INDRE BY-TERMINALEN
ved Københavns Universitet
Studiestræde 6 over gården
DK-1455 København K
Telefon 01 - 12 01 15

EUUG

EUROPEAN UNIX® SYSTEMS USER GROUP NEWSLETTER

Volume 3, No. 3
AUTUMN 1983

EUUG

European UNIX† Systems User Group

Newsletter Vol 3 No 3
Autumn 1983

† UNIX is a Trademark of Bell Laboratories.

Some Words from the Editor

First, an apology for a number of formatting errors in the last Newsletter. This was due mainly to the rush getting it ready before I left for the USENIX conference in Toronto. Of course, this one is rushed too....

The Dublin meeting was fun, and I think everyone enjoyed it. I always thought of Irelend as a country of greens, but mostly the colour I saw was black with a little cream on top....

Well, included herein is a report of the Dublin meeting. As Peter says in his report, many people submitted a paper. However, only two so far have been in a format which I can read with my computer.

In the future, there will be proper "Proceedings" produced from these meetings, giving authors the opportunity to have something to put down on their CV. It is not yet too early for you to submit a paper for the next meeting in Nijmegen in April.

One of the important things to come out of the Dublin meeting was agreement to form National UNIX† groups, even for the UK. The EUUG is changing structure, hopefully to better meet the needs of its members. It also gives more scope for each member to have a say in what goes on.

Finally, note that the name of the Mathematisch Centrum in Amsterdam has changed, it is now the Centrum voor Wiskunde en Informatica (CWS), or, in English, the Centre for Mathematics and Computer Science (CMCS). However, the old name is still recognised.

Jim McKie
Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam
The Netherlands

tel: +31 20 5924147
UUCP: mcvax!jim

† UNIX is a Trademark of Bell Laboratories.

32-bit UNIX? A unique opportunity with the New Perkin Elmer 3205

Perkin-Elmer offers **fully supported** 32-bit UNIX*, probably the world's best timesharing system, in an unbeatable package deal based on the brand-new super-compact model 3205.

Application development is exceptionally easy and fast, performance is optimised for good interactive response, and there's a vast range of time-saving O/S-linked utilities.

Educational Special**

8 user system

Hardware

Model 3205 cpu
512KB memory
50MB disc
Dual bus architecture
8 user ports
1 CRT
LP Interface

Software

UNIX Edition VII (Source)

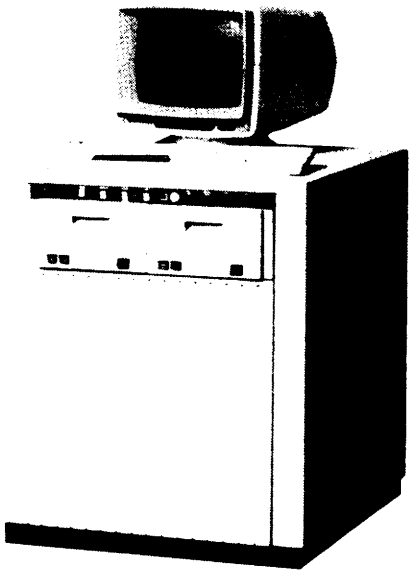
Source Code Control System
Full screen editors
FORTRAN 77
PASCAL
BASIC
C
Berkeley enhancements

All this for just

£16,700

Optional Extras

RM/COBOL
CORAL 66
LEX/32 Word Processing
SIBOL† (DIBOL)††



The world's first 32-bit UNIX implementation was on a Perkin-Elmer machine at Bell Labs by Ritchie/Thompson.

Completely standard Edition VII UNIX - Rapid response for up to 16 users - Memory expandable up to 4MB - Disc capacity greatly expandable - And a range of third-party products such as APL, Mistress, Informix and Unify (all RDBMs) to applications such as payroll, accounting and ledgers.

Commercial Special

8 user system

Hardware

Model 3205 cpu
512KB memory
50MB disc
Dual bus architecture
8 user ports
1 CRT
LP Interface

Software

Unix Edition VII (binary)

Source Code Control System
Full screen editors
FORTRAN 77
PASCAL
BASIC
C
Berkeley enhancements

All this for just

£21,670

Optional Extras

RM/COBOL
CORAL 66
LEX/32 Word Processing
SIBOL† (DIBOL)††

Note: *Unix is a trademark by Bell Labs. Sibol† is a trademark of Software Ireland. DIBOL†† is a trademark of DEC.
**Educational users must qualify and obtain academic source licence from Bell Labs, otherwise binary will be supplied.

To: Karen Jarvis, Perkin-Elmer Data Systems Ltd., 227 Bath Rd., Slough, Berks SL1 4AX
Telephone: Slough (0753) 77777

☐ I am interested in UNIX.
Please send information.

