remotely on whatever mainframe, workstation or PC on the network.

More interaction and co-operation between larger and smaller companies is another result of the the single European market. The larger the company, the larger its opportunities to invest in future-oriented systems. Small companies are frequently forced to seek salvation in certain restructuring solutions. Larger companies are realising that the key to success lies in anticipating the needs of European customers. This requires an obvious spread of responsibility and a systematic manning of marketing networks. Smaller companies who are able to form useful alliances can gain major opportunities this way. Those alliances and co-operative business networks all over Europe will surely benefit from an open systems environment.

Open Systems for an Open Market

The IT environment is not much different from the socio-economic environment as described above. As diversified and heterogeneous it is looking for more openness and interoperability. Millions of different PC's and workstations have entered most companies through the back door. With hardly a thought to co-ordination. This technological tidal wave now has to be correctly channelled. How do we avoid being swept away under mountains of incoherent data? How can these intelligent machines be usefully applied to the company's business objectives? Will everybody have to throw away their proprietary systems in order to be open? Every European IT user and business strategist is faced with major choices which will bind his company for the years to come.

The Open Software foundation believes that the cornerstones for the solution to those business and technological challenges are openness and more especially interoperability.

It should be however understood that openness is not a function of which operating system is used. An open environment is one that employs a standard set of interfaces for programming, for communications, for networking, for system management, and for user look and feel. The open systems environment must supply common ground for incompatible hardware, operating systems, software, and the people using them to transparently interact. It must supply supervisory services to all the resources throughout a computer network.

Standards are an important part of the migration path to open systems. With the unification of the EC market and the emergence end 1992 of a new economic territory, there will be an acceleration of the standardisation process. One will of course need more than ever to follow industry (*de facto*) standards or de jure standards, when they exist, in order to tie up the existing heterogeneous systems. But as IDC reported before in a review of the Unix Software Marketplace in Western Europe "The standardisation process takes time (...) and is far from complete". It is OSFs charter to help the information processing industry specify and standardise the interface definitions of an open systems environment, as well as to provide reference implementations of software adhering to those specifications.

OSF proved being a front runner in this process by launching successfully enabling technologies which are now accepted by the industry at large and will serve as reference platforms for the future standards. The graphical user interface MOTIF is one example and DCE (Distributed Computing Environment) will follow now that the major players in the industry have announced their plans to integrate DCE in their technology architecture or in their product development plans.

Interoperability - The Next Step

To the previously asked question "do we have to throw away our proprietary systems in order to be open?" OSF's answer is NO. Complexity is more than ever a reality in the corporate world. So is the need to preserve investments, in hardware and software, while enabling corporations to add technologies they need without jeopardising what they currently have.

Interoperability among diverse systems is the key for users to migrate smoothly from a proprietary to an open computing environment and to provide them with the freedom to choose the appropriate computing solution for the job at hand. To do so, they need to mix and match hardware and software from various vendors, easily access and protect the data stored in their networks, and apply a common management scheme to an array of diverse systems. OSF's Distributed Computing Environment (DCE) and Distributed Management Environment (DME) technologies address this interoperability need.

Drawing on the client/server model of computing, the Distributed Computing Environment (DCE) from the Open Software Foundation (OSF) allows companies to

transcend the limitations that geography traditionally has imposed on their business. The DCE lets information flow from wherever it is stored in a network to wherever it is needed. As a result, users can take advantage of applications and data scattered throughout the network. Accessing files and information from a remote branch office becomes as easy as retrieving it from across the room. An early adopter of the technology has been the European Commission. The EC has announced early last year that it plans to use DCE for a new generation of distributed applications as part of their multivendor computing strategy that supports about 10,000 end users.

The OSF DME, currently under development by OSF, draws on services provided in the DCE as well as object-oriented technology to manage stand-alone systems from multiple vendors as well as the growing number of distributed systems in use. The goal of the DME is to simplify the management of heterogeneous computing and network environments.

Conclusion

Interoperability is the answer to managing the complexity of today's and tomorrow's computing systems in Europe. The benefits of interoperability are far-reaching, and include not only operating system technology, but other enabling technologies, such as those discussed above.

We at OSF believe the open systems horizon extends far beyond operating systems. OSF enabling technologies provide a base which protects user investment in hardware, software, training and applications while allowing innovation to flourish, resulting in products which will enrich and bring diversity to the industry.

The effects of interoperability technologies on computer users will be profound. Users will be freed to select hardware, operating systems and software applications that best meet their current needs, and anticipate their future needs.

