NEWSLETTER

EUUG

European UNIX® systems User Group

Volume 8, No. 3
Autumn 1988

CONTENTS

EUROPEAN UNIX® SYSTEMS USER GROUP NEWSLETTER

Volume 8, Number 3
Autumn 1988

Editorial Team:

Philip Peake — Axis Digital, France
(EUUG Newsletter Editor)

Alain Williams — Parliament Hill Computers ltd.
3 Parliament Court
Parliament Hill
London NW3 2TS
U.K.

Typesetting:

Brian K. Goldstein — The Instruction Set Ltd.
City House
190 City Road
London EC1V 2QH
U.K.

Printed by: Rank Xerox (Copy Bureaux) Ltd.
68-74 Rochester Place
London NW1 9JX
U.K.

Published by: The EUUG
Owles Hall,
Buntingford
Hertfordshire SG9 9PL
U.K.

euug@inset.co.uk (...!mcvax!ukc!inset!euug)

This document may contain information covered by one or more licences, copyrights and non-disclosure agreements. Copying without fee is permitted provided that copies are not made or distributed for commercial advantage and credit to the source is given; abstracting with credit is permitted. All other circulation or reproduction is prohibited without the prior permission of the EUUG.

The editor reserves the right to alter any article submitted for publication. This right has been exercised in previous issues of the newsletter.

UNIX is a registered trademark of AT&T in the USA and other countries.
PDP is a registered trademark of Digital Equipment Corp.
XENIX is a registered trademark of SCO.
PostScript is a registered trademark of Adobe Systems.

ISSN 1011-4211

Editorial

Alain D. D. Williams
addw@phcomp.co.uk

Parliament Hill Computers Ltd
London NW3 2TS
UK
+44 1 435 0200

An Innovation

We have used a two column format in this newsletter to present papers by Jean-Louis Faraut and Phillipe Dax in both the original French and English. We hope that this will inject a more international flavour.

Joy Marino's article was submitted to me in Italian and translated in London. All proof that you don't need to have to write in English to write for the Newsletter.

OSF

Unix has shot into the news with the formation of the Open Software Foundation.

With IBM's entry into the scene something is going to happen. Their Communications Department can't tell me what is happening – I'm not going to be foolish enough to make a guess, but many of you have presented strong opinions on the net.

I am looking for someone to summarise such net discussions in the newsletter for me.

Your chance of fame – any offers?

Future Issues

Articles should be sent to me, preferably electronically at the above address. Please contact me if you want a chat or the newsletter article template.

The next copy dates are:

24 Oct for 1 Dec
23 Jan for 1 March

More Happy Events

Congratulations are due to Colsten Sanger and Kanae Takeuchi. By the time that you read this, they will have been married. We wish them a very happy and fruitful future together.

Typesetting News

Brian K. Goldstein typeset this issue, managing to include Cornelia's submission 10 minutes before the final print run.

Thanks to The Instruction Set for their invaluable assistance in producing this issue on time.

P.S. - Thanks for the champagne, Jr!

ACCES

Jean-Louis Faraut
inria!pastpe!cerisi!faraut

Ingenieur de recherche
Université de Nice

Abstract

ACCES - A new tool to allow management of access rights to commands on multi-user UNIX systems.
ACCES - Un nouvel outil pour faciliter la gestion des droits d'acces aux commandes d' Unix multi-utilisateurs.

Introduction

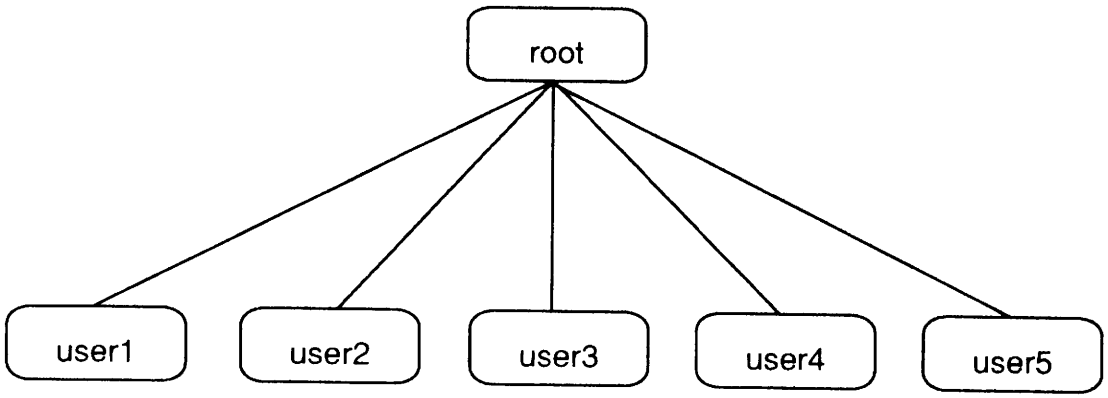
All UNIX systems have a ''super-user'' (normally called ''root''), which can bypass all protections, and modify protections so as to limit the permissions of each user.

In the case of a large machine, with many users this organisation is usually very simple, one super-user, and lots of other users, all at the same level, with basically the same rights. The diagrammatic representation of this schema resembles a ''rake'':

Introduction

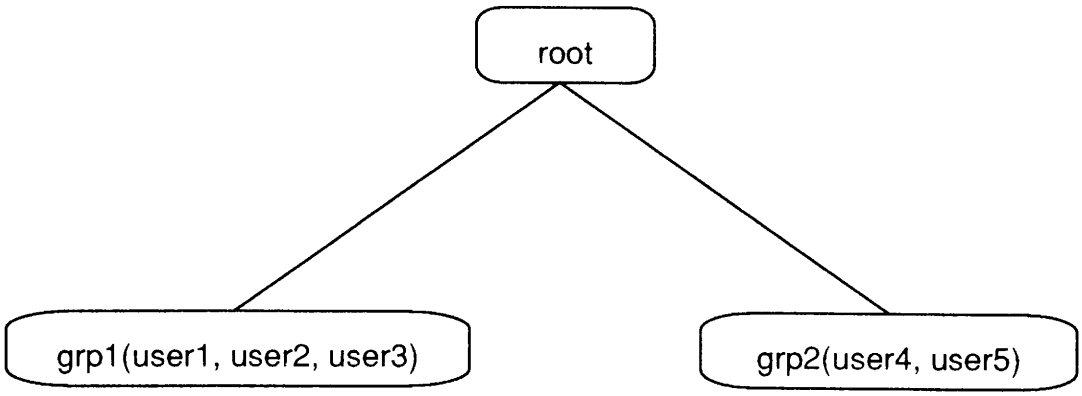
Tout systeme Unix suppose l'existence d'un utilisateur nomme ''root'' qui peut franchir toutes les protections (fichiers, processus, memoire) et les modifier afin de delimiter les droits de chacun.

Dans le cas d'environnement lourd (machine puissante avec utilisateurs nombreux), cette organisation qui est essentiellement du type ''rateau'', c.a.d. avec une tete et tout le monde sur le meme niveau peut ne plus convenir.



The group permissions can be used to further sub-divide this schema giving more selective access permissions:

On peut alors avoir recours a des repartitions par groupe d'utilisateurs a qui l'on accordera selectivement les droits d'acces:



The user passes from one group to the other using either the *newgrp* command (AT&T systems), or by being a member of several groups simultaneously on a BSD-based system.

This organisation, although efficient, presents several inconveniences.

Firstly, there is a confusion about access rights; the same attribute (group membership) controls two resources of a different natures, file access and command access.

Secondly, this system is quite limited in its possibilities; suppose that some group has access to an external network, a second group has access to a laser printer, and that a third has accesses to some other system resource, etc.

The limits of the system are rapidly reached, in that the management these groups becomes a nightmare, and the ease of use for the user non-existent.

Lastly, there is no hierarchical scheme for groups of users. This means that one cannot give a sub-group of a group any extra privileges without giving them to the whole group, or without upsetting the whole organisation by re-defining the whole permissions structure. There is also always a strong possibility that any such re-organisation will introduce errors as the system becomes more complex.

Also, how can one manage privileges that need to be given on a temporary basis?

et on passera d'un groupe a l'autre par la commande *newgrp* (chez ATT) ou en mettant les gens dans plusieurs groupes simultanement (chez Berkeley).

Cette organisation bien que plus efficace presente cependant plusieurs inconvenients.

En premier lieu, elle induit une confusion dans les droits d'acces; en effet, un meme attribut, l'appartenance a un groupe, conditionne l'acces a deux ressources de natures differentes: les fichiers et les commandes.

En second lieu, elle est limitee; supposons que l'on veuille accorder a tel groupe l'acces au reseau exterieur, a tel autre l'acces a l'imprimante laser, a un troisieme encore l'acces aux sources du systeme, etc.

On atteind ainsi rapidement les limites du syteme et la gestion devient tellement lourde que le confort des utilisateurs et leur securite risquent de s'en ressentir.

Enfin, il n'y a pas de hierarchie de groupes d'utilisateurs. Ceci implique qu' on ne peut pas accorder a un sous-ensemble d'un groupe (un "sous-groupe") des privileges particuliers sans les accorder a tout le groupe ou sans bouleverser toute l'organisation au risque de commettre des erreurs et d'introduire des failles pas toujours faciles a deceler dans les protections.

Et comment gerer les privileges que l'on ne veut accorder que temporairement?

Proposed improvement

The proposed solution consists of giving temporary rights (root permissions) to one or more users under strictly controlled conditions.

Note that this possibility already exists in the form of the SUID bit:

chmod u+s UNIX_file

But this only partially solves the problem, since the problem of sub-groups, and temporary privileges still remains. Also, the indiscriminate use of the SUID bit is not particularly compatible with a secure system.

We can thus imagine a single command with the SUID bit in place, which works with a data-base, modifyable only by "root".

This data-base contains a list of all the limited-rights commands, together with a list of users authorised to use the command with root permissions.

A limit-date could also be included to automatically revoke permissions after that date.

Implementation details

The implementation is based upon an Access Control List (ACL), with a defined syntax.

The program is in two modules, 'access.lex' and 'access.yacc', respectively lexical and syntactic analysers for the ACL file. The main program is written in C.

Syntax of the ACL file (in yacc syntax):

```
acl    : block
       | block acl
       ;
```

⇒ the file is a sequence of blocks

```
block : command_line    user_block
      ;
```

⇒ each block is a command and a user block

Amelioration proposee

La solution proposee consiste a donner temporairement les droits de root a un ou plusieurs utilisateurs dans des conditions bien determinees.

Notons que cela est deja possible en positionnant le "USER-bit" d'une commande appartenant a root avec la commande:

chmod u+s fichier_Unix

Mais cela ne repond que partiellement a la question, car le pb des sous-groupes ou des privileges temporaires demeure. D'autre part la multiplication des modes "s" n'est jamais tellement souhaitable pour la securite.

On peut alors imaginer une commande en mode "s" (et une seule) travaillant a partir d'une base de donnees (un simple fichier modifiable seulement par "root").

Ce fichier comprendrait autant d'enregistrements que de commandes Unix concernees; dans chacun d'eux l'on trouverait la liste des utilisateurs autorises

Une date limite pourrait etre affectee a chaque utilisateur, l'empechant d'utiliser cette commande au dela.

Details de l'implementation

La realisation est articulee autour d'une liste de controle d'acces (fichier "ACL") dont la syntaxe est definie une fois pour toutes.

Le programme comporte 2 modules acces.lex et acces.yacc constituant respectivement un analyseur lexical et un analyseur syntaxique pour le fichier ACL. Le programme principal se trouve dans l'analyseur syntaxique et est ecrit en C.

Syntaxe du fichier ACL (en langage de description "yacc")

```
acl    : bloc
       | bloc acl
       ;
```

⇒ le fichier acl est un ensemble de blocs

```
bloc : ligne_de_commande    bloc_utilisateur
     ;
```

⇒ chaque bloc comporte une ligne de commande et un bloc utilisateur

```
user_block : command_line
           | command_line  user_block
           ;
```

⇒ each user block is a collection of user lines

```
command_line : NAME   PATH
             | NAME   PATH1   PATH2
             ;
```

⇒ a command line consists of a command name, spaces, a pathname used to access the command, and, optionaly, a pathname for a log file to log each use of the command.

```
user_line : USER STAR
          | USER DAY SLASH MONTH
            SLASH YEAR
          ;
```

⇒ a user line consists of a login name, space, and a limit date which is the expiry date for the permissions, be this dd/mm/yyyy or an asterisk indicating that there is no time limit.

Refusal

Access is refused with one of the following error messages:

```
"Usage: acces command [args]"
                        : bad command syntax
"Unknown user"
                        : unknown user
"Who are you?"
                        : unknown user
"Can't open ACL file"
                        : ACL file open failed
"Command not in ACL file"
                        : command not in ACL file
"Permission denied"
                        : non-listed user
"Expiry date exceeded"
                        : expiry date passed
"Exec failure"
                        : exec failed
"Syntax error in ACL file"
                        : syntax error in ACL file
```

```
bloc_utilisateur : ligne_utilisateur
                 |ligne_utilisateur bloc_utilisateur
                 ;
```

⇒ chaque bloc utilisateur est un ensemble de lignes utilisateur

```
ligne_de_commande : NOM   PATH
                  | NOM  PATH1  PATH2
                  ;
```

⇒ une ligne de commande comprend un nom de commande, des espaces, un chemin pour acceder a la commande et eventuellement un chemin pour acceder a un fichier ou l'on enregistre tous les acces

```
ligne_utilisateur : USER   ETOILE
                  | USER   JOUR   SLASH
                    MOIS   SLASH  ANNEE
                  ;
```

⇒ une ligne utilisateur comprend un nom d'utilisateur, des espaces et une date limite qui est soit jj/mm/aaaa soit une etoile indiquant qu'il n'y a pas de limite d'acces.

Cas de refus

L'acces est refuse avec un message d'erreur dans les cas suivants :

```
"Usage: acces commande [args]"
                        : mauvaise syntaxe de la commande
"Utilisateur inconnu"
                        : getlogin n'a rien trouve
"Qui etes-vous ?"
                        : getpwuid n'a rien trouve (?)
"impossible d'ouvrir le fichier ACL"
                        : fichier ACL non lisible ou non accessible
"Commande ACL non trouvee"
                        : commande inexistante dans le fichier
"Permission refusee"
                        : utilisateur non liste pour la commande donnee
"Date limite depassee"
                        : fin de droits
"erreur d'exec"
                        : pb d'exec de la commande
"erreur de syntaxe dans fichier ACL"
                        : erreur de syntaxe ACL
```

Second chance

In certain cases (ACL file not found, permission refused, date limit exceeded) a second chance is given, by asking for the root password:

Password:

In the case of the password being correct, the command is exece'd in the traditional manner (using the PATH of the user), and executed as SUID root.

Conclusion

The tool described here does not replace the existing system administration tools. Nor does it disturb existing system organisation, it is simply an improvement. An improvement made without - at least we hope! - introducing an Achilles heel into the system.

This solution solves the majority of problems of access rights to UNIX commands, without disturbing the current groupings of users.

Actually, the tool applies not only to UNIX commands, independently of file access rights. Putting a user into a group, not only gives that user access to commands accessible by that group, but also to all files accessible by them - this is not always acceptable.

Thus, the system administrator has a more supple means at his disposal in the organisation of access rights. Problems of temporary access rights which have been solved before "by hand" or by *cron* can now be solved much more easily.

Deuxieme chance

Dans certains cas (commande ACL non trouvee, permission refusee, date limite depassee), on donne une deuxieme chance a l'utilisateur en lui proposant de taper le mot de passe root en reponse a la question :

Password :

Si le mot de passe tape est conforme a celui qui se trouve dans /etc/passwd, alors on cherche la commande dans l'environnement de l'utilisateur (variable PATH) et une fois trouvee, on essaye de l'executer avec les droits de root.

Conclusion

L'outil propose ici ne se substitue pas aux outils d'administration existants. Il ne remet pas non plus en cause l'organisation du systeme, mais vient plutot completer en l'ameliorant, et sans intoduire de talon d'Achille - du moins nous l'esperons - dans la gestion du systeme Unix.

La solution retenue devrait permettre de resoudre la plupart des problemes d'acces aux commandes Unix "sensibles" sans remettre en question la repartition des utilisateurs en groupes de travail.

En effet, l'outil propose ne s'applique qu'aux commandes Unix, independamment des fichiers; a l'oppose de mettre l'utilisateur dans un groupe c'est lui donner acces aux commandes accessibles par le groupe mais aussi aux fichiers accessibles par ce meme groupe ce qui n'est peut-etre pas toujours souhaitable.

Une plus grande souplesse dans la gestion et la repartition des droits est ainsi apportee a l'administrateur du systeme. De plus une reponse est apportee aux problemes des droits temporaires que l'on gerait habituellement a la main ou par *cron*.

Home-Directory Mail System

Dr. Andrew J Findlay
Andrew.Findlay@brunel.ac.uk

Manufacturing and Engineering Systems
Brunel University
Uxbridge
UB8 3PH

Andrew Findlay is a lecturer on the Special Engineering Programme at Brunel University, teaching computing and electronics. He is system manager for some of the University's networked computers, and is responsible for electronic mail systems across the whole campus.

Abstract

To make the mail system behave in a more obvious way on networks of workstations, mailboxes are moved into users' home directories. User Agent programs are modified, and a new mail notification system is introduced.

Background

Brunel University has a large network of Unix-based computers. There are about fifty machines covering seven departmental subdomains. There are four different processor types. All machines use Sun's NFS, and all users have a single home directory that is exported to all machines where they might log in. Most workstations are in communal areas so a user is likely to log in on a different machine each day.

The mail system is based on the *UK sendmail 1.4a* package [1], and users create *.forward* files to specify which machine their mail should be delivered to. This approach worked well when there were only a few machines, but has become increasingly awkward as the network has grown. To avoid locked-up machines, most user filesystems are soft-mounted. This means that there can be times when a machine is running but a group of user directories are not present. If mail arrives at such a time, the *.forward* file is not effective. A user's mail could be spread around the whole network, with one or two messages on each machine.

With increasing use being made of the mail system, it was felt that some improvements were needed.

Options

- *Use a big alias file*
  This would avoid the 'invisible forward file' problem, but it still requires each user to nominate *one* machine where they will read mail.
- *Mount a single /usr/spool/mail directory for the whole Campus*
  This is probably the simplest solution, but there would be problems with mail delivery

agents that run as root. For security reasons, NFS maps *root* into *nobody* when accessing remote filesystems. Mailers that expect to use lockfiles or *flock(2)* would have problems too.

- *Mount one /usr/spool/mail directory for each subdomain*
  It would be possible to arrange that only the (NFS) server machine ever delivered mail. This would remove the root-mapping problem, but would not solve the lockfile problem. Many users are registered in more than one subdomain, so this solution would not help them.

- *Create a Campus-wide 'mail server' as a TCP service*
  This is possibly the best solution, as it would be able to serve non-NFS and even non-Unix machines. The work involved is considerable, as all the User Agent (UA) programs currently in use expect mailboxes to be *files* in v7 mbox format.

- *Move the mailbox out of /usr/spool/mail into the user's home directory*
  Although the UAs still need modifications, it was felt that this would be the best compromise between effort expended and results obtained.

Mailbox in Home Directory

The main benefits of this approach are: The system behaves in an 'obvious' way.
If the user's home directory is available, then so is their mailbox. Mailboxes become subject to the same quota limits as other files. The existing mailbox format can be retained.

There are some problems remaining: Mail User Agent programs need modifying to reflect the new mailbox position. Some UAs react badly to the disappearance of a soft-mounted filesystem.

Files and Lockfiles

The v7/BSD convention places all mailboxes in /usr/spool/mail and names them with the username of their owner. Lockfiles are named by appending '.lock' to the mailbox name.

The home-directory mail system names all mailboxes '.mailbox' and all lockfiles '.mailbox.lock'. Mailboxes are identified by the directory they are placed in. The /usr/spool/mail area is not used at all.

Mailbox locking is the least satisfactory part of the home-directory mail system. Sun NFS is stateless: this means that the conventional 'exclusive create' mechanism cannot be trusted. None of the traditional atomic file operations can be used, as NFS does not guarantee atomicity. The 'correct' way to lock files is to use *flock*, which makes use of the lock daemon to handle local and remote files in the same manner. Unfortunately, not all implementations of NFS support the lock daemon. As a compromise, it was decided that home-directory mail would use lockfiles *and flock*. This would give complete safety between machines supporting *flock* and reasonable safety on all others.

The mail delivery agent may have to create a new mailbox to deliver into. In this case, the lockfile must be created before the mailbox. The *flock*-style lock is therefore always applied to the lockfile.

Delivery

A new program called *hdmail* was written to handle mail delivery. It is simplified by not doubling as a User Agent. *hdmail* is aware of the peculiarities of soft-mounted filesystems: if a directory is missing or a file suddenly becomes inaccessible, the value EX_TEMPFAIL is returned to *sendmail*. Mail for users whose home directories are not available will be held in the mail queue, and delivery will be attempted again at each queue run. This has the useful side-effect that *sendmail* will look for the *forward* file again at each queue run, making the forwarding mechanism much more reliable. If the necessary directory does not appear within a reasonable time, *sendmail* will return the message to the sender.

Hdmail runs as the recipient user, not as root. This gets round the root-mapping problem on remote filesystems, and also improves security. When secure NFS [2] is introduced, it may be necessary to have mail delivered only by the machine that supplies the user's home directory.

After the mail has been delivered, *hdmail* arranges to notify the user of its arrival.

Notification

The home-directory mail system allows a user to read and send mail when logged in on any of the fifty-plus Unix machines on Campus. When new mail arrives, the user expects to be told about it.

Under the BSD version of the mail system, this function is handled by the *biff* daemon. *Biff* is a UDP (datagram) service; it is sent a packet containing a username and a mailbox-offset. If the named user is logged on, the mailbox is inspected from the given offset to construct a suitable message for display on their terminal.