Name _____

Position _____

Address _____

_____

Telephone No. _____

PERKIN-ELMER

EUUG Meeting
Trinity College, Dublin, 7/9 September 1983

Peter Collinson
Secretary

Introduction

These reports are becoming somewhat redundant because most speakers have submitted a paper. The idea for future conferences is to have a formal call for papers which will be sent to referees and published as proceedings. This report is derived from my somewhat sketchy notes and will serve to give a guide to what went on.

Even on close inspection, I could find no reference in my notes to the word 'Guinness'. This is a pity because the best Guinness in the world is to be had in Ireland, the stuff called Guinness elsewhere just isn't the same.

Day 1 - 7th September

The first session of the conference was devoted to vendor presentations. As usual, these notes may be incorrect and aim just to give the flavour of the talks. For further (and correct) information, contact the vendor.

Item 1: 10.35am **Eddie Bleasdale, Bleasdale Computer Systems Ltd**

Bleasdale Systems

Eddie started with a brief history of his company and how they relate to UNIX systems. The hardware which the company is offering is based on the M68000 processor, which can be either 8Mhz or 10Mhz. There will be an upgrade to the M68010. Memory sizes are from from 768Kbytes up to 3.5Mbytes, the backplane is Multibus. Peripherals include: a high performance Winchester disc as a main disc subsystem; a floppy disc; ½″ magnetic tape and ¼″ streamer tape. It is also possible to have customised hardware options.

The UNIX system is the Unisoft version, which is a UNIX V7 system plus some Berkeley enhancements. The company also does some software work; it has enhanced the disc driver for the hardware and provides some diagnostic software in a monitor ROM. They have support for an array processor, a floating point processor and specialist I/O interfaces.

Bleasdale now offer the following languages: C, F77 (Silicon Valley Software, this can use the floating point processor), Pascal (two flavours: UCSD and Silicon Valley Software), BASIC+, RM/COBOL, Whitesmith's cross compilers, cross assemblers from Real Time Systems, APL, POP-LOG and LISP.

The company has collected an impressive list of applications packages, some of which are: the **xed** word processor, a spread sheet system from View Comp, the **Mistress** relational data base system, a payroll system called CPAY and a ledger system from UNIGEM. There were others but my hand would not move fast enough.

Item 2: 10.55am **John Olford, NCR Ltd**

The NCR Tower

Not a building but a computer, the TOWER 1632 is a M68000 based UNIX system. Its name is derived from the tall box in which it is packaged. The box is designed to sit quietly by a desk in an office, it does not have any noisy fans which would annoy people. The main internal bus is Multibus, but it does have its own memory bus which supports up to 2Mb. Discs are either 5½″ or 8″ Winchester, the main cabinet can support up to 60Mbyte and an expansion cabinet can support another 160Mb. Other devices include: floppy disc, a streamer tape drive and variety of terminal and printers.

The UNIX system is the Unisoft port and supports C, RM/COBOL, SMC/BASIC, PASCAL (an NCR compiler to the ISO standard) and F77.

Price: entry level £ 11,000; typical configuration £ 18,000 and a full configuration will be £ 50,000.

Item 3: 11.10am **John Roseman, Urban Software**

Urban Software

Urban Software are a software development house based in New York, they have one main product called **Leverage**. This is a screen based data capture system intended to generate ASCII files suitable for post-processing by all those UNIX tools which you know and love (or hate). They do not sell to end-users but prefer to sell through a third party - Sphinx Ltd in the UK.

The main theme of John's talk was precisely that there exist a lot of tools on the UNIX system and so why don't commercial people people use them!

Urban also produce a quarterly newsletter, I have two editions which contain extremely useful information on who produces what UNIX system on what machine and for how much.

John then went on to talk about the recent UNIX history, this has been documented in these pages before, so I will omit it.

Q. How does Urban Software make money?

A. Most revenues come from the CP/M version of the **Leverage** package. This is sold through a third party - Perfect Software.

Item 4: 11.25am **Frank Graham, Software Ireland**

SIBOL

Software Ireland are a subsidiary of the National Westminster Bank and have offices in Belfast, Dublin and California, employing 30 people. Their main areas of interest are software products for UNIX, RSTS and CP/M; training and various consultancy.

Their main UNIX product is SIBOL which is a commercial programming language which is compatible with DEC's DIBOL. DIBOL is DEC's main supported commercial language and has a lot of applications programs written in it.

SIBOL comprises a compiler, linker, editor, debugger and a library of external routines. It is written in C and so runs on all UNIX and UNIX look-alike systems.

Software Ireland have ported a number of commercial applications packages which they now sell under the title of PHENIX.

Item 5: 11.34am **Brian Rees, DEC**

DEC UNIX Systems in Europe

Well, after all this time..... a representative of DEC at a EUUG conference talking about DEC selling and supporting UNIX!!!