OSF NEWS

OSF Delivers OSF/I Release 1.1

Recent Development Boosts Performance, Internationalization, and Robustness Brussels, June 25th : The Open Software Foundation today announced the general availability of Release 1.1 of the OSF/I Operating System. This is the second major release of the operating system, which was first introduced in October of 1990.

"This release of OSF/I demonstrates OSF's clear commitment to the ongoing development and adoption of the OSF/I operating system in the marketplace," said David Tory, OSF's CEO. "OSF/I Release 1.1 provides a

robust and compatible platform that end users are demanding for their open systems environments."

OSF/I Release 1.1 includes:

- Enhanced internationalization - Enables application developers to reach world-wide markets without rewriting their application code to support different languages. Extended UNIX Codes (EUC) provide support for ideographic languages such as Korean, Chinese and Japanese. Additional work includes conformance to the X/Open XPG4 draft specification for wide-character interfaces.
- Scalability enhancements - Extend the reach of OSF/I systems to PC-class computers. OSF/I Release 1.1 runs on systems with as little as 4MB of memory, taking advantage of the OSF/I dynamic configuration capabilities to load and unload major subsystems, such as NFS, TCP/IP, or a System V file system, at runtime.
- SVID 3 compatibility - Ensures that applications written for System V Release 4 will be portable to OSF/I. Release 1.1 provides SVID 3 compatible STREAMS implementation. (SVID 3 is the specification for System V Release 4.)
- Performance enhancements - Boost responsiveness of the system, especially in the areas of virtual memory, NFS, and the loader.
- Standards compliance - Ensures that OSF/I systems evolve concurrently with relevant industry, national, and international standards. OSF/I is fully compliant with POSIX 1003.1 - 1990, ANSI C, and XPG3, among others. Release 1.1 also includes work based on POSIX drafts of 1003.2 for commands and utilities, 1003.4a for Pthreads and 1003.6 for security.

This release of OSF/I continues to provide the symmetric multiprocessing capability and security features required by the commercial processing market.

Parallel development continues in the OSF Research Institute on the OSF/I Microkernel technology, now available in snapshot form to OSF/I licensees.

The Release 1.1 tape includes three reference implementations for the following architectures: Intel 302 (80386 based), Digital DECstation 3100 (MIPS based); and the Encore Multimax (National Semiconductor based).

OSF/I Release 1.1 is priced at \$85,000 for a source license with full distribution rights; \$60,000 for a source license only. Existing source licensees may upgrade from Release 1.0 to Release 1.1 for \$25,000. Licensees who hold full support contracts will receive the upgrade without charge as part of their support services. University site licenses are available for \$5,000.

Binary royalty fees remain unchanged at \$65 per copy, with volume discounts available. The price to upgrade non-distribution source licenses to full redistribution rights remains unchanged at \$35,000. For further information, contact OSF Direct at +1-617-621-8700.

Open Software Foundation Unveils DME Roll out Plan

Brussels, 25th June : The Open Software Foundation made public the development and release schedule for its Distributed Management Environment (DME). OSF also announced that the first Snapshot release of the DME technology is now available to OSF members. Under OSF's Snapshot program, members are offered early access to source code throughout the development cycle.

"The end user community has been demanding a solution to multivendor distributed management," said Garry Baer, a technology manager for DME. "Responding to this market need, OSF has put in place an aggressive, phased roll out plan designed to make DME components available as rapidly as possible to facilitate broad early adoption."

DME provides an effective solution for systems administrators, who need efficient and reliable management services to keep their distributed computing environments operating smoothly. Application developers will benefit from the rich set of tools and services the DME framework provides for writing management applications. End users will benefit from knowing they can work effectively and efficiently, free from concern about how the system is running.

DME will roll out in a modular five-step process. The Distributed Services Release, targeted for general availability in the first half of 1993, will provide key distributed management services to the OSF Distributed Computing Environment (DCE) technology.

In the second half of 1993, the DME Framework Release will provide the integrated DME framework, development tools, and selected framework applications.

The OSF integration model is a collaborative effort between OSF and its technology suppliers. Sub-integration teams for discrete components are made up of technology suppliers and OSF. The final integration and testing will be done at OSF headquarters in Cambridge, USA.

Based on advanced object-oriented technology, DME is the first vendor-neutral platform for managing networks and distributed systems from different vendors. It is compatible with existing distributed systems while providing a means of migrating to newer technologies. In this way, DME ensures that organisations can capitalise on their investments in hardware and software.

AUUG MANAGEMENT COMMITTEE

SUMMARY OF MINUTES OF MEETING 04 December 1992

Present: Frank Crawford, Rolf Jester, Chris Maltby, Phil McCrea, Michael Paddon, Greg Rose, Peter Wishart, and Liz Fraumann.