Biff provides a good service on individual computers. It is not so useful on a large network of workstations, as a user may not be logged in on the machine that delivers the mail.

Four approaches to the notification problem were considered: Modify *biff* to read the new mailboxes, and invoke it on all known machines. Use a background 'watcher' process, such as *newmail* from the *elm* suite. Use the shell's MAIL variable. Create a new network service.

With the number of workstations and the volume of mail both increasing, it was felt that (1) would soon impose an excessive load on the system.

Solution (2) was ruled out as it would almost double the number of active processes on undergraduate teaching machines.

Solution (3) would be a backward step, as users would only find out about new mail when they returned to the shell.

It was decided to create a new network service, similar to *biff*, but with less work done by the daemon. The new service, *tell*, is a UDP service that receives a packet containing a username and a message. If the named user is logged in and has set the '*biff*' bit on their tty, the message is written to that terminal. For security reasons, only a limited set of control characters is allowed in the message. Any 'illegal' characters are converted to spaces by the daemon. The *tell* daemon provides a more general service than *biff*. It is not limited to mail notification, though that is its only function at the moment.

To avoid having to invoke *tell* on every machine on the network, another file is maintained in the user's home directory. The '.whereami' file records the name of each machine where the user has logged in, together with the time of login. When *hdmail* is ready to notify the user that new mail has arrived, it scans '.whereami' and sends a *tell* packet to every machine where the user has logged in within the last 48 hours. If the file does not exist or if no recent logins are recorded, a *tell* packet is sent to 'localhost'.

The '.whereami' file is maintained by a program called *mailcheck* that is invoked at login time from '.login' or '.profile'. *mailcheck* also prints a message if mail is waiting to be read, thus avoiding the need to modify the shells and the login program.

User Agents

The major UAs in use at Brunel are *elm*, *Mail* (/usr/ucb/mail), and *mailtool* (under SunWindows only). Since *mailtool* is a driver for *Mail*, it was only necessary to modify two UAs to satisfy almost all the users.

Elm was tackled first, taking version 1.5b as the starting point. References to mailboxes are scattered in the source code, which made the conversion rather tricky. The utility programs had to be modified individually. *elm* still uses lockfiles only; it is hoped that version 1.7 or 2.0 will have a cleaner interface to the mailbox, allowing the full locking protocol to be used.

Mail was modified using source code from the 4.3BSD distribution. The changes required were confined to two source files. Unfortunately, the 4.3 version of Mail is not compatible with *mailtool* so this work had to be repeated when Sun source code was obtained.

Other UA programs have been converted, including *prmail*, but not */bin/mail* (which is superseded by */usr/ucb/mail*). The interface between mail and *emacs* has not yet been converted. Other minor links to the mail system (such as the News '(Mail)' prompt) can safely be ignored.

Mail Transport Agents

Brunel uses *sendmail* for transport and routing of mail. The only modifications needed were in the part of the configuration file describing the 'local' mailer. *hdmail* replaces */bin/mail* for local mail delivery. The only major difference is that *hdmail* can only deliver to one user per invocation.

Operating Experience

Nobody noticed....

Apart from one incident when an automatic updating system installed the modified UAs on a machine that was not due for conversion, the transition has been smooth. Problems were encountered when re-compiling *Mail* on certain machine types: in one case caused by an optimiser bug, and in the other by bad memory-

allocation code.

The home-directory mail system is now operating in three of Brunel's six Unix-based subdomains, on two different CPU types. It is to be installed in the remaining subdomains shortly, which will include two more CPU types.

References

Crammond, J., "Domain Mailers: Sendmail", Workshop on Networking, UKUUG, December 1987

Taylor, B., "A framework for Network Security", Sun Technology, Spring 1988.

Have you seen Computing Systems?

You should be able to answer *yes* to that question at once and to recognise that **Computing Systems** is the recently launched journal of our American sister organisation **USENIX**.

The links between the EUUG and USENIX are many and well established, as anyone who attends EUUG conferences must be aware. It is no surprise therefore to note that on the Editorial Board of this new journal are three well known EUUG figures - Peter Collinson, Nigel Martin, and Jaap Akkerhuis.

The journal, as the President of USENIX made clear in his welcome message in the first issue, is a suitable home for papers of a practical nature rather than a theoretical nature. A touchstone used in selecting articles for inclusion is whether the information contained in them would be of likely value five years or more ahead.

Computing Systems, with its high standards and high hopes, and with a strong editorial team from the international world of **Unix**, has been well and truly launched. It has already attracted a great deal of attention.

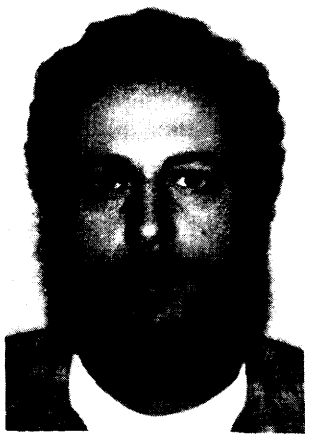
What more can we tell you? Probably nothing, except to say that at the special rate of $23/year (including postage) to EUUG members, it has got to be good value. If you have missed seeing a sample copy of the first issue: 'phone, fax, email, or write to Owles Hall and ask for one.

Act now - if you are not plugged in to **Computing Systems** you are really missing out.

i2u Annual Convention

Joy Marino
joy@ugdist.uucp

i2u Annual Convention
June 6th-8th, World Trade Center, Milano-Assago



Joy Marino is associate professor of Computer Science at DIST (Dipartimento di Informatica, Sistemistica e Telematica), University of Genoa. He started using UNIX in 1980, when he was interested in ADA: now he is still using (and teaching) UNIX, C and C++, but no more ADA.

He has been involved with *i2u* since 1984, first as editor of the *i2u* newsletter (**UNI***forum*) then (since 1986) as member of the Board. He has also been appointed as *i2u* representative in the EUUG governing board (probably because other members *think* that he is good at English).

Introduction

I cannot claim to be objective in this technical report on the Annual Convention of UNIX in Italy: of all the parties represented here there is one that I would like to give a special mention to, since it is one I am involved with. I am of course referring to the *users*, better: the Italian UNIX systems Users Group, which this Convention has definitely recognised as an essential reference point for the Italian IT industry. If I allow myself to be a little sentimental, as someone who has always believed in UNIX, from the time when they could say "it's crazy to have the same operating system for microcomputers and mainframes" without people laughing, as someone who remembers when *i2u* brought together 60 people for a first Round Table on UNIX (1984), as someone like me, the high point - the feeling of finally having "made it" - was when, on the morning of the first day of the Conference, the partitions were opened up to transform a 400 seater hall (by that time with standing room only) into a 700 seater.

That's enough sentimentalism for now - let's get down to brass tacks.

Programme

Monday 6th June: Tutorials

9:00-12:30 An introduction to X Windows System.

9:00-12:30 Evolution of the C programming language.

14:00-18:00 An overview of UNIX standards: from BSD and SVID to POSIX.

14:00-18:00 UNIX tools for Office Automation.

Tuesday 7th June

8:30 Registration.

9:30 Opening session (A. Camici, Chairman of *i2u*).

9:45 UNIX: the state of the art (J. Marino, *i2u*).

10:15 UNIX in the U.S.A. (P. Gray, /usr/group).

10:45 Coffee break.

11:30 UNIX: International market survey and trends (R. Masiero, IDC).

12:00 The UNIX market in Italy (D. Gerundino, Mate).

12:30 Lunch.

14:30 Open Session on "The future of UNIX systems".

- X/Open Directions (J. Totman, X/Open).
- Convergence of new AT&T Releases (J. Knowles, AT&T).
- New developments at SUN (B. Joy, SUN Microsystems).

16:30 Coffee break.

17:15 Open Session on "The future of UNIX systems".

Apollo - Digital Equipment - Hewlett Packard - Honeywell Bull - IBM.

Wednesday 8th June

9:00 Open Session on "The future of UNIX systems".
Olivetti - Philips - Siemens - UNISYS.

10:45 Coffee break.

11:30 Integration of UNIX and PC world (H. Chuck, Microport).

12:00 UNIX for department systems: a case study (M. Crescioli, Unirel).

12:30 Lunch.

15:00 Session on user experiences and applications.

16:30 Coffee break.

17:00 UNIX in Europe: the EUUG (K. Biel-Nielsen, EUUG).

17:40 EUnet and the UNIX network in Italy: strategies and perspectives (*i2u*).

18:45 End of the conference.

Introduction

The following notes are my personal thoughts about the conference. Usually *i2u* meetings are devoted to industry people, and are not technically oriented as EUUG's or other national groups' are. Furthermore, this time the Conference was scheduled to be held 15 days after the first public announcement of OSF, and we have had to rearrange the progamme in "real time" in order to find place for all voices and points of view.

Never the less, we could lever on the interest that OSF has built up about UNIX to a wide audience; *i2u* Convention has been one of the first opportunities in Europe for confronting the different points of view about the evolution of UNIX in the light of this announcement.

As a whole the Convention has been quite successful: 560 attendees at the Conference, over 350 for tutorials, a full floor of Centro Congressi - Milano Fiori filled up with exhibition booths for three days.

The mysterious object

My own statistics (obviously a rough one) of the most frequently used words came up with: 1) "OSF", 2) "POSIX", 3) "ABI", 4) "X/Open", and then all the rest. It is clear that everyone's attention is focused on the Open Software Foundation.

I would like to give special mention to Dave Nelson, appearing before the convention with two hats: as the Vice-President of Apollo, and as the official speaker for the much discussed Foundation. He gave his talk mainly with his second hat, but could not totally avoid his role as spokesman for Apollo.

His vision of the IT world outlined in his address, a very complex world with many different operating systems, countless communication networks and systems, very astutely set the scene for going on to describe the advantages of a new world based on UNIX (and OSF), but how could one fail to notice that many of the complications of the proprietary world are actually due to the partners who are now seated around the same table in the Foundation?

So what is the OSF? It is an autonomous organisation, organised as a non-profit making foundation, to which the founders have given either grants of several million dollars (90) - sufficient to produce the "critical mass" - and a fraction (large or small - this remains to be seen) of their wealth of know-how on open systems. OSF will not produce new standards, it will write lines of code; and in this world, where standardisation committees abound and where the real programmers stay well clear, one can only commend their initiative.

Two of the better proposals of OSF cannot be disputed: to have a process for actively encouraging new proposals and new technologies,

to promote technological progress through universities and research centres. In a word: to recreate the "virtuous circle" that made such an important contribution to UNIX at the time of Bell Labs and the first releases of the University of Berkeley. I'm not too convinced about the third proposal: "to have a product range based on the *relevant* industrial standards". Who decides what is "relevant"?

Quoting his words, the aims that OSF sets out to achieve are therefore:

— a decision-making process not modelled by any particular manufacturer (as is happening with AT&T);

— equal right of access to the specifications and to the initial development phases (and not 6 months delay in getting the sources of the next release of System V from AT&T);

— reasonable and consistent conditions for licencing agreements (and not the mess that AT&T put together for the licences of System V.3);

— implementations that really are hardware independent.

As you can see, a scenario where AT&T appears in quite a bad light, almost as if it were the "enemy" that wants to take over UNIX.

In practice things are not quite like this, not only does AT&T stand no chance of transforming UNIX into a proprietary product, but also - as Pamela Gray, Chairperson of /usr/group, said - it's organisational structure is so complex that it cannot be credited with a "perverse mentality" (even if it is capable of acting like a bull in a china shop).

Nelson answered a lot of questions, all aimed at clarifying the shape of this mysterious object. Here is a summary of them. Licence costs? - we will offer the best products at the lowest cost. When will OSF be available ? - we hope within 18 months, but this is not definite. Will there be validation procedures? - yes, the usual procedures, but these will not be strictly adhered to. Will proprietary expansions be possible? - of course, each producer will be able to expand as he thinks best.

The last two points give rise to certain doubts: what future is there for a product that from the start does not impose a guarantee of uniformity for the various implementations?

After Nelson spoke, the other partners in OSF[1] also outlined their views on the strategies for the UNIX environment, even if not all the addresses had a "strategic" line and at times provoked murmurs of disapproval and even a spontaneous applause for the Chairman of the session, Gregorio Lerma (Banco Popolare di Bergamo, but here acting in the capacity of Vice-President of *i2u*), who stood up to rebut some particularly heated comments.

The talk by Ron Kita, DEC, was not exactly riveting, but his allusion to possible future POSIX compatibility of VMS re-awakened the interest of some of the audience, who had attended the tutorial on "Standards in comparison" concluding with an open discussion on the role of POSIX. Here are a couple of questions which I would like to see answered at the next *i2u* Convention: "What place will non-UNIX operating systems have in POSIX?" or even "Will the POSIX compatibility of VMS have any relevance in the US Government tenders?"

AIX, IBM's version of UNIX, is to play a leading role in OSF. Many doubts have been voiced on the current and future AIX characteristics, doubts which Koch (who spoke for IBM) failed to remove completely in his talk. As far as certainties go, we can point to IBM's commitment to UNIX, with the announcements of AIX on the PS2 and on the 937X, but as far as uncertainties go, we can point to the AIX characteristics on which OSF is going to build its future, an AIX that is different from the current one, but derived from it - i.e. an operating system for which IBM still has to pay its licence fees to AT&T.

It would take too long to list all the speeches that made reference to OSF, especially since almost all of them broached the subject more or less directly. Chuck Hickey, of Microport, speaking about the integration between UNIX and the PC world, had to say his piece as well, and immediately pinpointed the central problem: OSF was born as a response to the notion of ABI - "Application

1. For those who did not know them, names were listed in one of the slides in his presentation: Apollo, Bull, DEC, HP, IBM. The rest - Siemens, Nixdorf, Philips - came later on, and apparently there was not enough time (?) to redesign the slide...

Binary Interface" - i.e. binary compatibility at the level of the individual applications. ABI was announced at the Uniforum in Dallas, initially as a proposal only for the SPARC processor by SUN (SPARC-ABI) but immediately afterwards for other major microprocessors as well, starting with the Intel 80386 (386- ABI). According to Hickey, this was not merely a response to a possible hegemonic attitude by the SUN/AT&T alliance, it was primarily worry about a newborn phenomenon that might develop in the same way as has happened in the PC world, where binary compatibility has made it impossible for any large corporation to control the market. The fact that the UNIX market, for the wide range of products from PC's to mainframe, is potentially far more extensive than the PC market, gives us grounds for excitement or anxiety (depending on our point of view...)

I have tried to act as an impartial observer at the Convention, and my introductory speech also tried to outline all the positive aspects of the two proposals and strategies, SPARC/ABI and OSF, which are at the centre of so much debate. But the Convention has left me somewhat perplexed: on paper OSF looks as good as it can be, but the various partners who belong to it are still too fragmented, the motives driving them to this step are too varied, just as their expectations seem to differ as well.

OSF can hope to really succeed only if it shows that it is truly authoritative, empowered with the type of authority that cannot be derived from proportionate sums of capital of the corporations that founded it (addition that does not have any significance anyway), but from the cultural prestige of an authentic "neutral" body. Perhaps Kim Biel-Nielsen, attending the Conference as the EUUG delegate, is right when he says that everyone ought to join OSF, especially all the user groups such as EUUG, to ensure that this initiative really does provide a means of progress for UNIX.

SUNny Bill

Bill Joy shone like the Californian sun - not just because he was dressed in a casual white and yellow outfit, nor because his company is called SUN, nor because he was the main point of reference that the AT&T speaker, Jim Knowles, had to turn to when replying to questions. Delivering an epochal style speech from on high, he steered well clear of the squabbles that other players in the field are embroiled in. Or rather, this is how things ought to have been, had it not been for a few punches below the belt which he managed to get in. Having sat next to Bill Joy on the podium and seen the simple scratch pad he used to base his presentation on, I can only express great admiration for his ability to construct a strategic presentation in "real time", but I can also understand why many corporations are not pleased with the explosive mix of AT&T capital and the SUN-strokes of genius. Bill Joy says that the future lies with standards, and quotes a long list ranging from ANSI C, to the floating point format of the IEEE standard, to the SCSI peripherals bus, putting in the middle of the list the "SPARC" (the RISC processor that SUN has publicised so brilliantly) as if this were a consolidated standard as well. At this point I would like to put forward two theories: either Bill Joy is "trying it on", or he really is a few steps ahead of everyone else, and has come to tell us what our immediate future looks like. I have no way of deciding which is correct.

This immediate future, which Bill Joy reckons is 1991, can also provide the recipe for making supercomputers: take 50 boards, each one with a 100 MIPS processor and about 100 MB memory, all of which is available from your High Street dealer, add magnetic or optical disks as necessary, and for a fraction of the cost of a CRAY you can build a 5000 MIPS machine, with 5000 MB main memory, and so on, not to mention reliability, with a potentially infinite MTBF. There was only one fleeting mention of the glue needed to tie together this array of processors: didn't someone perhaps say that UNIX is the glue of the IT industry?

I would have agreed with Catellan, from HP, who branded this "multiprocessing paradise" an example of superficiality, if it were not for one detail: exactly one year ago Bill Joy gave a talk at the EUUG convention in Helsinki, where he presented the same promises of increase in the computing power of the single processor (doubling every year from 1984 to 1992), but, on answering a specific question, excluded the possibility of development in the short term for multiprocessor architectures. We need to await the first results of the Menlo Park laboratory (where the real programmers of SUN and AT&T are working side by side) before we know whether the UNIX System V Release 5 will be a serious argument in support of these "visionary" talks.

Libert, Egalit, Fraternit

The *i2u* Convention was not just a conference and debate on strategic subjects, it also put on tutorials and product exhibitions.

The success of the tutorials is, I believe, confirmation that the Italians are much closer to the Japanese than to the Americans and are therefore "hungry for knowledge", and are interested in attending a UNIX convention to gain more insight into technical subjects, as well as hearing about AT&T and OSF. Incidentally, the tutorials on the first day were perhaps the only time when the organisation of the Convention proved a little shaky, especially since the "desire for knowledge" of people interested in UNIX was overestimated. This was a fault that is bound to be easily rectified next year.

The exhibition was rather drab, with stands all designed the same, the exhibitors names on impersonal and simply laid out cards. A whole set of "signals" that convey a *message* that even the most unaware visitor could not fail to miss: UNIX levels out everyone - large and small. Market opportunities can be full filled by offering better machines at more competitive prices: the starting point must be the same for everyone. Of the 31 exhibitors present, all of the main suppliers selling systems were there, and some of the software houses more active in the UNIX environment.

Conclusions

Anyone attending the *i2u* Convention to find out if it is worth investing in UNIX came away with definite answers: Masiero (IDC) was very clear on this point: "you can count on UNIX, now" is the moral of his talk, strengthened by dozen of bar-charts, pie-charts and trend analysis. "You can count on UNIX" has been also the six-column title of a report about the Convention that appeared afterward in the main Italian economical newspaper.

Anyone attending to see whether we will finally manage to combine all the versions into a single standard came away with provisional answers; the situation is certainly better than in the past, but we have still not reached a single UNIX version that is the same for everyone: anyone deluding themselves that the announcement of convergence between System V and SUN/OS (i.e. UNIX BSD) was the decisive step, would have been dismayed to hear the announcement of OSF. In my view the panorama that is unfolding is however acceptable; as we can see now, we have shifted from a past scenario with 3 or 4 versions (System V, Xenix, BSD, and X/Open), all quite different from each other, to a future scenario where we will have two versions at the most: System V.4 and derivatives on the one hand, and OSF on the other, with an extremely high degree of overlap between them (since both refer to the same consolidated standards: POSIX, X/Open). The world of UNIX has always been used to having different versions at the same time: continuing to follow the path already mapped out presents no big problem.

If I can quote Henry Spencer, one of the technical "gurus" in UNIX, who says: at a given point in time there is a set of functionalities that everyone treats as "standards", and another set seen as "expansions" on which there is no agreement as yet; at a later point in time the set that agreement has been reached on is enlarged, because consensus is reached, but at the same time the same set of expansions broadens, since other technological areas are incorporated or new applications are required.

1988 has been a significant time for both these sets.

The Trouble with PostScript and Device-independent Troff

Markku Sakkinen
markku@jytko.jyu.fi

Department of Computer Science, University of Jyväskylä
Seminaarinkatu 15, SF-40100 Jyväskylä, Finland

Graduated in mathematics in 1971 and got the Licentiate degree in 1972. However, had already lost his heart to computers and was not quite bright enough for a mathematical career. Worked at the university computing centre and also at the department of physics (putting up a laboratory computer system). Then did several years of Real Work in the Real World (sort of industrial automation mostly). Returned to his dear old university in 1985, partly in the hope to get a PhD before retirement age, but got to bear hardships, e.g. UNIX™.

Why the trouble to write this article?

I have done a lot of work, somewhat intermittently during a rather long time, trying to make the device-independent Troff formatter and a PostScript™ printer serve well the text-producing needs of our department. The documents range from one-page letters to entire books, very typical being journal and conference papers such as this one. The main languages are Finnish and English. There is not much mathematical notation on the average, because the principal area of research is information systems; the department belongs to the faculty of Social Sciences. The principal computers of the department's own are a DEC™ VAX™ (with 4.3BSD) and a newer Sun-3™.

Reflecting on the experience, I concluded that there are sufficiently important drawbacks in all software products concerned that they be brought to the attention of both those products' developers and the general public. This is all the more true since PostScript is now fashionable everywhere and the Roff family of formatters (Ditroff and the whole Documenter's Workbench™ especially) probably continue to be utilised by a significant part of UNIX installations for some time to come. Also, I have not encountered much critique on PostScript even in books like [Earn] and [Holz], at least not on those problems that I will stress in the sequel. This paper will not be only a list of complaints; I hope that it can help other people to install and exploit text formatting systems more smoothly, and give some useful or interesting insight into the principles of these tools.

PostScript

Basics

PostScript [Adob1, Adob2] is a page description language and a stack-oriented programming language at the same time. While a print job for a conventional printer is a sequence of printable characters interspersed with control characters and escape sequences where needed for special printer functions, a print job for a PostScript printer is a PostScript programme, in which the actual text to be printed is normally, but not necessarily, interspersed as literal strings. The print job can also output "anything" back to the client (host computer). The PostScript language is very interesting, flexible, and powerful in many respects: it contains e.g. a rich set of graphic operations. Nevertheless, there are surprising limitations in its primary function, namely the printing of characters.

The names by which fonts are known in PostScript are long and explanatory, like 'Times-Roman'™ or 'NewCenturySchlbk-BoldItalic'. The basic defining entity for the character set of any built-in PostScript font is a dictionary (associative array) called **CharStrings**. Its keys are the names of all characters in the font; for the upper and lower case letters of the English alphabet, the names are the letters themselves as one-character ASCII strings, for other characters, they are such rather self-explanatory words as 'three', 'comma', 'asterisk', 'parenleft', 'Sigma'. The value (object) associated with each key is the actual shape description that PostScript uses to draw the character.

According to the general principles of PostScript, it should be possible to get a shape description from **CharStrings** by using a character name as a key, convert it into executable and execute. In fact, this is how the **show** operator finally prints every single character, according to section 5.4 of [Adob2]. However, if one tries to do the same directly in a PostScript programme, nothing happens. My conclusion is that the true shape descriptions are completely protected even from execution (presumably for proprietary reasons) and that only the **show** operator has special access to them. Therefore, the only way to get characters printed is to use the **show** (or **kshow**) operator to strings of *encoded* characters; it is the encoding that proves to be a bottleneck in the business.

Character encoding

PostScript defines *strings* as special arrays with small nonnegative (0 to 255) integer elements. When a string is to be printed, the language itself does not define how these codes should be mapped to characters. In order to interpret the codes, each font description must contain an array of 256 character names called **Encoding** — user-defined fonts will not be accepted if they lack this array. String elements are simply used as indices into **Encoding**. There is an array in the system, **StandardEncoding**, which maps all printable ASCII codes and many codes between 161 and 255 to PostScript characters; it is used by most standard fonts as their **Encoding**.

In the normal case, a host computer communicates with a PostScript server (printer) by 8-bit or 7-bit transmission. The whole PostScript communication except for strings needs only 7-bit ASCII characters. When 7-bit transmission is employed, string characters with codes above 127 cannot be sent as such but as a four-character sequence: backslash followed by a three-digit octal number (digits coded in ASCII). This method, borrowed from the C language, can also be used to pass codes that would otherwise conflict with ASCII control characters.

The above encoding scheme is quite flexible. There are examples in the book [Adob1] on how to change the encoding to add Scandinavian characters and how to define an encoding to accept EBCDIC instead of ASCII characters in strings. Unfortunately, the scheme has drawbacks also. Because of possible conflicts with ASCII control characters, 67 codes should in principle not be used for printing characters, leaving 189 usable codes. This precaution can be necessary due rather to support software than to PostScript itself (cf. chapter 3), as the octal escapes are available. The standard encoding actually defines only 149 characters, and therefore 61 characters that exist in the normal fonts were left unencoded already in PostScript version 25.0. That is why one has to change the encoding first in order to get Scandinavian characters printed, as mentioned above. In PostScript version 47.0, other built-in standard text fonts have 228 and the Courier fonts 260 different characters; there is thus no way to have all of them encoded at the same time.

It is real lack of foresight that a page description language designed in the 1980's has been constrained to an 8-bit character code. After all, 8-bit EBCDIC code was introduced with the IBM® System/360 in the sixties, though most of the 256 possible characters were initially undefined there. With the backslash-octal number notation as it stands, one could express 512 different encoded characters. By devising a more efficient scheme, one could access even much larger internal character sets with "escape sequences" of just three (instead of four) ASCII characters. The most frequent characters would still be transmitted as themselves. Larger encoding vectors would demand more storage space, but they would not need to be excessively large. As regards the space taken by strings to be output, a PostScript server would in typical cases need to store only one word at a time in string form, in principle. The primitive memory management of current implementations (cf. section 1.4) evidently implies that in practice it can be necessary to store simultaneously all strings belonging to one print page.

As explained, a PostScript user is presently stuck with the 189-character limitation in the worst case, and cannot keep all characters that actually exist in a font mapped. In consequence, the need to temporarily access an unencoded character can arise. There would appear to be two ways to do it: either modify the encoding temporarily, or try to get the character printed directly from **CharStrings**, thus bypassing the encoding. Unfortunately, the second, more natural and straightforward alternative does not seem to work (cf. section 1.1), so we will look at the first one.

More about character sets

Modifying an encoding vector is not as simple as one might think, mainly because one must first copy a lot of structures, then do the actual modification, and lastly define the modified font as a new font to PostScript. (See the "cookbook" programme 18 in [Adob1].) The source code to accomplish this can be downloaded as a PostScript procedure once and invoked with appropriate parameters every time one or more new characters must be encoded. Even then, the extra work to be done by the PostScript engine will slow down printing. Moreover, there is a snag in the whole principle of re-encoding: Some (accented) characters are internally represented as composites of two or more simple characters of the same font. If such a composite character is encoded but one of its components is not, the character code will not cause anything to be printed. One must *guess* all such dependences, they are not clearly documented anywhere.

The composite character snag mentioned in the previous paragraph seems to imply that even the access of the **show** operator to the character descriptions is queerly bound to the encoding vector of each font. It is queer because surely the description of a composite character cannot refer to the components by code; otherwise the encoding of any component characters could not be changed without causing bizarre results. The PostScript language should be made more consistent and orthogonal in the handling of characters. Since an **execute-only** access attribute can be defined for any PostScript object, it would hardly constitute a risk for secrecy to let every character description be directly executed. This method to print seldom-needed characters would be much more efficient than the modification of encoding vectors.

Almost all *non-PostScript* printing devices and many display terminals of today recognise **national character sets** as a dimension orthogonal to **fonts**. At least as long as we have a lot of 7-bit-oriented hardware and software, this will remain an important distinction. An alternative to national character sets is one comprehensive international character set. PostScript does strive in this direction — the standard fonts of version 47.0 contain even the Thorn and Eth of Icelandic, so that virtually all languages with a Latin-based alphabet are catered for — but the encoding bottleneck makes it difficult to utilise the large character set efficiently, as explained earlier. Probably the most convenient way to introduce national character sets into PostScript would be to allow user modification of the **StandardEncoding** vector so that all fonts bound to it would automatically be affected. (If this article had been printed "at home" instead of being sent electronically as troff source text, you would see the Icelandic letters above; cf. section 2.3.)

There is in current PostScript one Symbol font, which comprises Greek letters, a host of mathematical and other special symbols not found in "standard text" fonts, and some punctuation characters common with other fonts. This approach is familiar from the various *roff programmes. It is probably a convention from the times when typesetter fonts had a very limited number of characters to regard this as a **font**. More naturally, the special symbols would be regarded just as an extension of the character set; they should be available in several different fonts, at least as Roman, italic, bold, and bold italic variants, why not in more than one typeface family as well.

A memory problem

One "programming-language" aspect of PostScript can have somewhat surprising effects on printer operations (cf. section 3.7 of [Adob2]). PostScript does not store composite objects (arrays, dictionaries, etc.) on the operand stack, only pointers (descriptors) to them; the actual objects are stored in heap (dynamic) memory, called *virtual memory* in PostScript parlance. But there is no explicit operator to deallocate a composite object that is no longer needed, nor is there automatic garbage collection. Therefore, memory gradually fills up during the execution of a PostScript programme. This is generally

harmless in ordinary printing jobs, because virtual memory is cleared between jobs (programmes). Also, there is an operator that restores the whole virtual memory to a state saved earlier; this is typically called between pages.

However, it is possible to put objects into virtual memory so that they will remain there from job to job, until the printer is physically switched off. This facility is deemed so machine-dependent that it has been left out of [Adob2] (cf. section 1.6); it is (or should be) described in the specific manual of every PostScript printer. One would often like to download both re-encodings of built-in fonts and complete non-standard fonts just once and then be able to utilise them in all jobs. In a typical spooling environment, a user cannot always be sure whether such a downloading has already been done after the most recent powering up of the printer or not, and may thus send the file again. What happens is that a new set of objects is created but the old ones are not deleted. If this is repeated, virtual memory fills up and print jobs start to fail, first those which create the most complex pages. To prevent this problem, any downloading file (PostScript programme) should be designed to test first and exit if the downloading appears to have already been done.

Client-server communication

One drawback of PostScript in comparison to most other page description languages *is* mentioned fairly often, e.g. in [Earn]: because of its human-readable style, long keywords, etc., it requires much more characters to be transmitted from client to server for the same printed text. This can be an important factor when slow communication links must be used. However, in typical cases client and server are physically close and have a relatively fast connexion (maybe even a parallel interface or a LAN); the data transmission will hardly be a bottleneck in the printing process.

Perhaps a more serious problem in PostScript communication is the protocol. PostScript printers can provide versatile feedback to the client, which is marked progress from conventional printers. But the data stream from printer to host computer may in the worst case contain both output explicitly generated by the current PostScript programme (print job), spontaneous messages from the printer, and responses to the host's status requests, all arbitrarily interspersed. This makes life hard for the host interface software. For instance, the filters in the TranScript package (section 3.2) can get so confused in some problem situations that the super-user must do several clean-up tricks (besides powering the printer off and on) before they will transmit anything again.

How to get information

There apparently are at least two different versions of the reference manual [Adob2] around. One version contains the device-specific information for Apple™ LaserWriter, of which most is applicable to other PostScript printers as well, but its main text part corresponds to PostScript release 23.0 (even in a 7. printing from August 1987!). The other version is updated with the features of PostScript release 25.0, e.g. packed arrays and a couple of new standard characters, but it does not contain the device-specific appendix. I do not know about any documentation that would tell about what new things there might be in release 47.0. The manuals [Adob3] and [Qume] do not even mention the considerable number of new characters that have been added to the built-in fonts — I had to find out myself.

Adobe Systems have defined a standard format called AFM (short for Adobe Font Metrics, obviously) for files describing metric (and encoding) information of PostScript fonts. This information is essential to host computer software, of course, but it is also meant to be readable by people. It may not always be easy to get an AFM file for every font in one's printer anywhere: the manufacturer or representative of the machine does not know or care, and support software packages typically support only a few fonts. Fortunately, it is possible to dig out approximately the same information out of the printer itself with a clever PostScript programme, but that requires a lot of hard work. We will return to these questions in chapter 3.

Device-independent Troff

Introduction

The device-independent (or typesetter-independent) Troff [Kern1] is a member of the Roff family of batch text formatters. In several contexts, e.g. the AT&T documentation, it is called simply 'troff'; if it is needed to speak about the old troff also, the latter is called 'otroff'. To avoid such confusion, we will henceforth call the

new formatter 'Ditroff'. It is part of the UNIX Documenter's Workbench (DWB) — the whole DWB package includes several preprocessors, a new version of Nroff, and other stuff [AT&T1, AT&T2], but Ditroff is practically the only component that one can install (with some effort) even on a 4.2BSD or 4.3BSD system. Some BSD-based UNIX systems have enough System V compatibility features to install practically all of DWB, but on a "pure BSD" system you cannot even read in the distribution tape directly because for some reason it is in **cpio** format. This discussion is based on release 2 of DWB; at least rather recently the release 1 has still been distributed as a standard or optional part of some microcomputer UNIX systems.

There are three important facets in Ditroff: what its input is like, how it works, and what its output is like. The input is essentially the same as for Troff (the old typesetter-specific formatter), but with some very useful additions. The main difference in the working is that Ditroff needs a a separate device description file and respective font description files for each supported printer type. The output is totally different from Troff output: it is printable ASCII throughout, always needs a postprocessor for actual printing, and thus is normally directed into a file or pipe. We will discuss the output format first.

Output format

The general format of Ditroff output is clear and simple: it consists of commands, each having zero or more (but typically one) parameters after the command character. White space never harms between commands, and it is even required after some commands to clearly delimit their last parameter. There are commands for motions, character printing, drawing, size and font specification, and device control functions. Very important for many purposes is that word, line, and page boundaries are explicitly marked with special commands, even though they cause no action in the default case when the whole file is printed on the type of device it has originally been intended for. Certainly, many people besides myself have sometimes been driven crazy by the stupidity of Nroff, which cannot be even forced to insert honest form feeds into its output.

Every page in a Ditroff output document is self-contained in the sense that all context setting such as mounting fonts and defining point sizes is done at the beginning of the page. Therefore, if a postprocessor is asked to print only selected pages from a large document, it is sufficient to read the general header information at the beginning of the file and the required pages, which can be found easily. It is possible also to feed Ditroff output to the postprocessor of another device than it was originally created for — this exactly is device independence. Besides such things that the new target printer physically cannot accomplish (e.g. impossible font sizes), most likely to go wrong then are non-standard (perhaps even some "should be standard") special characters. — The only information about the original target printer in a Ditroff output file is its name and resolution; this scarcity can further reduce the possibility of optimal printing on another kind of device, no matter how clever the postprocessor.

There is one exception to the above in the Ditroff output format, and that is its most common command: '*xya*', where *x* and *y* are decimal digits. This specifies that the current position is to be moved *xy* units to the right and then the character *a* is to be printed. It is thus a shorthand for the combination of a movement and a printing command: '**h***xy***c***a*'. However, the shorthand cannot be used if the movement is greater than 99 units. Surprisingly, there is no notation for a string of output characters to be printed with their default spacings; this would often reduce the size of an output file by more than 50% (cf. section 3.1). At least, besides '**c**', there could be a command to print one character and then advance the current position by its nominal width.

A nice flexibility in Ditroff is that the set of special characters is not fixed within the programme itself but defined in each device description file. Special characters are denoted in Ditroff output by the command character '**C**' followed by the special character's name followed by white space. This syntax would make long names possible, but they are not allowed in the input. It would be very practical if one could use PostScript character names for unusual characters in Ditroff input, too. It is hard to invent unique and mnemonic two-character names for large character sets. Font names cannot be longer than two characters, either; as many present printers have dozens of fonts available, the same difficulties arise as with special character names. If one gets some support software or Ditroff input text from two different sources, they will almost certainly conflict in their use of both character and font names.

There are no commands in Ditroff output (nor input, of course) to control text and paper orientation. It would be useful, though. Even many non-PostScript printers today allow both horizontal and vertical printing, at least. PostScript allows any oblique direction of text lines as well. The simple case in which a whole document should be oriented horizontally can be handled outside Ditroff, in a postprocessor or printing filter.

Fonts and character sets

Ditroff requires that there be, in a certain public directory, a respective subdirectory for each different output device, containing a device description file and font description files. It is this subdirectory that the device selection option '-T*name*' of Ditroff actually specifies. For every known font, the name of the font description file is the font's own name (at most 2 characters) suffixed by '.out'. Different PostScript printers may very well be described to Ditroff as the same device type.

The same non-existence of national character sets that was mentioned in PostScript holds *a fortiori* for Ditroff. All characters outside the U.S. ASCII set must be defined as special characters in the device and font description files used by Ditroff. Fortunately, in the input text one can avoid writing the special characters tediously everywhere: the **.tr** (translate) directive can be used to substitute special characters for ordinary ASCII characters. Thus, if one's terminal has the same national character set as the intended printing device, one can type even the non-ASCII characters as themselves when preparing the input text. An exception occurs because the escape character of Ditroff is the backslash, whose code is one of those officially reserved by ISO for national variants. For instance, in Finland and Sweden '\' is replaced by 'Ö'. Ditroff itself does allow the escape character to be redefined, but all standard macro packages count on its being the standard backslash.

The set of standard special characters defined in [AT&T2] is much too limited. For instance, it does not include any non-English letters except the Greek alphabet. This omission has the effect that Ditroff source text that contains such special characters is not portable between installations. Because the text of this paper was sent by electronic mail to the editor as Ditroff source text, I had to use one of the predefined strings of the **mm** macro package to obtain the letter 'Ö' in the previous paragraph — with rather inferior quality. In my own installation, I could have used the appropriate special character to print the PostScript 'Ö' with typographic quality. The same goes for the 'ä' letters in the heading of this article. In section 1.3, I had to omit the letters Thorn and Eth, so those readers who were unfamiliar with them before will still not know what they look like.

The alternative way for an installation to offer a national character set is to duplicate fonts — this seems to be a common solution, though not entirely attractive. For instance, many current PostScript printers have 33 different "standard text" fonts (and the number will increase with each PostScript release and printer generation). Even if just one national variant of each is needed in addition to ASCII, installations have trouble managing all of them and allocating Ditroff names to them. If one can remain in the same national character set in the whole of a document, this solution is adequate for the end user. If there is need e.g. to print ASCII braces and brackets, they can be defined as special characters in the national character set. However, if it is often necessary to change between national character sets, writing the input text becomes difficult. One difficulty is that always when changing temporarily to bold or italic style or another typeface, one must take care to specify the right variant of the required temporary font. Considering macro packages and trap handling, this can become complicated indeed. We will return to this question in the next chapter.

A user may run into a couple of font handling problems with the current release of Ditroff. It cannot interpret correctly font change escape sequences of the type '\f*N*' where *N* is a number, unless some font has been *explicitly* mounted on position *N* (the **me** macro package uses such sequences extensively). One may also get the error message 'Font *XY* too big for position *N*' when trying to mount a font; the cause can sometimes be that **biggestfont** (cf. below) has not been correctly defined when the device and font descriptions have been created.

Defining fonts

When one has to create or modify device or font descriptions, it becomes most apparent that Ditroff is not a perfectly mature and robust software product yet, although the original version

was taken into internal use at Bell Laboratories already in 1979, according to [Kern1]. There are severe limitations, not even hinted at in the manual [AT&T2], to what one can actually define. Most of these limitations are only detected when, after a seemingly successful generation of the description files, Ditroff simply crashes with a 'Segmentation fault' (or something equally informative) when trying to run with the new device. Customers with a source distribution can glean some of the limitations from various source files.

One limitation that is mentioned in the sources (and can even be changed by editing the appropriate header file) is that no more that 10 initially loaded fonts must be defined. Another is that the total number of distinct characters (that includes both ASCII and special characters) allowed in all fonts together is quite restricted. At least in the VAX source distribution, it appears to be 512. In contrast, the built-in fonts of PostScript version 47.0 already comprise over 600 different characters (counting the Zapf Dingbats™ font of fancy figures). Customers with a source distribution can extend this limit, too. But there can be at most 224 characters in a *single* font: one must wait for the next release of DWB to get this restriction (hopefully) lifted.

When installing font descriptions, one must take care that no font contains more characters (alias names are not counted) than the number **biggestfont** defined in the device description; otherwise the above-mentioned run-time error will appear. I have not tried out what happens if one tries to run Ditroff with a font exceeding the 224-character limit: whether one gets wrong characters, an error message, or a core dump.

The handling of special fonts is too primitive for some purposes. The definition classifies a font as 'special' or 'normal'. Whenever Ditroff finds that a character to be printed is missing from the current font, it searches for it in the mounted special fonts in the order of their mount positions. An explicit change to a special font is seemingly accepted by Ditroff but actually causes no effect! (There are even more subtleties with special fonts.) One should be able to specify a sequence of alternate fonts for each font separately. If this were possible, the font size limitation would not be serious.

The set of properties that can be described in the *device* description is not very complete or well thought out. For instance, it includes the list of possible point sizes, but it is not possible to tell whether the printer can perform height and slant modifications. Neither can the available drawing functions be specified. The width and length of paper, on the other hand, *are* in the device description. It is questionable if they belong there, since most printers can handle different paper sizes; at least there should be overriding options in the Ditroff command.

Input language

Everybody who has processed text with any *roff knows how difficult and awkward its commands and escape sequences can be. On the other hand, they are quite powerful. Various macro packages try to enable writing the input in terms of higher-level commands and in more structured form. Unfortunately, the effects of a macro package can interfere with those things that can only be done on the *roff level. This pertains especially to novel features of Ditroff, because many macro packages such as the Berkeley **me** are older and cannot take them into account.

Ditroff allows the slant of characters to be modified, as well as the height to be changed without affecting the width. Such operations are naturally not possible on all output devices, but on PostScript printers they are easy. For some reason, these operations are not available as ordinary commands, only as escape sequences, which can sometimes be impractical. Also, there are no predefined number registers in which Ditroff would store the current slant and height values, as it does to other similar parameters. Slant and height definitions seem to remain in force even across font and size changes.

Another novelty of Ditroff that cannot be implemented on all printers but can very easily be translated into PostScript are the drawing functions. There are escape sequences for drawing line segments, circles and circular arcs, ellipses, and B-splines. Of course, these cause very little work within Ditroff except calculating the position after the figure and passing the information to the output file.

A traditional lack in *roff capabilities that still remains in Ditroff is that it is surprisingly hard to get a desired amount of white space at the top of a page, at least on the first page of a document. Another is that one cannot specify character spaces to be increased or decreased in a similar

manner as line spacing can be adjusted with the .vs command. Such a capability would sometimes be useful for standout text. More importantly, some PostScript printers have so tight spacing in several fonts that characters touch each other; many people would like to avoid this displeasing effect. The lack is somewhat surprising since even such simple devices as Diablo® daisy-wheel printers have long offered 'increase character spacing' as a hardware operation.

Preprocessors and macro packages

Documenter's Workbench includes versions of the traditional *roff preprocessors Tbl (for tables) and Eqn/Neqn (for mathematical text and equations). A new preprocessor that can be used in connection with Ditroff only is Pic [Kern2]. It has primitives for picture elements such as line segments, boxes, arcs, ellipses, and splines. Very obviously, Pic and the new geometric facilities of Ditroff have been designed together. In release 2 of DWB, there is further a preprocessor to Pic, called Grap; its purpose is to plot time series and x-y graphs in an easy and flexible way.