Apologies from, Glenn Huxtable, Piers Lauder, John O'Brien, and Michael Tuke.

1. Publishing Minutes in AUUGN

It was agreed that the Secretary would produce a summary of the important parts of the minutes in AUUGN rather than publishing the whole minutes. This means that a summary of the minutes could be published before the minutes were ratified by the subsequent meeting and they would consume less space in AUUGN.

2. Finances

We have received \$10,480 from ACMS for the AUUG '92 profit. \$20,000 has moved from the cash management account to cover costs. Over the last 6 months we have sustained approximately a \$2,000 loss. The bank balance is \$14,220 and \$100,000 in the cash management account.

3. New Assistant Returning Officer

Greg Bond has been appointed by the committee to the vacant position of Assistant Returning Officer.

4. AUUG Lapel Pins

AUUG lapel pins are being prepared in time to be handed out at the summer conferences.

5. AFUU

Peter Elford has been selected to present his paper at the AFUU (French Unix Users Group) conference and Michael Tuke will present a tutorial. A joint AUUG/AFUU video/audio announcement is scheduled for 24th March Paris time.

6. Election Procedures

New election procedures have been produced, they will be published in the next AUUGN.

7. UniForum Technology Guides

As part of the affiliate member benefit UniForum provides a discount on their technology guides. AUUG can bulk order and pass savings to entire organisation. It has been decided to

- (1) Advertise them in AUUGN
- (2) Purchase enough and distribute to the institutional members (a benefit).
- (3) Modify the membership form to reflect a space for optional UniForum affiliate membership.

8. DataPro Round Table

DataPro and AUUG hosted a round table discussion with 5 leading MIS directors from user organisations. All were using different hardware and had not met previously. As a user group it was felt that we should be doing more of this. It could also be used to facilitate an institutional membership drive. It was recommended that we continue such events and consider hosting 2 round tables in '93. One in Brisbane and Melbourne.

9. Summer Conferences

Earlier the committee had decided that one summer conference each year should receive additional support to provide a larger conference in that area. This year the conference will be Perth. The committee felt the additional support would come in the form of airfare (from the east coast) and accommodations. The support would be a maximum of \$2,500.

10. Local Chapters

The tag line on the stationery will be modified to read "UNIX and Open Systems Users". It was also agreed to modify the screen colour to black rather than the blue to facilitate 2 colour printing with text. Chapters will be given a supply of stationery for their use.

It was decided a 6 month financial reserve would be allocated to each chapter in line with membership renewals. The monies will be based on the "official" membership numbers generated. As of 31 January 1993, 20% of the membership fee will be distributed to each chapter. It would help facilitate things if a bank account were already in place. The monies will be distributed to elected officials only.

11. Relationship with ACS

ACS members will attend the AUUG conference at AUUG member discounts and ACS, through their local newsletters, will promote the conference. AUUG will receive free access to the PCP accreditation scheme for the conference.

12. AUUG '93

The dinner event will be held at the Power House Museum. Costs for dinner (w/o beverage) is \$25.00 pp.

13. Other Business

Membership Drive - it was suggested we have a membership drive and work it in conjunction with the summer conferences, round tables, etc.

New look for AUUGN - as one of the single largest expenditure for the organisation, it was decided to pro-actively pursue advertising.

14. Next Meeting

18th February 1993 at Softway.

AUUG Membership Categories

Once again a reminder for all “members” of AUUG to check that you are, in fact, a member, and that you still will be for the next two months.

There are 4 membership types, plus a newsletter subscription, any of which might be just right for you.

The membership categories are:

- Institutional Member
- Ordinary Member
- Student Member
- Honorary Life Member

Institutional memberships are primarily intended for university departments, companies, etc. This is a voting membership (one vote), which receives two copies of the newsletter. Institutional members can also delegate 2 representatives to attend AUUG meetings at members rates. AUUG is also keeping track of the licence status of institutional members. If, at some future date, we are able to offer a software tape distribution service, this would be available only to institutional members, whose relevant licences can be verified.

If your institution is not an institutional member, isn't it about time it became one?

Ordinary memberships are for individuals. This is also a voting membership (one vote), which receives a single copy of the newsletter. A primary difference from Institutional Membership is that the benefits of Ordinary Membership apply to the named member only. That is, only the member can obtain discounts an attendance at AUUG meetings, etc. Sending a representative isn't permitted.

Are you an AUUG member?

Student Memberships are for full time students at recognised academic institutions. This is a non voting membership which receives a single copy of the newsletter. Otherwise the benefits are as for Ordinary Members.

Honorary Life Membership is not a membership you can apply for, you must be elected to it. What's more, you must have been a member for at least 5 years before being elected.