The Eqn preprocessor is, unfortunately, not as device-independent as Ditroff itself: it has a command option for specifying the intended output device but it cannot use any external tables — device-specific information on the two supported printers is programmed in. Fortunately, this does not seem to matter much. An auxiliary file, **eqnchar**, is normally used in connection with Eqn to define more symbols (e.g. logical and set-theoretical connectives). There are two versions of **eqnchar**, each designed to fit one of the DWB-supported printers. Most of the symbols become bad-looking or even unrecognizable on other (e.g. PostScript) printers.

The Pic preprocessor has hard-coded device dependencies like Eqn. When drawing vertical and horizontal lines, Pic tries to exploit appropriate special characters as much as possible, in preference to Ditroff's line-drawing primitives. Since Pic does not access the font tables of a non-supported printer, the sizes and alignments of those horizontal and vertical bars are not necessarily what it excepts; lines and boxes are thus not likely to come out well. The Pic version distributed with release 1 of DWB allowed the user to specify the resolution of a non-supported output device [AT&T3]. The Pic *manual* from release 1 also mentions that there is a '-D' option that instructs Pic to draw all lines by using graphic escape sequences. Fortunately, current Pic still seems to know this option, and so the just-mentioned line alignment problem can be avoided.

As was mentioned in the previous section, people normally do not write "raw" *roff input code, but use some standard or home-brewed macro package that offers a higher level of abstraction. "Standard" should really be placed into quotes because there are so many "standard" macro packages and rather widely differing versions of each. There is only one that is included in virtually every UNIX system: **man**, which is strictly specialised to formatting traditional UNIX-style manual pages. As for general-purpose macro packages, in BSD environments there are **ms** and **me**, whereas System V sports **mm**. (DWB further includes the macro packages **mv** for view graphs and slides, and **mptx** for formatting permuted indexes.) Consequently, I have always worked with the **me** macros, but when I got the EUUG article template and directions from the editor (for this very paper, which was then already written to the greater part), they suggested using **mm**. Well, I have converted my text, and could even test it by using the version of **mm** distributed with the DWB, but let us see if the macro package at The Instruction Set is compatible or not. If I had originally used really fancy features of **me**, the conversion to another set of macros would have been very tedious if not impossible.

Exploiting a "standard" macro package can thus impede the portability of *roff source text. One's own macros will not cause this drawback, as they can always be prepended to the main text. Another kind of difficulty arises when one needs to do some operations on the *roff level: it may interfere in nonobvious ways with the workings of the macros. One must be especially careful against conflicts in the names of number registers, strings, and macros — the packages have many internal macros in addition to those that can be called directly by the user.

Middlemen between Ditroff & PostScript

Principles

As mentioned in chapter 2.1, Ditroff always needs a postprocessor for the actual printing. To be exact, there is an option to the Ditroff command that produces "a printable ASCII approximation of the results", but it is interesting only for

preview purposes. Because the output language is simple, it is true to a large extent what [Kern1] states:

> "... it is straightforward to write a prototype driver for a particular typesetter, especially when one can steal an existing one as a model, although it may take some effort to make one of production quality."

Still, some qualification is necessary for this statement. Putting up the font description tables, which are normally used by both Ditroff and the postprocessor, can be a bigger task than writing the driver / postprocessor itself.

If the model of the printing process employed by a printing device is very different from that of Ditroff, it can be more difficult to write a good postprocessor, or perhaps even impossible to exploit all capabilities of the printer. For instance, this is true about the national character sets. I have written, for a non-PostScript printer, a rudimentary postprocessor which understands some special characters (with zero width) as instructions to change to different national character sets, but this is possible only because the fonts of the printer concerned are of fixed width (Ditroff does not know of the change when looking up the characters' widths). Another difficulty can arise from a too large character set in the printer. The same kind of trickery — using some special characters to switch from one part of the character set to another — could be imagined even here if there are enough pairs (triples, quadruples, ...) of characters of the same sizes so that they can be mapped on the same Ditroff characters by pairs (triples, quadruples, ...). It would certainly not be easy. If there are only more characters *in one font* than Ditroff can support, then one must define the printer font as two or more Ditroff fonts. This can make it very unconvenient to use the font in practice.

Writing a Ditroff-to-PostScript translator cannot be very easy for the reason that PostScript is a complex language and difficult to debug. Furthermore, the postprocessor alone is not enough: at least the character width tables of all fonts are absolutely necessary for Ditroff to work. Therefore, one is not very tempted to "do it oneself" but rather to buy a commercial postprocessor. Normally one will get some useful goodies on the bargain, in addition to the absolute necessities.

PostScript documents have not necessarily the same kind of page structure as Ditroff output: in the general case, they are programmes rather than "documents". Nevertheless, for the overwhelming majority of PostScript programmes that are straight documents, there is a page-structuring convention recommended in [Adob2]. Postprocessors usually follow that convention. It enables printing filters and utility programmes to select pages from a document, to print in reverse order if the sheets will stack in right order that way, to find out quickly all fonts required by a document, etc.

The necessity of explicit spacing information between characters in Ditroff output (section 2.2) looks somewhat silly especially when the output is converted to PostScript: within each word, all explicit spacing disappears again in normal cases. Thus, a source text word 'Computer' may appear in Ditroff output as 'cC72o50m89p57u61t39e50r', but in PostScript again as '(Computer)'.

TranScript

TranScript™ is a PostScript support software package for UNIX systems originated by Adobe Systems Inc., the developers of PostScript itself. The following discussion pertains to the version adopted for Sun workstations and supported by Sun Microsystems, release 2.0 [Sun1]. The package contains filters that convert plain text, Ditroff and old Troff output, Sun raster files, and several other device-specific output formats, as well as **plot** output, to PostScript. A great number of PostScript programmes comes with TranScript, including all "cookbook" examples of [Adob1]. However, we will concentrate on things that have to do with Ditroff.

Some of the positive sides of TranScript are the following: It has a good collection of print filters, which can also collect any output coming back from the PostScript printer to a log file. Even otherwise, the standard spooling system as such would not be able to handle PostScript communication intelligently. Most of the previously mentioned conversions can be effected simply by the appropriate **lpr** options, which is especially handy for remote jobs. The actual Ditroff postprocessor has options to orientate the output horizontally and to set the printer in manual feed mode for the duration of a job that needs special stationery. It is possible for a knowledgeable user to embed PostScript code within Ditroff input. The distribution includes

complete AFM files of all supported fonts.

Some of the negative sides of TranScript, according to personal experience, are as follows: At least here in Finland, the package was very expensive (even the one-CPU binary licence). It does not support the newer PostScript fonts, only the Times, Helvetica™, and Courier families and Symbol. Naturally, it does not know about the extended character set of the current PostScript versions. The truly *catastrophic defect*, however, is that there is no possibility to use Scandinavian letters from Ditroff (after several months' waiting, we got support from the Sun representative for printing *plain text* in the Finnish national character set). Another part of this "U.S. only" attitude is that it is not possible to specify paper sizes, the American parochial standard size is silently assumed. This can be fixed somehow for Ditroff by editing the appropriate PostScript prologue file (anything that looks like 11 or 8.5 inches there has a high probability to mean paper length or width, respectively). With the **enscript** programme that converts plain text to PostScript, the user can do nothing because the paper size is programmed in.

Devps

Devps™ [Pipe] is a PostScript support package developed by Pipeline Associates, Inc. Its European representative is Penetron Oy, who have added national character set support to their distribution. They have also added a programme to generate Ditroff device and font description files, because that is not included in the *binary* DWB distribution kit from AT&T. Our department has used Devps mainly to fill the "holes" in TranScript, but the reader should note that it lacks some sophisticated features that TranScript does have, in turn.

Some aspects in which Devps is better than TranScript follow: It is available at a reasonable price, even as multi-CPU source licence. It supports the Palatino™ font family and includes some downloadable fonts (outline and Hershey fonts); there is also advice on how to make desired downloaded fonts resident beyond jobs. It includes PostScript programmes that can e.g. extract font metric information from *the printer*, albeit slowly, either in AFM format or ready for Ditroff font description. The documentation is more complete. Most important to us are the national character sets; they are implemented as different fonts so that there is a U.S. Times-Roman and a Finnish Times-Roman, etc.

Some aspects in which Devps is clearly worse than TranScript are: There are conversion programmes for Ditroff output and plain text only, the latter being much simpler than the corresponding filter in TranScript. There is no print filter that takes care of any output back from the printer (some tricks are suggested in the manual for that purpose; their success may depend on the particular UNIX version). No AFM files are included in the distribution.

There are many other support software packages similar to TranScript and Devps on the market, e.g. the third-party software catalogue [Sun2] lists several of them. They are often bundled with DWB, which is practical for the customer. I have no experience on any others than the two packages just described.

Some own work

The combination of an input-logging print filter from TranScript and information-extracting PostScript programmes from Devps encouraged me to some investigations and further developments of my own. In the beginning, making even trivial-looking modifications to PostScript programmes tended to make them crash. PostScript is really difficult to use as a real programming language, mainly because one must take care of the stack being always in the right state for each operation. Nevertheless, I got listed the names of *all* characters (the unencoded ones included) in each built-in font from its **CharStrings** dictionary and saw that there were so many new characters unmentioned in any documentation (cf. section 1.2).

The next step was modifying the PostScript programme that constructed metric information, to handle more characters. Ditroff names had to be invented for the new characters. Also, the PostScript programme that changed the encoding for Scandinavian fonts (in the printer) was enhanced so that almost every code between 128 and 255 is now mapped to some PostScript character. I finally got rid from using alternate (U.S.A./Finland) fonts — the 7-bit codes now have their Finnish interpretation and ASCII characters such as '[' and '\' have special character names. There was quite a lot of work before all device and font descriptions for Ditroff were in order. Fortunately, we have a source distribution of DWB: Ditroff and its auxiliary **makedev** programme had to be rebuilt with several array sizes enlarged (cf. section 2.4).

These developments have been made around the postprocessor of Devps, for which there was already a good beginning from Penetron. One would have needed still much more work for the postprocessor of TranScript; I studied that possibility first and fairly extensively. A problem that has surfaced later is that some kinds of complicated input (e.g. as produced by **eqn**) can crash Devps on the Sun, but fortunately not on the VAX. The complete work flow in document formatting at our department is thus funny: Ditroff is normally run on the machine each particular user happens to have his/her session on, either Sun or VAX; its output must in some cases be sent from the Sun to the VAX for postprocessing by Devps (i.e. converting to PostScript), but in most cases Devps could be run on either computer; the PostScript document is always printed on the Sun (to which the printer is physically connected), using the filter from TranScript.

Conclusion

Both device-independent Troff and PostScript (printers) are useful tools, but presently their cooperation is not as smooth as could be hoped. You can expect to utilise them together without a lot of initial work and involvement on your side mainly if you need only English letters and a limited set of special characters in your texts. However, this depends on the Ditroff-PostScript support software. When selecting support software, check very carefully that it really has all the features your application needs — compare at least two or three packages. You cannot exploit many of the fancy capabilities of PostScript (that can sometimes be stressed in printer advertisements) in combination with Ditroff. And, unless your UNIX is System V or very well compatible with it, you may not be able to install Ditroff at all, not to speak about other components of the Documenter's Workbench. The portability of Ditroff source text is problematic.

It is really to be hoped that a new release of DWB would appear soon, be a production-quality tool, and remove at least the most irritating limitations of the current release. As for PostScript, some obvious improvements would be welcome in order to maintain its position as a leading page definition language. The updating of available PostScript documentation to describe the newest releases of the language would be really urgent, of course.

Acknowledgements

This work was supported by the Ministry of Education (doctoral programme in information technology) and the Academy of Finland.

UNIX and Documenter's Workbench are trademarks of AT&T.
PostScript and TranScript are trademarks of Adobe Systems Inc.
IBM is a registered trademark of International Business Machines Corp.
DEC and VAX are trademarks of Digital Equipment Corp.
Sun Workstation is a trademark of Sun Microsystems, Inc.
Apple is a trademark of Apple Computer.
Times, Palatino, and Helvetica are trademarks of Allied Corp.
ITC Zapf Dingbats is a trademark of International Typeface Corp.
Devps is a trademark of Pipeline Associates, Inc.
Diablo is a registered trademark of Xerox Corp.

References

[Adob1] Adobe Systems Incorporated, PostScript Language Tutorial and Cookbook, Addison-Wesley 1985.

[Adob2] Adobe Systems Incorporated, PostScript Language Reference Manual, Addison-Wesley 1985.

[Adob3] PostScript Language Supplement for the Qume ScripTEN (Version 47.0 Revision 1), Adobe Systems Incorporated 1987 *(Reorder no. 32147)*.

[AT&T1] UNIX System V Documenter's Workbench Software Release 2.0 User's Guide, AT&T 1986 *(310-004 Issue 1)*.

[AT&T2] UNIX System V Documenter's Workbench Software Release 2.0 Technical Discussion and Reference Manual, AT&T 1986 *(310-005 Issue 1)*.

[AT&T3] UNIX System V Documenter's Workbench Software 1.0 Preprocessor Reference, AT&T 1984 *(307-153 Issue 2)*.

[Earn] Rae A. Earnshaw (ed.), Workstations and Publication Systems, Springer-Verlag 1987.

[Holz] David A. Holzgang, Understanding PostScript Programming, SYBEX (San

Francisco) 1987.

[Kern1] Brian W. Kernighan, A Typesetter-independent TROFF, *Computing Science Technical Report No. 97*, Bell Laboratories 1982.

[Kern2] Brian W. Kernighan, PIC — A Graphics Language for Typesetting, Revised User Manual, *Computing Science Technical Report No. 116*, AT&T Bell Laboratories 1984.

[Pipe] Devps™ User's Guide *(and associated manual pages)*, Pipeline Associates, Inc.

[Qume] ScripTEN Guide To Operations : Printer Installation and Operation, Qume Corp. 1987 *(Reorder no. 32206)*.

[Sun1] Installing the Sun™ LaserWriter *(and associated manual pages)*, Sun Microsystems, Inc. 1986 *(Part No. 800-1249-04, Rev. A)*.

[Sun2] Catalyst, January 1988, Sun Microsystems, Inc. 1987.

Warning: The reports [Kern1] and [Kern2] can be hard to obtain and very expensive in proportion to their extents of 13 and 26 pages respectively.

Note: See chapter 1.6 on the different versions of [Adob2].

Network GmbH events in September

27 - 29 September
ED '88
Electronic Displays - Exhibition and Conference
Wiesbaden, Germany
Rhein-Main_halle

27 - 29 September
Unix in Deutcheland '88
Annual meeting of the German Unix Users' Group (GUUG)
Hardware and Software Exhibition
Hannover, Germany
Congress Centrum Stadtpark

For more information contact:

Network GmbH
Wihelm-Suhr-Straβe 14
D-3055 Hagenburg
+49 5033 7057

News from the Netherlands

Frances M.T. Brazier
frances@psy.vu.nl
vupsy!frances

Department of Cognitive Psychology
Vrije Universiteit
Amsterdam

Frances is the secretary of the board of the NLUUG, and is their representative in the EUUG's Governing Board.

Conferences

Spring Conference

Our last conference in May was on Integrated Project Support Environments - IPSEs. As our experience with poster sessions at the spring conference a year before had been quite positive, they were again held at this conference. In contrast to our autumn conference, the spring conference is not adorned with an exhibition.

The Board of the NLUUG experimented with a slightly different form of organisation for this conference. Instead of appointing one member of the Board to be responsible for the programme, a programme committee was formed in which three members of the Board and one member of the NLUUG (with more knowledge of the subject) participated. This formula worked to our satisfaction and will therefore be followed in the future.

The following presentations were given in May (if more detailed information is required addresses are available):

— Introduction to IPSEs
Phil Mair, The National Computing Center Ltd, Manchester

In the opening talk, an overview of the architecture and levels of integration of IPSES was provided. The basis for discussions on developments within this area was thus formed: the Alvey Committee's proposal was discussed at length, the two current standards initiatives PCTE and CAIS were placed in perspective, and current implementations of commercial-DP IPSES and real-time IPSEs were considered.

— Object oriented design with workstation-based tools and the importance of the EDIF standard
Jelle Visser, Pierre Schmit, Koning en Hartman Elektrotechniek, Delft, Cadre Technologies, Switzerland.

The possibilities available for the implementation of an ADA developmental environment in all phases of a project, were discussed. The first step taken in this direction is the design and implementation of a Buhr

diagram-editor. Further development of tools as such, however, requires specification of the interface. A uniform interface standard for the exchange of data between CASE, CAE and ATE environments is necessary. The EDIF standard is an attempt to bridge the gap.

— IPSEs: background information, categories and expectations
Frans Ververs, Technical University, Delft

The problems encountered in this field of research were introduced on the basis of the terminology employed. The possible and plausible combinations of the terms: integrated, development, software, programming, system, engineering and environment employed, do not necessarily provide the most efficient categorisation of the systems currently in development. The types of users for which such systems are designed is more likely to succeed on this count. The advantages of UNIX and the current state of art were discussed. Optimistic views of the situation in both the near (1995) and the far future (2nd generation IPSEs), were presented and discussed.

— The Network Software Environment - NSE
Dick Mansvelder, Sun Microsystems Nederland BV, Soest

NSE is an environment designed for the support of large software development projects for the management of parallel development in heterogeneous networks. The major concepts employed in this system and a few of the problems encountered were addressed. Naming and management facilities for all types of objects were mentioned, as were facilities for hierarchical distribution and configuration of objects during development.

— PIMS - Project Integrated Management System
Dick Biekart, BSO, Eindhoven

The first prototype of PIMS, a system that is being developed within the EEG ESPRIT programme, has been implemented in a SUN-3 environment in Common Lisp and Flavors. PIMS is an integrated set of tools designed to support the management of software projects. Tools for task definition, planning resource management, tracking, reporting and risk analysis have been implemented, all with the same interface and all within the same object oriented project model. Each tool presents a different view of the model, depending on the relevancy of the information required. The object oriented information manager is not only responsible for the the model itself but also for the knowledge base (with data on project management, and parts of the user interface. The tools communicate with these modules via the information manager. The information manager is furthermore responsible for communication between the tools themselves.

— Computer aided development and maintenance of complex systems using EPOS
Peter Göhner, Gesselchaft für Prozesrechnerprogrammierung mbH, Germany

The fundamentals of this engineering and project management oriented support system were discussed in detail. The seven interactive modules: Requirements, System Design, Project Specification, Analysis, Management, Documentation and Communication were described separately, but also in connection with the rest of the system. The advantages of such systems (consistent, complete and unambiguous system or software requirements documents, well-structured, well-documented software design and code with very low residual error levels, time effective project management, etc) were discussed on the basis of this quite extensive system.

— Project Support Environments, Integrated?
Dik Fikker, Fysisch en Elektronisch Lab/TNO, Den Haag

The FEL is involved in a number of international, (some military) initiatives concerning the development of PSEs. An overview of the initiatives, goals and purposes of the international projects was presented. The need for integration was again emphasised; the problems involved were discussed on the basis of a number of models (APSE, Special ER-model). The insight thus provided was strengthened by an explanation of PSE, the proposed model for the implementation of PTIs (interface sets). The success of integration was, however, as yet not achieved. This model was not sufficient for an IPSE: the syntax has been defined (in PTIs), but the semantics are still missing.

— UNIX and RTEE: the Real-Time Engineering Environment
M.J.A. Steltman, Westmount Technology, Delft

The title suggests a broader topic than was discussed. The RHTX debug environment, a component of the RTEE, was the topic actually addressed. RHTX is a host-target development system which enables programs to be developed and generated on a host-workstation and debugged and tested on a real-time target microprocessor system connected to the host. The architecture of the system was described in detail.

Autumn Conference

Pre-announcement

NLUUG Autumn Conference 1988
UNIX and Standardisation
November 10, 1988 at De Reehorst
Ede, The Netherlands

Unix and standardisation
With the increasing importance of UNIX for a large number of applications and for diverse areas of data processing, the need for the definition of formal standards has increased. Effort is currently being spent, by various organisations, to produce UNIX-related standards.

For instance:

Portable Operating System Interface
- POSIX (IEEE 1003)
System V Interface Definition
- SVID (AT&T)
Common Applications Environment
- X/Open
C Language
- ANSI X3J11
Systems Application Architecture
- (IBM)

In addition, standardisation activities in the areas of communication and window management, of great importance to UNIX users, should not go unmentioned, for example:

ISO OSI	TCP/IP
NeWs	X-Windows

Not all developments follow standard specifications: universities develop their own "standards".

The Dutch Unix Users Group (NLUUG) is organising a one-day conference on November 10, 1988, on the above mentioned standardisation activities. The programme will include technical sessions, a number of tutorial sessions and an exhibition of UNIX hardware and software products.

All EUUG members are more than welcome.

AFUU News

Philippe Dax
dax@enst.enst.fr

Ecole Nationale Supérieur des Telecoms

Philippe Dax has been the editor of Tribunix (the French Unix Newsletter) since 1985. He discovered Unix V7 on an Onyx in 1981. He was part of the bunch that created the AFUU in 1982. Philippe also wrote a book on C in 1983.

Philippe is an engineer in the Computer Centre at ENST (Ecole Nationale Supérieur des Telecoms), where he struggles with "Unity" under VMS in the spirit of Unix philosophy, e-mail, and daily clears our spool directories. He is waiting for Godot (native Unix workstations)!

Chers adhérents,

Après l'annonce de la constitution de l'OSF, ``l'Open Software Foundation'', par IBM, DEC, HP, Apollo, Bull, Siemens et Nixdorf, de nombreux adhérents ont demandé á l'AFUU quelle était sa position. De plus une discussion s'est engagée sur le réseau Eunet pour susciter commentaires et réactions.

L'actualité exige de commenter ce mouvement. Certains ont parlé du partage de la Planète, de guerre ouverte entre deux géants, de ''Yalta de l'informatique'', de torpillage de la standardisation, bref, de quoi se régaler dans le choix des titres á sensation. D'autres préfèrent ne pas trop alimenter le débat, pensant que cela ferait une mauvaise publicité á UNIX.

A mon sens (je ne parle pas au nom de l'AFUU mais exprime ici mon avis personnel), toute cette histoire démontre qu'UNIX est devenu le véritable enjeu des années 90 jusqu'á l'horizon de l'an 2000 et probablement au-delá (sous une autre forme).

Dear members,

After the announcement of the creation of the Open Software Foundation (OSF), by IBM, DEC, HP, Apollo, Bull, Siemens and Nixdorf, many members have asked the AFUU what its position is on this development. There has also been a discussion on the network (EUnet), trying to find out what are people's attitudes.

Certainly, some comment is needed. Some have spoken of 'Partitioning the planet', of 'Open war between two giants', of 'The Yalta of the computing world', 'Torpedoing of the standardisation efforts'. Ohers prefer not to engage in open debate, believing that it can do nothing but damage the reputation of UNIX.

My personal opinion (ie, that of Philippe Dax, not the AFUU) is that this event simply shows that UNIX has really become the real 'game' in the computing world, the most significant OS for the '90's and into the 21st century.

Enfin!. les "Grands" ont choisi UNIX comme la solution la plus appropriée par rapport á la pression du monde des utilisateurs. Les utilisateurs, c'est eux, qui, á un certain degré, ont engendré inconsciemment cet état de fait. C'est eux, qui, au cours des années 80 ont manifesté leur "ras le bol" vis-á-vis des constructeurs qui se croyaient toujours en terrain conquis ou en situation de monopole. C'est eux, qui, confrontés á une informatique de plus en plus hétérogène, ont poussé á la standardisation. C'est eux qui ont découvert UNIX comme le seul système permettant la portabilité des applications sur tout un éventail quasi-mondial de machines. C'est eux qui dictent d'une certaine manière quelle direction doivent prendre les constructeurs. Ce sont, enfin, les constructeurs qui l'ont compris et tentent maintenant d'anticiper le mouvement des utilisateurs.

Si UNIX, comme cela a été annoncé, atteindra près de 25"%" du marché mondial vers les années 92-94, il est tout á fait normal que le gâteau soit convoité, tout comme celui du monde PC l'a été avec les compatibles asiatiques.

Ceux qui croyaient en UNIX sont probablement effrayés ou agassés par ce tapage, mais qu'ils se réjouissent car leur cause a été entendue. Ceux qui n'y croyaient pas, ou ne voulaient pas y croire, des noms!!!, sont maintenant contraints d'y croire.

Maintenant..., que sera l'UNIX des années 90? Sera-t-il OSF? ATT-SUN? Sera-t-il ouvert? Sera-t-il unifié? Lá est tout le débat, mais une chose est sûre, il sera POSIX. De toute manière, les analyses, les commentaires, les prises de position ne manqueront pas dans les mois qui viennent, et á ce propos, lors des "Journées UNIX de Grenoble" jumellées avec la "Convention AFUU 88" les 17 et 18 novembre prochains, vous en saurez plus après les exposés qui traiteront sur ce sujet (voir dans ce numéro "Prochaines manifestations").

At last!. The 'giants' have chosen UNIX as the most appropriate solution, following the pressure generated by users. 'THE USERS', are the people who have, only half conciously maybe, shown during the course of the 1980's that they have had enough of manufacturers wanting to do except conquor teritory, and hold it in a monopolistic fashion. It is they, who, confronted by a computing industry which becomes more and more hetrogenous, have pushed for standardisation. It is they who have discovered UNIX as the only system giving portability of their applications onto a world-wide set of machines. It is they who dictate (in a certain fashion) which direction the manufacturers must take. These (the members of OSF) are the manufacturers who have finaly understood, and are trying to anticipate the wishes of 'the users'.

If UNIX, as has been predicted, takes 25% of the world computing market in 1992-94, it is completly normal that the 'cake' will have to be shared, as was the case with the IBM PC, and the Asiatic compatibles.

Those who believe in UNIX are probably worried by this 'incursion', but they should rejoice because their cause has been enlarged. Those who don't believe this, or who didn't want to believe in this cause will now be forced to believe!

Now ... where will UNIX be in the '90s? Will it be OSF?, AT&T/SUN? Will it be 'open'? will it be unified? There is the debate, but one thing is certain, it will be POSIX. In any event, the analysis, the comments, the taking of sides will not be lacking in the comming months. Apropos, during the UNIX days, at Grenoble twined with the "Convention AFUU 88" on the 17th and 18th of November, you can learn more about this subject, as several exhibitors propose to examine this subject.

Rappelons ici, que le rôle de l'AFUU est de promouvoir UNIX sous toutes ses formes et ses tendances, ce qu'elle a déjá réalisée dans son passé á propos des UNIX : Version 7, System 3, Xenix, BSD 4.? et System V; des Stantards : /usr/group, SVID, X/Open, IEEE, POSIX. L'AFUU n'a jamais pris parti ni pour l'un ni pour l'autre, et entre la bataille OSF - ATT/SUN, son attitude sera d'informer les adhérents, de susciter des échanges d'opinions et d'organiser des débats. Ces colonnes vous sont ouvertes...

Remember, the role of the AFUU is to promote UNIX, in ALL its forms and tendancies. This it had done in the past with the different versions of UNIX : V7, S3, Xenix, BSD4.?, SV; and the standards: /usr/group, SVID, X/OPEN, IEEE/POSIX. The AFUU has never taken sides, and in the battle OSF - AT&T/SUN its attitude will be simply to inform all its members of what is happening, to solicit exchanges of opinion, to organise debates etc - it is YOU, the members who must take the final decisions. The Newsletter columns are always open to you...

POSIX Standardisation Developments

Cornelia Boldyreff
corn@cs.brunel.ac.uk

UK POSIX Panel Convenor
Department of Computer Science
Brunel University
Uxbridge UB8 3PH
England

Cornelia Boldyreff is a member of the British Standards Institution technical committee on Application Systems, Environments and Programming Languages. She acts as Convenor and Chairman of the BSI C Language Panel; and is one of the UK Principal Experts on the ISO Working Group on C. She is also Convenor and Chairman of the BSI POSIX Panel; and is one of the UK Principal Experts on the ISO Working Group on POSIX.

Recent Developments

The recent ballot on the IEEE POSIX 1003.1 which closed in July was successful with over 85% in favour. The plan is to bring out a final revision, draft 12.4, which will be published at the beginning of September; and seek IEEE Standards Board approval of this draft when the board next meets in October 1988. It was agreed at the March meeting of the ISO Working Group on POSIX that this revised document adopted by the IEEE would be registered as the Draft International Standard. So by the end of 1988, we can expect to see an approved IEEE Standard for POSIX P1003.1; and a registered DIS. Approval as an ANSI standard and a full ISO standard should follow in the coming year. The ISO DIS will be subject to six month ballot; at present, there are no outstanding unresolved issues likely to hold up progression to an ISO standard.

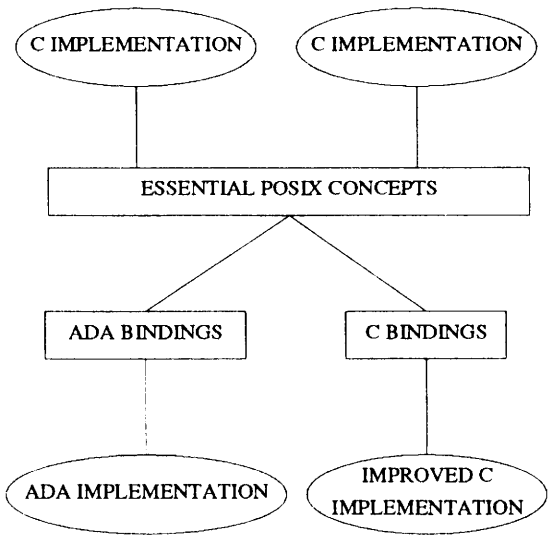
An issue which has been much discussed in the past few months has been the options allowed by the present draft. Early in the year, the US National Bureau of Standards brought out an interim FIPS which in effect removed options. This move and concern from the ISO community regards the undesirability of options has resulted in the elimination of most options from the standard; those remaining are as follows: the multiple groups option, the job control option and the administrative/security option which restricts *chown*. A more precise term in place of options to describe this aspect of the standard is alternatives; thus, for example, if the symbol, _POSIX_JOB_CONTROL is defined, it indicates that job control is supported and otherwise, that it is not.

Future Work

At the ISO WG15 meeting, it was agreed to initiate a division of the work item recognising that P1003.1 is simply the platform for a family of related standards. The next version of POSIX 1003.1 is committed to being a language independent definition of the interfaces currently defined in terms of the C language. Work within the IEEE developing an Ada binding for POSIX

has already begun to address this issue: and there is much interest within the ISO group in progressing this work.

The process of abstracting the essential POSIX concepts from their various implementations which underlies this work on language bindings is bound to benefit the users of the UNIX family of operating systems by resulting in a better conceptual understanding and subsequently better implementations.



Recent and Future Meetings:

27 Jun -1 Jul	P1003	Denver, CO
7 Sept	POSIX Panel	BSI, London
20-21 Oct	WG15	Tokyo
24-28 Oct	P1003	Hawaii
9 Nov	POSIX Panel	BSI, London
9-13 Jan 89	P1003	Fort Lauderdale, FL
Apr 89	WG15	Ottowa, Canada
17-21 Apr 89	P1003	St. Paul, MN
Jul 89	P1003	Monterey, CA
Oct 89	WG15	Brussels
Oct 89	P1003	Brussels
Jan 90	P1003	New Orleans, LA
Apr 90	WG15	Paris
Oct 90	WG15	USA

British Standards Institute - Ada Evaluation Reports

These are now becoming available. BSI tell the EUUGN that the reports point out the strengths and weaknesses of compilers enabling selection of the compiler best suited to a project, and allow more effective use of the chosen compiler.

You can either buy one copy at £250 (+ VAT), or by using their subscription service you can have all the reports produced in one year for £2000.

For more details please contact:

Terry Calvert
BSI Marketing
PO Box 375
Milton Keynes
UK MK14 6LL
+44 908 220908 Extn 2094

Draft Proposed ANSI/ISO C Standard: Developments

Cornelia Boldyreff
corn@cs.brunel.ac.uk

UK C Panel Convenor

Recent Meetings

The ISO Working Group Meeting

ISO Progress

The London meeting of WG14 held from the 13-14 June 1988 was attended by representatives from Denmark, the Netherlands, the UK and the USA. At this meeting, Bill Plauger, the ISO Convenor, tabled the X3J11 Response Document to comments from the second public review; this document included a large section devoted to the comments submitted by the ISO community.

Bill Plauger agreed to ensure ISO comments would be processed at the September meeting of X3J11. At present, the UK has no outstanding unresolved comments which would require other than editorial changes to the draft. However, WG14 was unable to resolve the Danish NO vote in the recent ISO DP ballot. Resolution of this requires a further substantive change to the ANSI draft and would require a further public review before ANSI approval could be sought. The crux of the Danish objection is that the trigraphs for { } and [] are unusable; and that an alternative is required.

It was agreed by WG14 not to submit any further draft for ISO ballots until a stable X3J11 document was available. If following the third review, a new draft is voted in by X3J11 at their September meeting and this draft is submitted to ANSI for approval, then WG14 agreed to either submit this draft to ISO for registration as a DIS if it is acceptable to Denmark and the UK, or to resubmit it for a further DP ballot.

X3J11 Progress

At their April meeting, X3J11 processed comments from the second public review; and made a number of substantive changes to the draft standard. The most controversial of these was dropping *noalias*. As a result, a third public review of three months is required. This review is currently in progress. If you wish to comment on the May 13, 1988 draft under review, you should obtain a copy via your ISO representative or from Global Engineering at the address below:

Global Engineering Documents, Inc
by calling +1 800 854 7179 or +1 714 540 9870.
Expected Single Copy Price US$65.00 (draft standard and rationale).

Global Engineering is located in Santa Ana, California, USA, which is in the Pacific Standard Time (PST) zone.

Future Meetings and Projected Targets

Any comments on the X3J11 May 13, 1988 draft will be processed at X3J11's September meeting. If following the third public review, no substantive changes have been made to the draft by X3J11 in September, it will go forward for administrative processing by ANSI; and emerge as an ANSI C Standard in the latter part of 1988 or earlier next year. If the ANSI standard is acceptable, WG14 will put it forward for registration as a DIS; otherwise there will be another ISO ballot on the draft as a DP. There is strong support amongst the ISO community for a common standard which does not deviate from the ANSI standard.

Future Meetings:

9th Aug 1988	BSI IST/5/14	London
26-30 Sept 1988	ANSI X3J11	Cupertino, CA
8th Nov 1988	BSI IST/5/14	London
12-16 Dec 1988	ANSI X3J11	Seattle, WA
10-12 Apr 1989	ANSI X3J11	Phoenix, AZ
Dec or Mar	WG15	with X3J11

EUUG

European UNIX® systems User Group

Autumn 1988 CONFERENCE

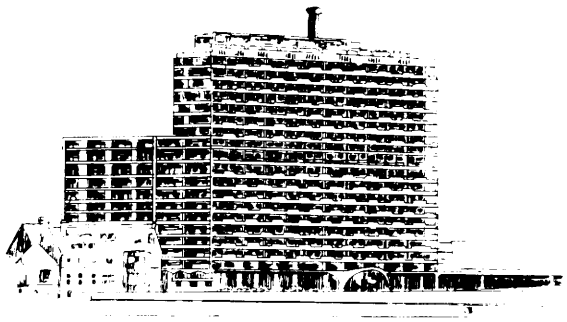
Tutorials and Showcase

New Directions for UNIX

Hotel Estoril-Sol

Cascais, Portugal

3-7 October 1988



The **EUUG Autumn '88 Conference** will be a truly European event giving a Forum for the varied and interesting work on UNIX which is currently taking place in Europe. All the lecturers will be recognised authorities on the various fields to be discussed.

The main conference is expected to cover such areas as **Real Time; Security Issues; Distributed Processing; Multi-processors and Parallelism; Supercomputing; Internationalisation; Fault Tolerance; Transaction Processing; Virtual Memory; Object Oriented Approaches; Videotext Applications; Standards and Conformance Tests.**

The conference will be complemented by two days of Tutorials, on 3rd and 4th October, which will cover the following subjects:
Users introduction to the X Window System; POSIX Implementation; and Curses (Day 1) plus Programming with the X Window System; Sendmail; and System Performance & Management (Day 2).

For further details on this important event, including information on accommodation, please contact:

European UNIX® systems User Group

Owles Hall, Buntingford, Herts SG9 9PL, UK
Tel: Royston (0763) 73039 +44 763 73039 (overseas)
Facs: Royston (0763) 73255 +44 763 73255 (overseas)
Network address: euug@inset.uucp

Cost of attending the conference will be £240 for members of EUUG and £280 for non-members if booked and paid for before 22nd August 1988. Cost does not include accommodation which must be booked and paid for separately, but it does include coffee, tea, lunch, a copy of the proceedings and attendance at the gala folk evening dinner.

Tutorials

Tutorials will be held on 3 & 4 October 1988 but will be open to members only. The cost will be £176 for one whole day tutorial including refreshments, lunch and course notes if booked and paid for before 22nd August 1988.

(Each tutorial lasts for one whole day and will start at 09.30)

The Tutorials will be:

Monday 3rd	*Users introduction to the X Window System*	*POSIX Implementation*	*Curses*
Tuesday 4th	*Programming with the X Window System*	*Sendmail*	*System Performance & Management*

In outline they will cover:

M1

Users Introduction to the X Window System — *By Watson, Clenton & Caruthers*

This tutorial is intended for those who will be using the X Window System, and for managers of such people. It is NOT a programming tutorial of any kind; there will be no discussion of programming, libraries, toolkits, or other such topics. It may be of interest, however, to those who will be programming with the X Window System, but as yet have never worked with it. No prior experience with windowing systems is necessary or assumed. Topics covered will include:

- General concepts of windowing systems
- Networking concepts, Network transparency in X
- Structural overview of the X Window System
- Display server concepts: windows, icons, mouse, keyboard
- Window manager concepts, interacting with window managers, menus
- Two significant window managers; uwm and menuwm
- Configuring and customizing X; .Xdefaults, etc

M2

POSIX Implementation — *By Shane P. McCarron & John Quarterman*

This tutorial is about the IEEE 1003.1 portable operating interface standard, POSIX. The tutorial starts with a brief history of the standard and its relations to other standards, such as the 1984 /usr/group standard, the X/OPEN Portability Guide, and the ANSI X3.159 C Standard. It ends with implications of related POSIX draft standards, such as 1003.2 (shell necessary to implement applications that will run on a POSIX conforming implementation and be as portable as possible to other such interfaces. It includes suggestions on what to do and what to avoid in a number of troublesome areas, such as signals, pipes and FIFOs, and the terminal interface.

Attention is paid to the differences between the POSIX interface and those of System V Release 3 and 4.3 BSD. This tutorial is primarily oriented toward applications developers who want an understanding of the portability restrictions required by POSIX, and how those restrictions relate to their current programming practices. However, since the session is a summary of the differences between POSIX and current implementations, it will also be of use to system implementors who need to understand what enhancements are required for POSIX conformance.

Attendees should be experienced application writers who want to learn how to take best advantage of POSIX in writing portable programs, and interface implementors who want to know what application developers expect in their interface. Attendees should have read the POSIX standard beforehand. There is no license requirement for attendance.

Shane P. McCarron is a Systems Analyst with the Minnesota Educational Computing Corporation. He joined the 1003.1 working group in 1985, and became the secretary in September, 1986. In addition to his role in P1003.1, he is also a member of P1003.2 and ANSI X3J11.

M3

Curses — *By Prof. Axel Schreiner*

Introduction:
What are curses?
What is terminfo?
NOT: How do I write terminfo for a new terminal?

Curses-Desk-Calculator:
a program where all curses functions can be tried out interactively — participants are welcome to sources
Effects of the various functions (families) are shown using this program.

Window-Shell:
a program which manages several windows on one terminal screen and permits switching between them — participants are welcome to sources. Windows can be opened and closed, moved, and resized. Inside a window, one can use a C desk calculator, look at files or command output, run an interactive program such as a shell, full-screen edit, a file, or look at on-line help. The window-shell illustrates very many typical problem settings for screen-oriented applications and offers solutions that can easily be extracted. All curses functions dealing with a single terminal are used in the window-shell.

The tutorial will concentrate on building the window-shell and dealing with all curses — and system-call-related problems. Knowledge of C is essential, knowledge of lex and yacc would be helpful but is not mandatory.

T4

Programming with the X Window System — *By Watson, Clenton & Caruthers*

This tutorial is intended for programmers who will be developing applications or utilities based on the X Window System. It is assumed that attendees will be competent programmers, and

will have some fundamental knowledge of windowing systems in general, and preferably of the X Window System in particular. Those without such experience would benefit greatly from attending the previous class before attending this one. Topics covered will include:

— *Basic programming concepts of the X Window System*
- Coordinate systems
- The byte stream data connection
- Conceptual objects used in X
- X data types, structures

— *Xlib interface*
- Naming conventions, argument passing and return
- Detailed examination of common routines

— *Toolkits*
- Xtk, the "standard" toolkit
- Xray, a very popular alternative
- InterViews, object oriented toolkit in C + +

T5

Sendmail — *By Forrest Smoot Carl — Mitchell, Managing Partner of Texas Internet Consulting*

I. Overview

A. Function: 1. Routing; 2. Address Rewriting; 3. Queuing; 4. Mail delivery; 5. Aliasing.

B. Architecture: 1. Sendmail program; a. Queue daemon; b. SMTP daemon; c. Deliverer.

2. Support files and directories: a. sendmail.cf & sendmail.fc; b. mail queue; c. aliases; d. sendmail.hf; e. sendmail.st; f. booting; g. forwarding mail

3. Support programs and functions: a. new aliases; b. mailq; c. syslog

4. Typical mail deliverers: a. /bin/mail; b. /usr/bin/uux; c. SMTP delivery

C. Typical installations: 1. Stand alone; 2. Simple UUCP site; 3. Simple ARPA Internet site; 4. Complex mail relay

II. Configuration file
A. Format and syntex; B. Boilerplate stuff; C. Address and header rewriting: 1. How rewriting works; 2. The macro language: a. Special macros and classes; b. User defined macros and classes; 3. Rewriting rules and rulesets; 4. Special rulesets: a. 0 (zero) — find the right mailer; b. 3 — canonical form; c. 1, 2 & 4 — header rewriting; 5. Mailer configurations: a..local mailer; b. SMTP mailer; c. UUCP mailer

III. Hints and tidbits
A. Sendmail and the ARPA Internet namesaver: 1. Canonical names and aliasing problems; 2. Being sure headers are correct; B. Cluster configurations: 1. Workstation cluster; 2. General addressing and making all machines local; 3. Sharing aliases in a LAN environment; C. Setting up mail relays and gateways: 1. Small sites; 2. Single host; 3. Large complex sites; D. Tuning hints: 1. Queueing versus direct delivery; 2. Delivery modes; 3. File protections. E. mailer options and which ones are useful; F. Debugging hints: 1. using -v; 2. using -d; 3. testing configurations; 4. finding the value of a built-in macro; G. Special headers and aliases: 1. MAILER-DAEMON; 2. Apparently-from; 3 Received from lines and debugging

T6

System Performance & Management —
By Brian E.J. Clark, Pyramid Technology

This tutorial concentrates on two areas of interest to the managers of large UNIX based systems.