It's also possible to subscribe to the newsletter without being an AUUG member. This saves you nothing financially, that is, the subscription price is greater than the membership dues. However, it might be appropriate for libraries, etc, which simply want copies of AUUGN to help fill their shelves, and have no actual interest in the contents, or the association.

Subscriptions are also available to members who have a need for more copies of AUUGN than their membership provides.

To find out your membership type, examine your membership card or the mailing label of this AUUGN. Both of these contain information about your current membership status. The first letter is your membership type code, M for regular members, S for students, and I for institutions, or R for newsletter subscription. Membership falls due in January or July, as appropriate. You will be invoiced prior to the expiry of your membership.

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

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

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

AUUG Incorporated

Application for Institutional Membership

AUUG Inc.

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

AUUG Membership Secretary
PO Box 366
Kensington NSW 2033
Australia

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

This form is valid only until 31st May, 1993

..... does hereby apply for

<input type="checkbox"/> New/Renewal*	Institutional Membership of AUUG	\$325.00
<input type="checkbox"/> International Surface Mail		\$ 40.00
<input type="checkbox"/> International Air Mail		\$120.00

Total remitted

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

* Delete one.

I/We agree that this membership will be subject to the rules and by-laws of the AUUG as in force from time to time, and that this membership will run for 12 consecutive months and becomes renewable on the following January or July, as appropriate.

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

Date: / / Signed: _____
Title: _____

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

For our mailing database - please type or print clearly:

Administrative contact, and formal representative:

Name: Phone: (bh)

Address: (ah)

Net Address:

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

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

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

Name on card: _____ Signed: _____

Office use only: _____ Please complete the other side.

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

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

Who: _____ Member# _____

Please send newsletters to the following addresses:

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

..... Net Address:

.....

.....

.....

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

..... Net Address:

.....

.....

.....

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

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

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

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

- | | |
|--|--|
| <input type="checkbox"/> System V.3 source | <input type="checkbox"/> System V.3 binary |
| <input type="checkbox"/> System V.2 source | <input type="checkbox"/> System V.2 binary |
| <input type="checkbox"/> System V source | <input type="checkbox"/> System V binary |
| <input type="checkbox"/> System III source | <input type="checkbox"/> System III binary |
| <input type="checkbox"/> 4.2 or 4.3 BSD source | |
| <input type="checkbox"/> 4.1 BSD source | |
| <input type="checkbox"/> V7 source | |
| <input type="checkbox"/> Other (<i>Indicate which</i>) | |

AUUG Incorporated

Application for Ordinary, or Student, Membership

AUUG Inc.

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

AUUG Membership Secretary
PO Box 366
Kensington NSW 2033
Australia

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

This form is valid only until 31st May, 1993

I, do hereby apply for

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

Total remitted

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

* Delete one.

I agree that this membership will be subject to the rules and by-laws of the AUUG as in force from time to time, and that this membership will run for 12 consecutive months and becomes renewable on the following January or July, as appropriate.

Date: ____ / ____ Signed: _____

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

For our mailing database - please type or print clearly:

Name: Phone: (bh)

Address: (ah)

Net Address:

.....
.....
.....

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

Please charge \$____ to my Bankcard Visa Mastercard.

Account number: _____ Expiry date: ____ / ____.

Name on card: _____ Signed: _____

Office use only:

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

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

Who: _____ Member# _____

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

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

Title: _____

Signature: _____

AUUG Incorporated

Application for Newsletter Subscription

AUUG Inc.

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

AUUG Membership Secretary
PO Box 366
Kensington NSW 2033
Australia

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

This form is valid only until 31st May, 1993

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

Name:

Phone: (bh)

Address:

..... (ah)

.....

Net Address:

.....

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

For each copy requested, I enclose:

<input type="checkbox"/> Subscription to AUUGN	\$ 90.00
<input type="checkbox"/> International Surface Mail	\$ 20.00
<input type="checkbox"/> International Air Mail	\$ 60.00

Copies requested (to above address) _____

Total remitted

AUD\$ _____

(cheque, money order, credit card)

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

Please charge \$ _____ to my Bankcard Visa Mastercard.

Account number: _____ Expiry date: ____ / ____.

Name on card: _____ Signed: _____

Office use only:

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

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

Who: _____ Subscr# _____

AUUG

Notification of Change of Address

AUUG Inc.

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

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

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

Old address (or attach a mailing label)

Name: Phone: (bh)

Address: (ah)

.....
.....
.....
.....
.....

Net Address:

New address (leave unaltered details blank)

Name: Phone: (bh)

Address: (ah)

.....
.....
.....
.....

Net Address:

Office use only:

Date: ____ / ____ / ____

Who: _____

Memb# _____