As UNIX gains credibility as a "standard" delivery vehicle for a wide range of different applications environments, the ability to assess and manage the systems performance has become increasingly important.

This tutorial deals with both performance measurement and systems management. It will carry out a full review of the performance measurement tools available as part of the normal UNIX tool set, and also examine the products becoming available within the commercial marketplace.

Some time will be spent examining the output from these tools and examining how they relate to the load on the running system and its hardware configuration.

As the power of systems increases, so do the problems of system management. Terms such as mainframe and supermini become increasingly confused as superminis find their way into mainframe environments.

UNIX alone offers very little in the way of systems management tools, relying on the skill of the System Administrator to devise the necessary tools out of the UNIX toolkit.

As UNIX permeates the commercial marketplace this approach becomes less viable and so there is a range of tools emerging to answer this need. The second part of this tutorial will concentrate on these tools, why they are needed, and how they can be used to provide the management facilities needed for the commercial systems manager.

Accommodation

Accommodation may be booked direct with the five star ESTORIL-SOL, where the conference is being held, by using the booking slip provided. Price for each person is as follows:

Please note that the hotel requires a deposit at the time of booking.

HOTEL ESTORIL SOL

Double room each person per day	£37
Single room each person per day	£65

The Secretariat can also put delegates in touch with hotels offering cheap accommodation and can suggest alternative accommodation once the above hotel is fully booked. Early booking is recommended to avoid disappointment.

Cancellations

It is regretted that no refund of fees will be possible in case of cancellation, unless the cancellation is made more than one month before the start of the Conference. No cancellation will be accepted unless it is sent to the EUUG Secretariat in writing.

Language

The official language of the Conference will be English. No translation will be provided.

Liability

The EUUG will not accept any responsibility for damage to property or injury to persons during the entire event. Participants are recommended to arrange for their own personal travel and health insurances.

Student Grants

Grants are being offered to assist students to attend the Conference. An application must be made well in advance of the Conference. A decision will be made before the event whether an application qualifies for a grant. Payment will not be made until after the Conference but the applicant will be able to proceed in the knowledge that the grant will be forthcoming.

Applications should be made on the form at the back of this booklet, together with a booking for the Conference on the other form. If your booking is dependent on obtaining the grant, please write on the top of your booking form: **"GRANT DEPENDENT".**

Priority will be given to:

1. Students giving a talk at the Conference

2. Students doing work for the EUUG or a National Group

3. Students

4. Other deserving cases like research students.

You can apply for (partial) coverage of expenses for travel in Europe, accommodation and Conference fees, but not for meals. Student status or other deserving status, must also be documented by a copy of a valid student registration card or the like. After the event, original bills must be included with the claims form. The Secretariat can help you find suitable student accommodation.

Exhibition

A showcase Exhibition will be open during the event. Companies wishing to exhibit should contact the EUUG Secretariat for full details and booking form.

How to Book

To book a place at the Tutorials and/or Conference, complete one booking form for each person and return it *with* the full remittance to:

The Secretariat	tel:	+44 763 73039
European UNIX systems User Group	fax:	+44 763 73255
Owles Hall, Buntingford	email:	euug@inset.uucp
Hertfordshire SG9 9PL, UK	or	...!mcvax!ukc!inset!euug

Use a photocopy of the booking form for each additional person. Please note that bookings can only be accepted when accompanied by payment. Telephone bookings will not be accepted. The EUUG Secretariat will acknowledge your booking by sending you a receipted invoice together with further details for registration.

All payments must be made in pounds sterling (£).

Payments may be made in one of 3 ways.

1. By UK Cheque or Bankers Draft, made payable to EUUG, and drawn on a UK bank. Eurocheques are acceptable, but each cheque must be £100 or less.

2. By Direct Payment to the EUUG's bank, which is:

The Bank of Scotland	Account Number: 00613997
61 Grassmarket	Bank Sorting Code: 80-31-50
Edinburgh	
Scotland EH1 2JF	

Please tell your bank that you will pay all charges so that EUUG will receive the full amount due.

3. By VISA, with card details appearing on the Booking Form.

NOTE: Closing date for all bookings is 26th September 1988.

Costs

EUUG national group or direct members		Cost if booked and paid for *before 22nd August*	*after 22nd August*
3 day conference	1 person	£240	£260
1 day tutorial	1 person	£176	£195
Non-members			
3 day conference	1 person	£280*	£300

* *The reduced fee also applies if 3 or more delegates apply together.*

Booking Form for Conference and Tutorials

Please complete this form and send it, with cheque or evidence of payment, to **EUUG Secretariat, Owles Hall, Buntingford, Herts SG9 9PL, UK** (Block capitals please)

Surname _____ **Usual first name** _____

Company/Organisation _____

Address _____

_____

Country _____ **Post/Zip Code** _____

Telephone/Fax/Telex/Email _____

EUUG member? Yes ☐ No ☐ Student? Yes ☐ No ☐

Please read the sections on **"COSTS"** and remember that pre-booking saves money.
All payments must be made in pounds sterling (£).

CONFERENCE
Please reserve me a 3 day place for the Technical Sessions. ... ... £ _____

TUTORIALS (members only)
Please reserve me a place for Tutorial No_____on Monday, 3rd October ... ... £ _____
Please reserve me a place for Tutorial No_____on Tuesday, 4th October ... ... £ _____

Do you require Vegetarian meals Yes ☐ No ☐

TOTAL £ _____

EUUG
Please enrol me as an institutional member of EUUG via the appropriate national group Yes ☐ No ☐

Please read the section **"HOW TO BOOK"**

PAYMENT METHOD

☐ UK Cheque, Bankers Draft or Eurocheque. The cheque must be enclosed.

☐ Direct Payment. The bank advice note showing details and date of payment must be enclosed. *All bank charges must be borne by you and not the EUUG — please tell the bank this.*

☐ by **VISA**

Name as it appears on the card (block capitals) _____

Address of cardholder _____

_____

_____

Card Account No. _____ Date of Expiry _____

Signed _____ Date _____

Student Grant Application Form

Return to **EUUG Secretariat, Owles Hall, Buntingford, Herts SG9 9PL, UK**

(Block Capitals Please)
Name .....

Address .....

.....

.....

.....

Telephone .....

EUnet address .....

Position .....

University/Organisation .....

Status Speaker: ☐ Helper: ☐ Student: ☐ Other: ☐ tick as appropriate

Student status or other deserving status must be documented by a copy of a valid student registration card or the like.

Expenses requested/reclaimed (Give best estimate where necessary):

Travel .....

Accommodation .....

Conference fee .....

Total amount .....

Date .....

Signature .....

OFFICE USE ONLY
Received EUUG .....Granted .....Amount .....

4.3BSD Manuals

The USENIX Association now kindly offers all members of the EUUG the opportunity to purchase 4.3BSD Manuals[1].

The 4.3BSD manuals sets are significantly different from the 4.2BSD edition. Changes include many additional documents, better quality of reproduction, as well as a new and extensive index. All manuals are printed in a photo-reduced 6" × 9" format with individually coloured and labelled plastic "GBC" bindings. All documents and manual pages have been freshly typeset and all manuals have "bleed tabs" and page headers and numbers to aid in the location of individual documents and manual sections.

A new Master Index has been created. It contains cross-references to all documents within the other six volumes. The index was prepared with the aid of an "intelligent" automated indexing program from Thinking Machines Corp. along with considerable human intervention from Mark Seiden. Key words, phrases, and concepts are referenced by abbreviated document name and page number.

While two of the manual sets contain three volumes, you may order complete sets only.

The manuals are available now. To order return a completed "4.3BSD Manual Reproduction Authorisation and Order Form" to the EUUG secretariat along with your remittance. You **must** be an EUUG member.

The EUUG has bulk shipped these manuals from the USA thereby saving you 5 Kg transatlantic postage.

Manual	Cost
User's Manual Set (3 Volumes) User's Reference Manual User's Supplementary Documents Master Index	£25.00/set
Programmer's Manual Set (3 Volumes) Programmer's Reference Manual Programmer's Supplementary Documents, Volume 1 Programmer's Supplementary Documents, Volume 2	£25.00/set
System Manager's Manual (1 Volume)	£10.00

4.2BSD Manuals are No Longer Available

1. Tom Ferrin of the University of California at San Francisco, a former member of the Board of Directors of the USENIX Association, has overseen the production of the 4.2 and 4.3BSD manuals.

4.3BSD Manual Reproduction Authorisation and Order Form

Date: _____

As an EUUG member[2] in good standing, and pursuant to the copyright notice as found on the rear of the cover page of the Unix®/32V Programmer's Manual stating that:

> "Holders of a Unix®/32V software licence are permitted to copy this document, or any portion of it, as necessary for licenced use of the software, provided this copyright notice and statement of permission are included,"

I hereby appoint the USENIX Association/EUUG as my agents, to act on my behalf to duplicate and provide me with such copies of the Berkeley 4.3BSD Manuals as I may request.

Signed: _____
Institution (if Institutional Member): _____

Name: _____
_____
_____
_____
_____
Phone: _____
Network Address: _____

The prices below include surface postal charges. All payments must be in Sterling by means of: a cheque drawn on a UK bank, or a Eurocheque, or instruction to charge your VISA card (quoting the number, the date of expiry, the name of the card holder, and the address which VISA use when corresponding with you).

User's Manual Set (3 Volumes)	_____ at £25.00/set =	£ _____
Programmer's Manual Set (3 Volumes)	_____ at £25.00/set =	£ _____
System Manager's Manual (1 Volume)	_____ at £10.00 =	£ _____
Total	_____	£ _____

Make your cheque payable to "EUUG" and post it with this order form to:

EUUG Secretariat
Owles Hall
Buntingford
Hertfordshire SG9 9PL
UK

This page may be photocopied for use.

2. Members of EUUG National groups are automatically members of the EUUG.

As the UNIX recruiters. .

We have a wealth of experience in providing consultancy and high level introductions within the UNIX world.
Having expanded our operations in registry recruitment, we are utilising the strong influence and special position that we hold within the UNIX community with companies both large and small. We are keen to hear from UNIX and 'C' literate men and women thinking of a career move.
We will want to interview you to assess your career aspirations and abilities.
We are likely to have met the client companies we will approach on your behalf.
Our company sources are impeccable and close, so your application will be handled quickly, directly, and professionally.
Several of our clients around the UK have urgent permanent positions which they are keen to fill. We will furnish you with full details when you contact us but here are some brief outlines.

INGRES: Senior levels for London and Leeds.
UNIX Kernel/Comms Development: London and Berks.
UNIFY/ACCELL – West of London.
Image Processing – London, Cambridge, Herts, Bristol and Surrey.
X-Windows (low level) London.
UNIX Systems Programming – Large systems experience – Wilts.
Graphics/Workstations – Beds, Berks, Middx. and Surrey.
UNIX/'C' Technical programmers – Manuf. Co. Lancs.
Accounting/UNIX/4 GL – London, Manchester, Bristol, Sussex
UNIX Systems Support – W. London and UK – Wide.
S/W Tools/Case Development – Surrey
Sales Support – London, all areas.

We have many other requirements nationally, so if you are a UNIX professional looking for a career move and you would like the opportunity to discuss it with professionals then:

Fax, email, write or phone...

LINDA ELISHA — *Evening Tel. No: 01 317 8472*
MARCUS LANGFORD-THOMAS — *Evening Tel. No: 01 681 3905*
GERARD FAWCITT — *Evening Tel. No: 01 341 7301*

Daytime Telephone (01) 378 7588 Fax (01) 378 1869 email UK UUCP LINDA @lje

LJE recruitment

LJE UNIX Recruitment 48-50 Gainsford Street, Butlers Wharf, London SE1 2NE.

OPEN LOOK Graphical User Interface

Janet Davis
janet@uel.co.uk

Market Communications
Unix Europe Limited (UEL)
International House
Ealing Broadway
London W5 5DB
+44 1 567 7711

Janet Davis is Market Communications Manager for AT&T Unix Europe.

She has been with AT&T since September 1987 and is responsible for the promotion of UNIX System V and related products and services throughout Europe.

Janet was previously marketing executive with Thorn EMI Computer Software.

The OPEN LOOK graphical user interface, announced in April 1988, employs common sense graphic symbols instead of written commands to help users work more efficiently with UNIX® System V based computers.

OPEN LOOK supports AT&T's commitment to open systems and the need for a standard user interface. It is the next step in expanding the UNIX Systems marketplace. By providing a standard interface across various hardware environments, applications developed with the OPEN LOOK interface will have access to a larger market. The interface has already generated endorsements from key industry players including Lotus, Ashton-Tate and Xerox.

The UNIX operating system is nearly 20 years old. Many parts of the system have evolved over the last two decades, but the user interface of the standard system has not. With the introduction of the OPEN LOOK User Interface, that situation is changing.

The UNIX system user interface evolved in the late 1960s and early 1970s with the emergence of time-sharing as the leading new computing style. Users were primarily scientists and engineers who used terminals to share minicomputers.

This interface was revolutionary in its time for its simplicity and power. A "shell", or command interpreter read command lines from the keyboard and executed a separate process for each command. The system also provided powerful programs to accommodate a variety of terminals for common editing operations so that applications could be made much simpler and still support a consistent user interface.

The Need for a New UNIX System User Interface

In the late 1980's, the UNIX system is being used by customers who have wide ranging needs.

® UNIX is a registered trademark of AT&T in the U.S.A. and other countries

Technology is changing. Even inexpensive personal computers now have displays capable of high quality graphic presentations and mouse pointing devices that increase usability. New computer input and output technologies such as voice recognition and on-screen video are beginning to appear in desktop computers.

Therefore it is time for a new user interface for the UNIX system; one that is accessible, consistent, and meets the needs of its new users. A new UNIX system interface should be easy to use and offer a comprehensive flexible way for programmers to take advantage of it. It must also preserve the investments of the past and prepare us for the future. The OPEN LOOK User Interface is such an interface.

Why call it The OPEN LOOK User interface?

A major trend in computer systems today is the trend to open systems. Open systems are those which are available from multiple vendors, so procurements can be competitive and so common platforms can be developed that preserve customer investments in applications software, documentation and training.

As part of its commitment to open systems AT&T is participating with the industry in defining standards for the UNIX system and its derivatives. The POSIX standard being developed in the United States, the X/Open standard being developed in Europe, and the efforts of the Sigma group in Japan aim to expand a common definition of a portable and open operating system. This common definition can then be used by any number of vendors to produce different implementations of the same operating system.

To support open systems, AT&T will make the OPEN LOOK User Interface available for use in all standard systems, including appropriate licences to all relevant copyrights, patents, and trademarks that pertain to the OPEN-LOOK User Interface and the standard Application Programmer's Interface toolkits.

How did AT&T decide on OPEN-LOOK User Interface?

AT&T, with industry partners such as Sun Microsystems investigated the requirements for the right UNIX system user interface with care, researching works inside and outside the UNIX system environment. This research covered interface designs from the 1970s through the 1980s, weighing the features, motivations, and resulting successes.

Today's most recent user interface developments were scrutinised. AT&T's research experience was combined with reviews from leading industry consultants. AT&T's work on graphical user interfaces dates back to 1965 and a project known as GRAPHIC I - a remote graphical display console. Subsequent work on GRAPHIC II resulted in a key graphics patent, as did work on a bit-mapped graphic workstation and the overlaying of graphic windows.

None of the system user interfaces in the marketplace today was judged suitable for the UNIX system of the 1990s. Each had its good and bad points, but none combined the necessary simplicity, power and openness. As a result, the OPEN LOOK technology was designed for AT&T by Sun Microsystems, Inc. of Mountain View, California. Sun's design is based on original work, contributions from AT&T, and on technology licenced from the Xerox Corporation, which originated many of the concepts present in today's computer interfaces.

What were the Design Goals?

Simplicity:

A user doing a new task on the computer, with no model from prior experience, needs the interface to be simple. This makes the interface easy to learn and to master.

Consistency:

A user doing a new task but having a model from prior experience, either on a computer or from real world experience wants the interface to be consistent. This allows the user to learn the new system very quickly.

Efficiency:

Once familiar with a task, a user will perform it with the least effort - the fewest keystrokes, the shortest movements of the cursor.

What is the environment for OPEN LOOK User Interface?

Because the UNIX system is highly flexible, it has always been used for a wide range of purposes. The designers of OPEN LOOK wanted it to be as adaptable as UNIX systems. They wanted it to work for:

- standard as well as large screens
- a variety of screen resolutions
- a true multitasking environment
- interface technology that is advancing rapidly

Each aspect of the OPEN LOOK User Interface design was examined to see how well it worked. Some common interface designs were abandoned as too unwieldy for one use or the other. New designs were invented to meet needs that are just emerging.

What are the Design Principles of OPEN LOOK User Interface?

Good designs are the result of just a few, carefully considered principles, and the consistency of the whole is more important than the brilliance of a single part. These are the design principles that guided the OPEN LOOK User Interface design:

Good Graphics and Visual Metaphors:

Careful graphics designs help make the interface intuitive, easy to learn, efficient to use. Visual metaphors were used to make operations clear. Ideas that could not be well expressed graphically were discarded.

Functional Balance:

The goals of simplicity, consistency and efficiency sometimes conflict. All parts of the design must balance the three and work for new users, for experienced users doing new tasks, and for experts.

Less is More:

One concept is better than two. One way to do an operation is better than two. There is less to learn, less to remember. These design principles lead to a more detailed list of design rules that are "laws of physics" of the OPEN LOOK User Interface system.

Select then Operate:

Operations follow the familiar noun-verb model: select the object, then pick the action.

Objects have Properties:

OPEN LOOK User Interface objects can have properties. These can be examined and set in a consistent way.

Help for Everything:

Pushing the Help key always gives a message.

Cut/Copy/Paste:

Cut, copy, paste are used to move information.

Stay Up is Universal:

A general mechanism is provided to allow any part of the user interface to be "pinned" to the desk for repeated use.

Visual Controls:

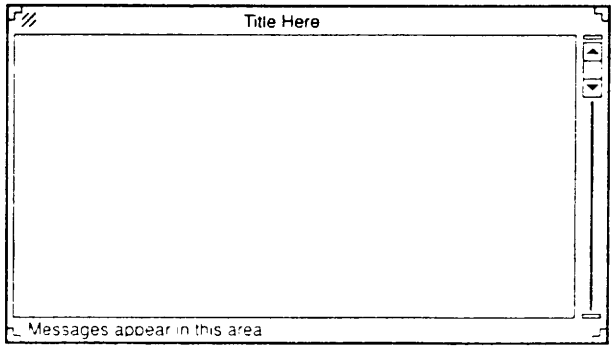
All controls are accessible on a visible control panel.

The OPEN LOOK System

The elements of the OPEN LOOK User Interface:

The Mouse

Basic movement of the cursor is controlled by a one, two or three button mouse. On a three-button mouse, the left button is used to select the object to be manipulated, the middle button to extend or adjust the selection (i.e. select more text) and the right button produces a menu - either for an object or a region of the screen. Selecting, adjusting and producing menus can be performed if the mouse has only one or two buttons.



OPEN LOOK Window with Scroll Bar

Desktop

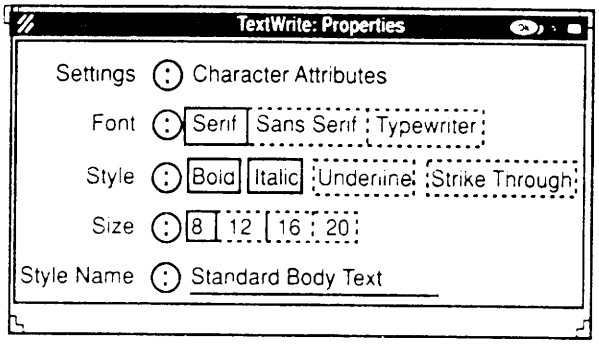
The desktop, or basic blank screen, uses windows that are capable of either overlapping or remaining separate. To move them out of the way, a window may be "closed" or reduced to a small icon or symbol in the corner of the desktop. Every object on the desktop is moved in the same fashion; using a mouse to move the cursor to the object to be selected then "dragging" the object across the screen by holding down the mouse button and moving the mouse. In the same way,

windows may be "stretched" by selecting and dragging the window frame corners using the mouse. These concepts are applied the same way in the OPEN LOOK user interface so the user only has to learn this basic concept once.

Windows, Command Boxes, Property Sheets

Windows, command boxes and property sheets are variations that allow the user to see and control information that pertains to the function being performed. All three are controlled in a consistent way. For example, the box in the upper left hand corner closes the object, whether it's a window, command box or property sheet.

- windows are the centre stage, where text is edited or the engineering design displayed.
- Command boxes pop up to allow the user to fill in more information after he or she has initiated a command. For instance, a user who's just saved a document will see a command box appear so the file name may be entered.
- Property sheets are used to set the properties of objects - such as the tab setting in a document or the font for a paragraph.



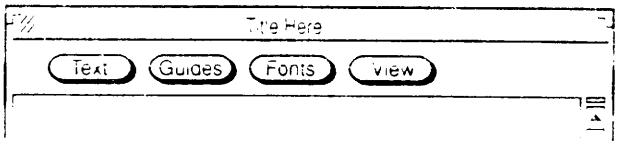
OPEN LOOK Property Sheet

Push Pin

The push pin is used to "pin" to the screen such things as property boxes that may be needed repeatedly and would otherwise disappear each time they were used. For instance, a complicated piece of text may require many variations in fonts - such as heavy use of italics. Making these property changes would be a long and cumbersome process if the property box disappeared after each change. However, pinning the property box to the screen allows the user to refer to it again and again thus saving keystrokes and time.

Control Panels

The controls for an application are on "control panels". These control panels are like the dashboard of a car. They contain the buttons to push to make things happen, like printing or editing a document. The buttons are "pushed" by moving the cursor to be button and clicking the select key on the mouse.

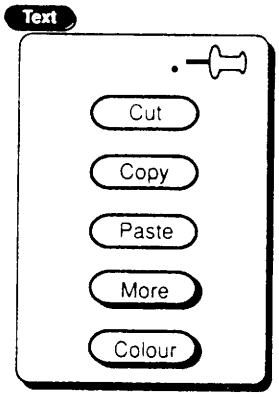


OPEN LOOK Control Panel at top of Window

An application can have more than one control panel. For example, when a spreadsheet and a graph are on the screen at the same time, it makes sense to put a control for the graph next to the graph and a separate control panel next to the spreadsheet.

Menus

The OPEN LOOK Interface allows users to access menus in a couple of ways. Some buttons on control panels are shaded so they appear to be the top of a stack of buttons. Manoeuvreing the mouse to these control panel buttons and clicking the mouse's menu button displays the stack as a menu.



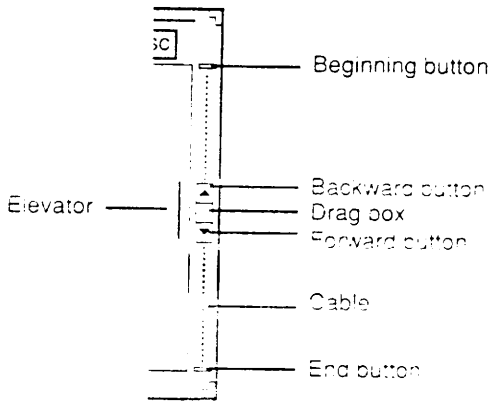
OPEN LOOK Menu

In addition, menus can also be made to pop up when they're needed. For instance, a user who wants to insert, delete or move text first selects the text to be edited, then clicks the mouse button to access a menu from which choices can be made to perform the editing function.

For the user's convenience, no matter where he or she is on the screen or what application is being performed, a menu can be made to pop up to give quick access to important controls. Unlike some user interfaces where a user must manipulate and search to find the appropriate control, the control comes to the cursor, rather than the cursor searching out the control.

Scroll Bar

Scrolling through text occurs through the use of an "elevator" symbol and gives the user the ability to move up and down in the text quickly and easily. The bold portion of the elevator cable provides a visual clue as to the size of the displayed portion in relation to the entire text.



OPEN LOOK Scroll Bar

Pushing the up arrow on top of the elevator car moves it up one line or down one line if the down arrow is pushed. The control buttons are placed next to the car so the cursor need not be walked to the top or bottom of the window to push the buttons. In addition, the cursor moves up or down with the elevator's control button so the user can click the cursor repeatedly without having to reposition the cursor with every click, thus saving keystrokes and movement.

Multitasking

Like the multitasking abilities of the UNIX system, windows on an OPEN LOOK interface can perform functions simultaneously. Users can start an operation in one window while working in another. The title bars of working windows turn grey so the user knows at a glance which windows are busy and which have completed their work. To avoid confusion, each window has its own control panel. Like any other object, windows - either individually or several at a time - can be moved, closed, enlarged or manipulated.

Toolkits for OPEN LOOK

Toolkits contain pre-programmed components or "tools" that applications programmers use to build a user interface for a particular application. For instance, the "scrolling" function is a tool. A programmer who wants to adapt this function to a particular application doesn't have to start from scratch to develop it because it's already part of the toolkit. The OPEN LOOK Toolkits will establish a consistent "Open Look" style among different applications. Providing the tools will encourage developers to support this "look" and they'll be less likely to want to recreate the interface for a new application.

Because the toolkit is "object-oriented" - which means it provides the objects such as scroll bars, menus and command boxes - it can easily adapt to new objects or applications. Its flexibility allows applications programmers to customise elements of another interface while remaining within the OPEN LOOK interface environment. This flexibility will allow the toolkits to conform to the varied requirements of UNIX system applications developers now and in the future.

Initially AT&T will be providing two toolkits:

— OPEN LOOK XT toolkit based on X-Windows systems

— OPEN LOOK NDE toolkit based on X11/NeWS

AT&T will circulate OPEN LOOK specifications for comment this summer and will make them available in the third quarter of this year.

Please contact the author if you would like a copy of this document.

These will include specification of the common style for applications, the Application Style Guide, as well as descriptions of the programming interface for OPEN LOOK under two toolkits, both of which AT&T will support via a single graphics platform.

The first availability of OPEN LOOK features in an AT&T product will be this summer in a window manager for the 6386 workstation, followed by an XT toolkit in the fourth quarter 1988 and an NDE toolkit in the first quarter 1989. Source code for the OPEN LOOK user interface toolkits are scheduled to be available in early 1989.

USENIX Association News for EUUG Members

Donnalyn Frey
donnalyn@uunet.UU.NET

Frey Communications

Ms. Frey is the USENIX Association Press Liaison. She provides members of the press, USENIX Association members, and EUUG members with information on the activities of the USENIX Association.

Summer 1988 Conference

The San Francisco Summer 1988 Technical Conference and Exhibition was informative and enjoyable. Over 3,200 people attended the conference, the largest in the Association's history. All of the tutorial sessions sold out, some well before the conference. The technical presentations presented new and interesting information and were well received by attendees. The BOF sessions were, as a whole, well attended. One session, the Open Software Foundation BOF, attracted a large crowd. The speakers at the BOF, technical managers at OSF, answered questions and spoke of their plans for the future.

The conference reception was held Thursday night at the Exploratorium, a hands-on science museum. Attendees dined among physics, chemistry, and biology experiments that invited participation.

The Proceedings of the San Francisco conference sold out on the last day of the conference. A second printing will make additional proceedings available soon. To request copies of the San Francisco proceedings, please send mail to {uunet,ucbvax}!usenix!office.

USENIX Signal CONTEST

The gauntlet was thrown and the challenge was accepted. The winners of the USENIX signal contest was not an European this year, but an American. The judging committee consisted of both Americans and Europeans to avoid any bias toward Americans. Thousands of entries were received and a truck was used to move all the entries to the hotel bar for the judging. As the evening wore on, and the judges had too much fun, the winners were culled from the losers.

The next day the conference attendees anxiously awaited the announcement of the contest winners. First prize was copy of "The C Programming Language", autographed by Dennis Ritchie and donated by Prentice-Hall. Second prize was an EUUG penknife, donated by the EUUG.

The presentation was made under an EUUG banner. The contest announcement and awards were presented by Peter Colinson, British, with the assistance of Jean Wood, American, and Andrew Hume, the Bell Labs Murray Hill resident Australian announcing the first prize winner.

An attempt by EUUG to commandeer the awards ceremony was put down, but the EUUG standard still flew valiantly over the smoldering hulk of the podium. After order was restored, it was reported

by the usually reliable suspects that Peter Colinson grudgingly acknowledged defeat by the brilliant American entries. When confronted with the allegation, Colinson refused to deny it.

The winning entry in the USENIX signal CONTEST was:

SIGSEUSS

I meant what I said
And I said what I meant
Your program is buggered
One hundred Percent

The FaceSaver Project

The FaceSaver Project was again successful at the San Francisco conference. The FaceSaver combines photographs and registration information onto sticky labels that conference attendees could give to exhibitors and other attendees. After the conference, the attendee list will feature a postage stamp size portrait beside each name and address.

For the San Francisco conference, attendees who previously had their faces saved found three pages of labels already in their registration packets. Both photographs and registration information could later be updated in the FaceSaver room.

The FaceSaver portraits are captured via a video camera using AT&T Targa M8 graphics boards installed in Bell Technologies PC AT clones running the SCO XENIX version of the UNIX operating system. Portraits are printed using a Postscript laser printer.

The FaceSaver project is run by Lou Katz. It is sponsored by the USENIX Association to aid in improving attendee recognition at the conference. A sample FaceSaver label is reproduced below.

FaceSaver 2/88

Donnalyn Frey
703-764-9789
uunet!donnalyn
Frey Communications
PO Box 2051
Fairfax, VA 22031

FaceSaver 2/88

Donnalyn Frey
703-764-9789
uunet!donnalyn
Frey Communications
PO Box 2051
Fairfax, VA 22031

Winter 1989 San Diego Conference

The next USENIX Association technical conference is scheduled for January 30 - February 3 in San Diego, California. Tutorials will be given on Monday and Tuesday, January 30 - 31. The technical sessions will be presented Wednesday, Thursday, and Friday, February 1 - 3. The call for papers for the San Diego conference was sent to USENIX Association members in July. The deadline for papers submitted for the conference is October 7, 1988. Papers are requested for formal review as candidates for inclusion in the conference. Papers that are accepted will be presented at the conference and published in the conference proceedings. Suggested topics for papers includes (but are not limited to): Performance Analysis and Tuning New User Interfaces and Applications System and Network Security Networking and Distributed Services RISC versus CISC in UNIX Software and System Management Tools Standards Graphics and Electronic Publishing Evolution of UNIX for the 1990s

All papers should describe new and interesting work. For full information on submitting a paper for the San Diego conference, contact the USENIX conference office at P.O. Box 385, Sunset Beach, CA 90742, USA and request a copy of the call for papers for the conference. Technical questions should be referred to Greg Hidley or Keith Muller at sd@usenix.ucsd.edu. The deadline for submissions is October 7, 1988.

C++ Mini-Conference

The USENIX Association's first C++ Mini-Conference will be held October 17 - 21, 1988 in Boulder, Colorado. The mini-conference was begun to accommodate all the people who wanted to attend the smaller 1987 C++ workshop. The conference includes two days of tutorials and three days of technical sessions. The conference is intended to provide information for both new and experienced C++ programmers. For information on the conference, contact the USENIX conference office at P.O. Box 385, Sunset Beach, CA 90742, USA.

Conferences and Workshops

If you need further information on upcoming annual USENIX Association conferences or workshops, contact the USENIX conference office at P.O. Box 385, Sunset Beach, CA 90742

USA. The conference office can provide you with information on the annual Computer Graphics, Large Installation Systems Administration, UNIX Security, and UNIX and Supercomputers workshops. The office can also provide information on the C++ mini-conference and the semi-annual technical conferences.

Unix for Managers

A conference is aimed at managers who wish to obtain an overview of Unix; various workshops will provide an opportunity to talk to Unix experts.

The conference is being organised on 14-16 Sept 1988 in Munich.

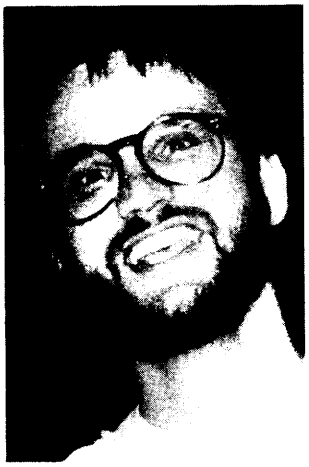
For further information please contact:

GI Deutche Informatik-Akademie GmbH
Godesberger Allee 99
5300 Bonn 2
+49 228 37 67 53

EUnet News

Daniel Karrenberg
dfk@cwi.nl

Centrum voor Wiskunde en Informatica



Daniel received a graduate degree in Computer Science from the University of Dortmund, West Germany, in 1987. During his studies he was exposed to the Unix Operating System and worked as a systems manager & consultant. He is a co founder of the GUUG & was heavily involved in the starting stages of the German part of EUnet. He currently works at the CWI (Centrum voor Wiskunde en Informatica) in Amsterdam, the Netherlands, where his responsibilities include local & wide area networking as well as the operation of the EUnet international backbone node called "mcvax". He also is a member of the the EUUG Executive Committee with responsibility for EUnet. He enjoys being involved in both managerial & operational aspects of EUnet.

Abstract

Short news items from Europe's largest network for the research and development community.

This is intended for those interested in news about recent developments in EUnet. If you would like to know more, come to the next EUUG conference in Portugal where I will give a more detailed report (which might be printed in this forum if the editor bears with me).

News Flash

The load average of mcvax (a poor little vax 11/750) just reached 25.4 again. The average average :-) is about 18.

Infrastructure

Our international workhorse mcvax is really bogged down at the moment, but relief has arrived. The Dutch group (NLUUG) have received a HP 800 series machine as a kind donation from Hewlett Packard. This machine was christened hp4nl. It has been placed next to mcvax and the news feed for the Dutch sites is being transferred as I write this. hp4nl is equipped with the new "Trailblazer" modems which support uucp and achieve transfer speeds around 14000bit/s (no spelling error, I counted the 0s twice!).

This really speeds up local news feeds!

While hp4nl already provides some relief to mcvax, we are looking to replace mcvax as well. The EUUG has received several generous offers. Testing of some machines is currently under way and I hope to bring good news to Portugal.

The 9600bit/s link to the news and mail service of the Usenix Association uunet has been running smoothly and is carrying almost all of EUnet's mail to and from the United States. Introduction of the link helped stabilise costs for mail and news exchange with the US while the volume is increasing steadily. The line is still fully paid for by EUnet, but it is expected that Usenix will start to cover a share of the costs this year.

Due to the work of Simon Kenyon we have also received reliable signs from the U.S. Internet administration that EUnet will be granted full IP access to the Internet in the next few weeks. This will make it possible for backbones to deliver mail even more quickly and to provide such services as fetching information available only via "anonymous ftp". The details of this still have to

be worked out. These developments and the expected completion of a new trans-atlantic fibre optics cable have prompted us to order an upgrade of the U.S. link to 64kbit/s. This will be only marginally more expensive than the current 9600bit/s link. This is a sign that the European PTTs are becoming more competitive. We actually received a very glossy folder from the Dutch PTT offering their leased line services including bargain discount schemes. We were stunned!

Inside Europe the Eunet infrastructure continues to grow and as traffic grows it becomes more and more attractive to replace links over public X.25 networks with leased lines. A line between CWI and INRIA, the French backbone site has been in operation for some time running IP over X.25. A line to the University of Kent (England) is due to be delivered in August. A 64kbit/s circuit between the campus where CWI is housed and CERN in Geneva was ordered last week. It will be jointly financed and shared with the high energy physics community's HEPNET. This is the first example of such cooperation between two European networking organisations and hopefully it's the start of a trend. There are rumours from unido that a link between CWI and Germany is going to be ordered real soon now. Another candidate for a fast connection to EUnet is the Scandinavian community with its new powerful infrastructure called Nordunet.

While EUnet extends it's infrastructure other European networking organisations do likewise. Talks have been going on over the last year to explore the possibilities of sharing some of the infrastructures in order to reduce costs and enhance services. Piet Beertema and Bjorn Eriksen are participating in a joint working party with representatives from other major networks to create a shared pan-European X.25 infrastructure for research and development. Piet has written a report that shows that with current funding the bandwidth on the most crucial links can be increased by at least an order of magnitude, if – the infrastructure is shared by all participants. Some money has been promised by the European community and the government of Norway to start this activity off this year, but a lot of formal problems will have to be solved to create a stable shared infrastructure. EUnet will continue it's own plans while participating in the sharing exercise until the situation is more stable.

News Flash

The load average on mcvax has dropped below 8. Something must be wrong, I'll have to check!
– X.25 connection fixed again. I'll have to hurry now, deadline is tomorrow!

New EUnet Services

There have been many calls for news archiving services for groups such as comp.sources.* and that need is recognised by the backbones. Some of them already operate archive servers and some plan to do this in the near future. Once the new Dutch and European backbone machines are in operation we will start with some central support and coordination activities in this area. The same goes for general information servers holding documents, site information and the like. Suggestions from users are welcome.

A study has been completed on how EUnet could be migrated to the use of ISO protocols. The conclusion is that it's worth starting now, it can be done, that resources are needed and most importantly that it should be done in a user driven way rather than by coercion. I'll give a detailed account of this in Portugal.

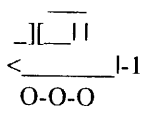
A printed electronic mail directory is currently under development. It will hopefully contain information about as many European organisations connected to R&D networks as possible. The directory will be useful as an "off line" reference tool usable especially by people not yet on the net. Backbone sites often get requests for comprehensive information about reachable organisations from people who want to connect to EUnet. This directory is hoped to fill that gap. Of course care is being taken that it can't be used as a junk mail list. We still aim to have them available in time for Portugal.

Well, that's about it. The load on mcvax is back above 20, my pot of coffee is empty. See you in Portugal.

EUUG Software Distribution

Frank Kuiper
euug-tapes@cwi.nl

Centre for Mathematics and Computer Science
Amsterdam



YAAFANT! (Yet Another Announcement For A New Tape)

Due to reasonable demand, the volume of material, and its rather special purpose, a new tape has been made available. It is called the "EUUGD14 ISODE" tape. This is version 4.0 of this non-proprietary implementation of some of the OSI parallel protocols suites as defined by the International Organisation for Standardisation (ISO), the International Telegraph and Telephone Consultative Committee (CCITT), and the European Computer Manufacturer Association (ECMA).

This release of ISODE is coded entirely in C, and is known to run under the following operating system without kernel modifications:

BSD 4.2 and 4.3
Ultrix
AT&T Unix SVR2 and SVR3
AIX
HP-UX
ROS
Pyramid OsX

Since a Berkeley Unix system is the primary development platform for ISODE, the documentation and source are somewhat slanted towards that environment. The tape contains some 12Mb of both tools and documentation. EUUG will send you the tape only. If you want the complete documentation on paper (some 800 pages!), you will have to order this tape as follows:

Send a cheque or a purchase order for 200 Pounds Sterling to:

Department of Computer Science
Attn: Soren Sorensen
University College
Gower Street
London, WC1E 6BT
United Kingdom
Telephone: +44 1 387 7050, extension: 3680

Specify either 1600 bpi 1/2-inch reel tape, or sun 1/4-inch cartridge tape. The tape will be written in tar format and returned with a documentation set, via DHL. Do not send tapes or envelopes. Documentation only is the same price.

EUUG and, its counterpart in the USA, Usenix are currently making arrangements to exchange their software library. If all goes well, and I have no reason not to think so, members of the EUUG will be able to purchase the Usenix software distributions via the EUUG, in the same way that the EUUG software distributions are handled now. Of course, this also means that Usenix members will be able to obtain the EUUG software distributions.
As soon as I have the Usenix tapes, and know the exact distribution requirements (some of the Usenix tapes contain source, licenced by AT&T), I will announce the availability of them in an EUUG Newsletter.

As the diversity of Unix machines grows, both in hardware and in software flavours, the demand for the EUUG software distributions on floppy disk grows as well. Because of the limited available space on floppies, it is not feasible to make all of the EUUG distributions available on floppy. However, I hereby ask the readers of this newsletter to inform me of the type of floppies they would like to have (format specifications, both hardware and software wise please) and the kind of software they would like to have the

floppies contain. Currently we are considering to make a few highly interesting tools available. But you will have to tell me what you consider to be the most interesting tools.
You can reach my at the snail mail and electronic mail addresses listed at the beginning of this article. This is your chance to get some REAL software on to your machine, if all you have to communicate with a outside world is a floppy disk unit :-)

The rest of this article consists of the well known list of distributions that are now available on both 1600 bpi reel-tapes, and QIC-24 format streamer-tapes. (800bpi and QIC-11 on demand only).
As always, anyone is invited to make their own tools, games, etc available for publication on an EUUG tape. Please contact me for more details. Don't hesitate, just put the results of many nights of serious programming and hacking in the public domain, and you might even become famous!

This is a list of all the current (August 1988) EUUG software distributions. It is a short description of the available tapes. Any changes to the contents of the tapes, as well as announcements of new tapes will be placed in the EUUG Newsletter.
Prices of the tapes are in Dutch guilders (DFl), and do not include VAT-taxes. They do contain postage cost for surface mail within Europe. Any special shipment costs, like with DHL, will be billed to you.
The first price listed is for reel-tapes in tar 1600 bpi format, the second one is for distributions on cartridge tapes in QIC-24 format. Prices for 800 bpi reel tapes and QIC-11 cartridges may differ from the ones listed.
Note that you have to be an EUUG member (or a member of a local UUG) to obtain tapes at list prices. Non-members will have to pay an extra DFl 300,- per tape.

EUUGD1 R6: UNIX V7 system, specially made for small DEC PDPs (11/23, 11/34, etc). The Kernel supports the UK terminal driver. V7 source licence minimum.
Price: DFl 120,-/180,-

EUUGD2: Early Pascal compiler of the Free University of Amsterdam. V7 source licence minimum.
Price: DFl 120,-/180,-

EUUGD3 R3: Currently not available.

EUUGD4: Software tools, sampled by the Software Tools Users Group. Most of the software is written in Ratfor, for which a Fortran support tool is included. This tape is available in different formats: DEC RSX, DEC VMS, UNIVAC, IBM MVS, UNIX tar, MIT line feed format, and MIT card format (80 columns).
Price: DFl 150,-/180,-

EUUGD5: A collection of benchmark programs made up by EUUG.
Price: DFl 60,-/180,-

EUUGD6: (USENIX 83.1) USENIX tape, containing contributions from various UNIX System Group Members. This is a licence dependent distribution: V7, V32, SIII, V6 or no licence disclosure available.
Price: DFl 240,-/300,-

EUUGD7: UNIX|STAT Version 5.2. A collection of about 25 data manipulation and analysis programs written in C by Gery Perlman.
Price: DFl 60,-/180,-

EUUGD8: A collection of useful software, based on the so called Copenhagen tape (EUUG Unix conference Autumn 1985).

EUUGD9: A collection of useful software, based on the so called Florence tape (EUUG Unix conference Spring 1986). Price: DFl 150,-/210,-

EUUGD10: MMDFIIb. Multichanel Memo Distribution Facility (version IIb). This is a powerful, domain oriented mail system with access control and the ability to communicate over a variety of network systems including TCP/IP, JANET, UUCP, PHONENET, etc. It has been ported to a variety of UNIX's including but not limited to 4.[123]BSD, 2.9BSD, System III/V on a variety of different hardware. You should first obtain a licence agreement by sending a message to euug-tapes@mcvax. Return the signed licence with your order.
Price: DFl 90,-/180,-

EUUGD11: This is the 'Boat' tape: the Helsinki EUUG 1987 spring conference. It contains about 25 Megabytes of programs, games, etc. Including: jove, less, nag, news, rn , uEmacs, uuencode and lam.
Price: DFl 120,-/180,-

EUUGD12: This is the Dublin EUUG 1987 autumn conference tape. It contains about 26 Megabytes of programs, games, etc. Including: copytape, crc_plot, fastgrep, jove, kermit, notes, uupc, nethack, cron, sendmail, mh, Recipes, brl-gw, isode, pcip, pctelnet.
Price : DFl 120,-/180,-

EUUGD13: The latest conference tape for the London EUUG 1988 spring conference tape.

1. Commands and Application Programs

X	X-window 11 release 2
afio	manipulate archives and files
cake	maintain groups of related files like make
calctool	window based calculator.
changed	looks for files that have changed after given date
chat	Chat in real-time with other users (System V)
config	print details of machine and C compiler configuration
copytape	copy magtapes on systems with only one tape drive
cpmod	copy modes, ownerships and times of file
g++	GNU c++ compiler
gawk	GNU awk
gcc	GNU C compiler 1.18
gdb+	GNU debugger
gemacs	GNU emacs editor version 18.50
graphedit	a general purpose graphic editor
isode	ISO Development Environment
kermit	The Kermit file transfer program version 4E(067)
lem	interactive package for editing simple graphics
little-st	little Smalltalk system
magtapetools	magtape handling package
mcc	merge C compiler
modemcap	a modem independent dial(3) package
mstools	read and write MSDOS formatted diskettes from Unix
mush	The Mail User's Shell for electronic mail
news	the netnews system
nrofftab	nroff driver table compiler/de-compiler utility
pd-diff	public domain diff with the -b and -c options
pdtar	tape (or other media) file archiver
perl	Practical Extraction and Report Language (C,awk,sed,sh)
postscript	PostScript interpreter
psfig	a troff preprocessor for PostScript figures
pshalf	produce PostScript pages two to a (paper) page
ptc	Pascal to C translator
qsubst	qsubst - query/substitute strings in files
qterm	query a terminal to determine its name
revgrep	search a file for a pattern backwards, provide tails
ru	compact list of users on all
sc	spread sheet calculator
semex	interactive exerciser for System V semaphore operations
sets	performs set operations on its arguments
shar	create shell archive file for extraction by /bin/sh
shsem	Utilize System V semaphores from the shell
substr	extract a substring from the input arguments
vmail	tty interface to MH
w	display users and processes (for System V)

2. System Calls

3. C Library Subroutines

pc-curses	screen/window management library
regexp	regular expression handler
rpc	library routines for remote procedure calls
syslog	logging routines

4. Special Files

pty	pseudo tty driver for system V machines

5. File Formats

6. Games

moria4.85	a dungeon adventure game in the manner of rogue
omega	another adventure game

7. Miscellaneous

8. System Maintenance

arc	a general archive utility
autobaud	terminal speed detection
backup	perform tape backups
enable	enable, disable getty on tty lines
smail	UUCP mailer with routing
survey	generate simple plot of system load and # of users
sush	restricted shell to grant specific limited privileges
watcher	system monitoring program

Price : DFl 120,-/180,-

EUUGD14: This is version 4.0 of this non-proprietary implementation of some of the OSI parallel protocols suites as defined by the International Organisation for Standardisation (ISO), the International Telegraph and Telephone Consultative Committee (CCITT), and the European Computer Manufacturer's Association (ECMA).

This release is coded entirely in C, and is known to run under the following operating system without kernel modifications:

BSD 4.2 and 4.3
Ultrix
AT&T Unix SVR2 and SVR3
AIX
HP-UX
ROS
Pyramid OsX

Since a Berkeley Unix system is the primary development platform for ISODE, the documentation and source are somewhat slanted towards that environment. The tape contains some 12Mb of both tools and documentation in machine readable form. EUUG will send you a tape only.
Price: DFl 120,-/150,-

If you want the complete documentation on paper (some 800 pages!) with the tape, you will have to order this distribution as follows:

Send a cheque or a purchase order for 200 Pounds Sterling to:

Department of Computer Science
Attn: Soren Sorensen
University College
Gower Street
London, WC1E 6BT
United Kingdom
Telephone: +44 1 387 7050, extension: 3680

Specify either 1600 bpi 1/2-inch reel tape, or sun 1/4-inch cartridge tape. The tape will be written with tar format and returned with a documentation set via DHL. Do not send tapes or envelopes. Documentation only is the same price.

EUUG Software Distributions Order Form

If you want to order any tape, please write to:

EUUG Software Distributions
c/o Frank Kuiper
Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam
The Netherlands

For information only:

Tel: +31 20 5924056 (or: +31 20 5929333)
Telex: 12571 mactr nl
Internet: euug-tapes@cwi.nl

Please note that for distributions D1, D2, D3 and D4 (and in some cases also for D8) a copy of your source licence agreement with AT&T for at least UNIX version 7 should be enclosed. Note also that you have to be an EUUG member (or a member of a national UUG) to obtain tapes at list prices. Non-members will have to pay HFl 300,- per tape extra as handling fee. Please enclose a copy of your membership or contribution payment form when ordering. Do not send any money or cheques, you will be invoiced. Tapes and bill will be sent separately.

All reel tapes come in tar format, 1600 bpi. 800 bpi is possible on request. Cartridge tapes come in tar format, written with dd, with a blocking of 126b. This is a so-called QIC-24 format. QIC-11 is available on request.

This page may be photocopied for use.

Name: .....

Address: .....

.....

.....

I would like to order the following:

.....

.....

.....

EUUG (or national UUG) membership form enclosed? Yes / No
Copy of AT&T source licence enclosed? Yes / No

"I declare to indemnify the European UNIX systems User Group for any liability concerning the rights to this software, and I accept that EUUG takes no responsibilities concerning the contents and proper function of the software."

Signature: .....

Date: .....

UNIX Clinic

Colston Sanger
olibc1!colston@olgb1.oliv.co.uk

Olivetti International Education Centre

Colston Sanger is a lecturer at the Olivetti International Education Centre, Haslemere, UK and a visiting lecturer in the Faculty of Engineering, Science and Mathematics at Middlesex Polytechnic, London. As soon as he finishes this article he is going off to get married...

Efficiency considerations and shell functions in the SVR2 shell

This month's topic is efficiency considerations in the UNIX SVR2 shell. At the end, I'll also talk about shell functions: how you can use them to add new commands to your login shell, and also as subroutines within shell-scripts.

Does it need to be more efficient?

Way back before the beginning of time (1 January 1970 in the UNIX world) when computers were big and slow and expensive, programmers put a lot of effort into making their programs more 'efficient'. Today, computers are cheap and programmers are expensive. Efficiency, while still a consideration, now primarily means saving programmer, not machine, time.

The first question, then, is: does it it need to be more efficient? There is little point in spending weeks making a script more efficient if it is only going to be run once a year.

The second question is: have you timed it with `time` or `timex`? Either will give you the elapsed, user and system times taken in the execution of a command, but `time` reports in tenths of a second whereas the newer `timex` reports in hundredths (and also has several useful options). As an example, here is `timex` run on a shell script `xwho` (extended `who`):

```
$ timex xwho
sue       tty62      Jul 16 08:49
yvonne    tty34      Jul 16 08:56
tony      tty14      Jul 16 09:00
peter     tty72      Jul 16 13:27
sandy     tty61      Jul 16 13:27
colston   tty11      Jul 16 13:44

real         1.22
user         0.34
sys          0.50
```

'Real' is the elapsed wall-clock time; 'user' is time spent actually executing the user program; and 'sys' is time spent inside the kernel executing system calls on behalf of the user program.

PATH organisation

Given that your script really does need to be more efficient, what can you do? First of all, you could look at `PATH` organisation. The default value of `PATH`, the environment variable that defines the search-path for commands, is:

```
:/bin:/usr/bin
```

meaning the current directory (the null string before the first colon), /bin and /usr/bin, in that order. For most users this setting is inefficient. Most of the time they are executing programs in /bin or /usr/bin. It makes sense, therefore, to change the default value of PATH (the best place is in /etc/profile) to:

```
/bin:/usr/bin:.
```

so that the current directory is searched last. Note that current directory is also defined here explicitly as . 'dot'.

Another practice that may result in minimal searching is to reset PATH within your shell script. You may also want to use the absolute or relative pathnames of commands. (If a command name contains a /, then the search-path defined by PATH is not used.) Here is a shell script, paths, illustrating some of these techniques. If you are a system manager, you may want to use it to check the setting of PATH in your users' .profile files:

```
# paths - checks settings
#         of PATH in
#         users' .profiles

PATH=/bin:/usr/bin

trap ' ' 1 2 3
awk -F':' '$3 > 99 { \
 printf("%s %s\n",$1,$6)}' 3
 /etc/passwd \
| while read LOGIN LOGINDIR
do
    echo "$LOGIN\t\c"
    if [ -f $LOGINDIR/.profile ]
    then
        ( cd $LOGINDIR
        if grep 'PATH=' .profile
        then
           :     # do nothing
        else
           echo '$PATH not set'
        fi
        )
    else
        echo 'No .profile'
    fi
done
```

Referencing files

Interpreting a long pathname makes a significant demand on system resources, especially if individual directories are large. If you need to read (or write) many files in a directory, it's far better to cd to that directory first. For example, the following two scripts do the same job, but the second runs faster and demands fewer system resources:

```
for file in /usr/you/src/talk/*
do
    pr $file | lp -dpr1
done
```

```
cd /usr/you/src/talk
for file in *
do
    pr $file
done | lp -dpr1
```

Note that in the second script, the pipe to lp is also outside the loop.

Pipelines

In pipelines, it's common sense to put reducing filters before non-reducing filters. Aim to reduce the amount of data being processed at each stage. For example, these two pipelines are equivalent, but the second is more efficient:

```
$ ps -ef | sort -t' ' +1n | \
  grep tony

$ ps -ef | grep tony | \
  sort -t' ' +1n
```

Use shell built-ins wherever possible

Use shell built-in commands wherever possible. They are faster, and there is no process creation overhead. Also, in the UNIX System V Release 2 shell, more commands - such as echo and test - have become built-ins.

Similarly, if you need to call a shell script from within a shell script, if possible 'source' it with the . command.

Shell functions

Shell functions are new in UNIX System V Release 2. Unfortunately, apart from a brief paragraph under sh in the manual, there's very little in the way of documentation on either syntax or how to use them.

So, how are shell functions used? So far as I can see, there are two ways of using shell functions: first, to add commands or alias existing commands in your login shell - much as you would use `alias` in `csh` or `ksh`; or second, as subroutines within a shell script - like the . built-in command or the `source` command in `csh`.

Within a shell function, you can use any shell construct or built-in command, call any UNIX utility, call any other shell script (though that's a bit silly), or call any other shell function - plus, of course, shell functions, like shell scripts, can be recursive. So what's the difference between a shell script and a shell function? Unlike shell scripts, shell functions are executed within the current shell, so you avoid the process creation overhead. Also, the syntax is different.

The basic format of a shell function is:

```
function_name ()
{
    commands ...
}
```

Using shell functions to add commands to your login shell

Here are some short examples of how you can use shell functions to add commands to your login shell. What you do is put them all into one file called, say, `.funcs` in your login directory and then 'dot' the file in your `.profile`, as in:

```
# Example .profile file
stty cr0 nl0 ofill ixany
stty kill '^x' intr '^c'
PATH=/bin:/usr/bin:$HOME/bin:.
MAILCHECK=60
CDPATH=.:..:../..:$HOME
export PATH MAILCHECK CDPATH
export PS1 PS2
# This is the important bit:
. $HOME/.funcs
#
trap '. $HOME/.logout' 0
```

Anyway, here are those example shell functions. The first is called `chdir` and was written specially for MS-DOS types:

```
chdir()
{
# chdir - change PS1 prompt to
#         echo working directory.
if [ $# -lt 1 ]
   then cd
else
   cd $1
fi

if [ "`pwd`" = "$HOME" ]
   then PS1='$ '
else
   PS1="`pwd | sed s?$HOME/??`>"
fi
}
```

This next one I actually use quite a lot. I call it `sw` because there's a VAX/VMS DCL command called that, and also because it's short for 'switch':

```
sw()
{
# switch directory
if test -z "$OLD_DIR"
then
    OLD_DIR=$HOME
fi
NEW_DIR=${1:-$OLD_DIR}
OLD_DIR=`pwd`
cd $NEW_DIR
pwd
}
```

The next one is very simple - it just aliases the shell built-in `exit`. I wrote it because I thought there was more consistency between `login:` and `logout` than with `exit` or `CTRL-d`:

```
logout()
{
# logout
exit
}
```

Finally, here is a rather longer example that implements command history. It also has a bit of history attached to it. A couple of years ago, I got very fed up with somebody on one of my UNIX courses who went on and on about how UNIX or the SVR2 shell didn't have command history (he was either an MS-DOS type or a `csh` fan - I forget which) so, after dinner, I sat down and wrote this:

```
history()
{
# shell command history
# Don't forget to export PS1 in
# your .profile
while echo "$PS1\c" ; read CMD
do
  if test "`echo $CMD | \
            cut -c1`" = "~"
  then
      NUM=`echo $CMD | \
           cut -c2-3`
      if test -n "$NUM"
      then
          CMD="`sed -n $NUM'p' \
               $HOME/.history`"
          eval $CMD
      else
          cat $HOME/.history | \
              nl
      fi
  elif test "$CMD" != ""
  then
     eval $CMD
     echo $CMD >> $HOME/.history
     tail $HOME/.history > \
            /usr/tmp/HIST.$$
     mv /usr/tmp/HIST.$$ \
            $HOME/.history
  fi
done
}
```

So there you have it: the SVR2 (Bourne) shell with history. As I remember, Brian Kernighan in the report on the 'new' `awk`,[1] shows how you can do something very similar but probably much better, with `awk`.

Using Shell functions as subroutines

Just enough space left to show how you can use shell functions as subroutines. There are two things to remember: first, shell functions must be defined before they are referenced; and, second, shell functions don't `exit`, they `return`.

As subroutines, a typical structure would be:

```
shell_function_1()
{
    ...
}

shell_function_2()
{
    ...
}

main_loop()
{
if test ...
then
    shell_function_1
else
    shell_function_2
fi
}
```

It would be nice to have an example, but there it is - perhaps another time. On the other hand, if anybody out there is `really` interested, I'll post the code in this article (plus even more examples, with subroutines!) to `eunet.sources`.

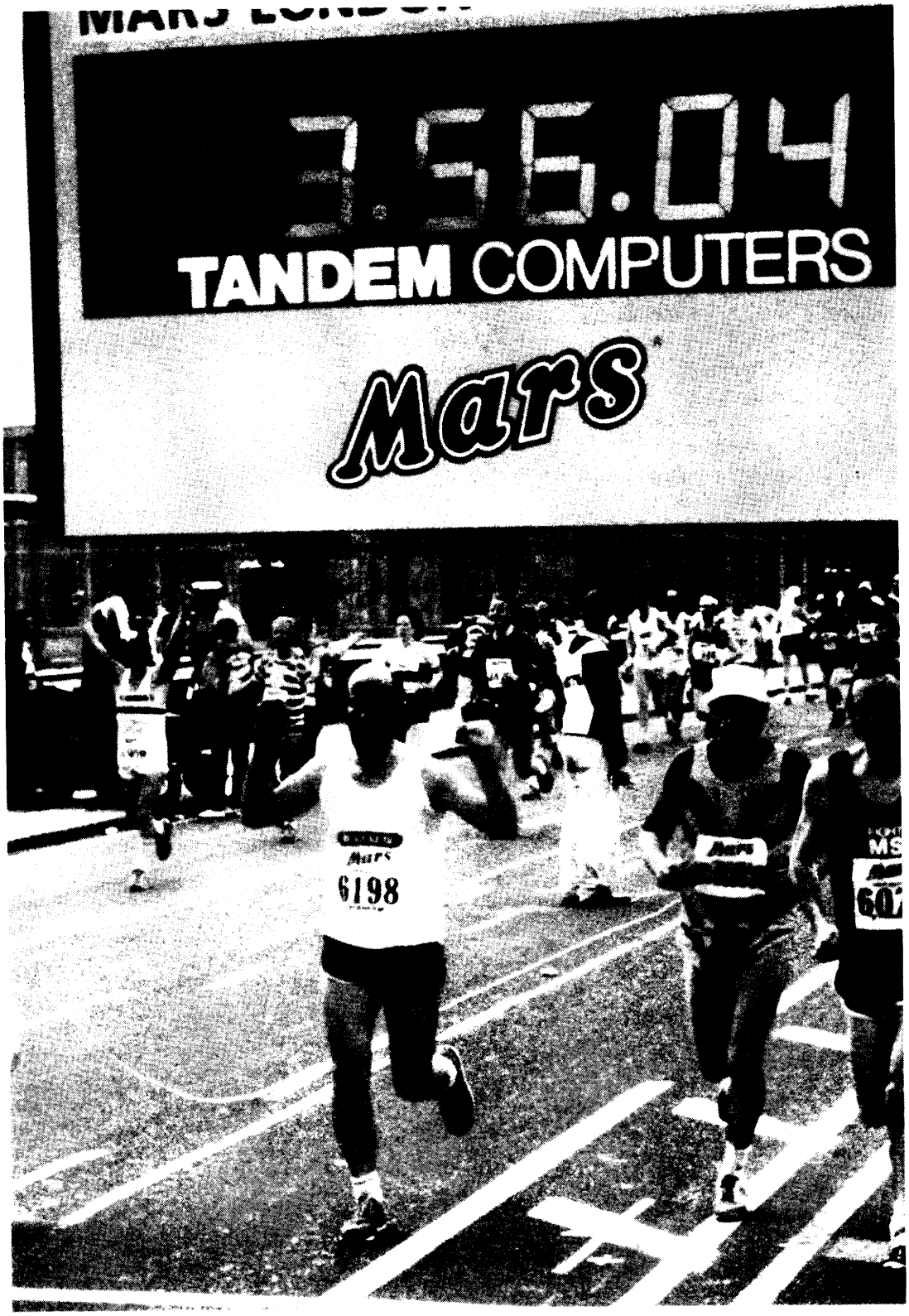
1. AT&T Bell Laboratories, *Computing Science Technical Report*, No.118 (1985). There is now a section on the new *awk* in Volume 1 of the UNIX System V Release 3 `Programmer's Guide`.

EUUG

Glossary

Here are the definitions of a few not-so-common English words that can be found in the newsletter. Where a word has several meanings the way in which it is used in this issue is the one explained.

adequate	Enough, sufficient
adhered	Stick to, behave according to
albeit	Let it be completely true that
allegation	(Unproved) assertion
astutely	clever, crafty
confront	Meet in defiance, oppose
consensus	Agreement, majority view
consists	Made up of
consolidated	Solidify, unite product
convey	Carry
cull	Select from ...
delude	Deceive
diverse	Several, more that one
enclose	Surround, shut up
hegemonic	Ruling, supreme
impede	Obstruct, hinder
omission	Non-inclusion
parochial	Confined to narrow area
podium	Raised platform
preserve	Keep safe from harm
presume	take for granted
prevent	Stop, hinder
promote	Help toward
purchase	Buy
rebut	Turn back
rectified	Put right
related	Have reference/be similar to
riveting	Hold attention
scarcity	Not enough of
slant	Lean
snag	Drawback, unexpected obstacle
tended	Inclined to ...
unique	Only one of


3.56.04
TANDEM COMPUTERS
Mars
Mars
6198

AUSTRIA - UUGA
Friedrich Kofler
c/o Austro Olivetti
Rennweg 9
A-1030 Vienna
AUSTRIA

BELGIUM - BUUG
Marc Nyssen
Department of Medical Informatics
VUB, Laarbeeklaan 103
B-1090 Brussels
BELGIUM

DENMARK - DKUUG
Mogens Buhelt
Kabbelejevej 27B
DK-2700
Bronshoj
DENMARK

FINALND - FUUG
Johan Helsingius
OY Penetron Ab
Box 21
02171 ESPOO
FINLAND

FRANCE - AFUU
Miss Ann Garnery
AFUU
11 Rue Carnot
94270 Le Kremlin-Bicetre
FRANCE

GERMANY - GUUG
Mr Laengle
GUUG
Mozartstrasse 3
D-8000 Munich 2
WEST GERMÄNY

ICELAND - ICEUUG
Marius Olafsson
University Computer Center
Hjardarhaga 4
Reykjavik
ICELAND

IRELAND - IUUG
John Carolan
Glockenspiel Ltd
30 Iona Crescent
Dublin 9
EIRE

ITALY - i2u
Ing Carlo Mortarino
i2u
Viale Monza 347
20126 Milano
ITALY

NETHERLANDS - NLUUG
Patricia Otter
Xirion bv
World Trade Centre
Strawinskylaan 1135
1077 XX Amsterdam
THE NETHERLANDS

NORWAY - NUUG
Jan Brandt Jensen
Unisoft A.S.
Enebakkvn 154
N-0680 Oslo 6
NORWAY

SWEDEN - EUUG-S
Hans Johansson
NCR Svenska AB
Box 4204
17104 Solna
SWEDEN

SWITZERLAND - UNIGS
Professor Wolfgang Fitchner
Institute for Integrated Systems
ETH Zentrum
CH-8092 Zurich
SWITZERLAND

UNITED KINGDOM - UKUUG
Bill Barrett
Owles Hall
Buntingford
Hertfordshire SG9 9PL
UNITED KINGDOM

The European UNIX systems User Group
Owles Hall
Buntingford
Hertfordshire SG9 9PL
UK
+44 763 73039