DEC now fully supports a PDP11 UNIX system called V7M-11 which has been announced and can be ordered. A UNIX system will be available to run on the PC350 some time towards the end of the year.

The 32-bit VAX native system (which was called ULTRIX on the slides) will be 4.2BSD running on the range of VAX processors from the VAX11/730 to the VAX11/782. The system will have support for the new distributed file architecture (HSC50).

DEC are also committed to support VNX which will run a UNIX system under VMS, they are also busily engaged in installing programs into VMS which show the 'cultural aspects of the UNIX system' - for example, the shell. They have some other programs which are UNIX utility look-alikes, for instance they have a package with the same functionality as SCCS.

All VAX processors running VMS, VNX or native UNIX systems will be supported by DEC. Although it is unclear whether UNIX licensees who are licensed by other means than DEC will be able to get hold of the appropriate device drivers in order to support DEC's field service engineers.

There will be a full array of software support service options for VNX. The educational services will concentrate on VAX/VMS.

Item 6: 11.50am **Matthew Wallace, Hewlett/Packard**

The HP9000

The HP9000 system is a 32-bit processor which was launched in September. It runs a UNIX system and HP are committed to put UNIX systems onto a wide range of their processors. Currently, the HP9000 is targeted towards the fields of Computer Aided engineering (e.g. Graphics - CAD/CAM) and areas which involve software productivity.

The UNIX system, called HP-UX is System III plus the usual Berkeley enhancements. HP have their own graphics package and their own Data Base Management system. They intend to support an Ethernet working to the IEEE 802.3 standard.

HP are launching a M68000 system running UNIX in the next two months.

Item 7: 12.03pm **Mike Salmon, Taurus Technology**

Taurus Technology

Taurus are a case of *mv 'Structured Methods' Taurus*. They are engaged in training of various sorts and can generate 5 day workshops in UNIX and C. The majority of their business is on-site training for customers using the local machines. Recent customers include British Telecom and Rediffusion Computers.

Internally, they use a 32-bit Gould machine and an ONYX.

☞ Lunch (but no Guinness!) ☜

In the afternoon, there were two parallel sessions. Fortunately, I could only manage to go to one. The session which I attended appears first.

Item 8: 2.00pm **Emrys Jones, Cgram Software**

EUUG Business

The EUUG business is very hard to take notes about, largely because it tends to be vague acrimonious discussion.

The main item informed the membership of the proposed change in the structure of the organisation. Currently, the composition of the EUUG board does not really reflect the actual structure of the group. Apart from the UK, all other countries have a national group. It is felt that it would be better to (a) set up a national UK group and (b) change the EUUG board to comprise the chairmen (or nominated members) of the national boards. Since this full EUUG board would be too big and unwieldy to actually take decisions, a smaller unit called the *EUUG Executive board* should be formed from the EUUG board, this small group would be responsible for day-to-day running of the EUUG.

David Tilbrook (Imperial Software Technology/Imperial College) undertook to establish a group to get the UK National Group off the ground. David said that the main aim of the UK group should be to get all UK sites connected via EUNET. There will be a preliminary discussion about this on September 23rd, 1983.

The second item of business informed the membership of the formation of a UK Chapter of /usr/group. /usr/group is an organisation in the USA which consists of commercial vendors and users. The organisation is a prime mover in the UNIX standards debate and has produced a draft standard for system calls. There was some (extremely acrimonious) discussion about why the UK /usr/group needed to be formed in the first place. In general, my personal opinion (note: personal) is that the aims of EUUG are to *promote* UNIX and the aims of /usr/group are to *vend* UNIX. These aims are not mutually exclusive, but the groups may need to to go about things in different ways.

The next spring EUUG conference will be in Nijmegen, Holland from 16th→18th of April, 1984.

Item 9: 2.34pm **Jim Kennedy, Bill Murphy, AT&T**

Licensing and other issues

There was not much new stuff from AT&T but I'll include most of the information I have because licensing is always one of the hot issues when talking about UNIX. Some of this is derived from the Bonn Proceedings.

System V licensing is just taking a long time, there are lots of applications in the pipeline. Academic applications have been shelved until all the commercial licences are issued. The table below gives the licensing costs:

System V licence & Fees ($)			
	Initial CPU	Additional CPU	Upgrade from SIII
Commercial	43000	16000	1000
Educational/Administrative	16000	400	0
Educational	800	400	0

For educational licences, AT&T will charge $800 to license as many machines as you want as long as all machines are licensed in one application. For this they will supply a standard distribution. Any additional requests will then cost $400.

In response to a question about royalty payments for selling time on a machine, Bill Murphy said that there would be no further timesharing agreements and there would be no cost for selling time.

Bill then went on to talk about the software support which AT&T are selling. The base for this 'phone-in' system is Lisle, Illinois. Level 1 support will cost $150 ($50 for an additional CPU) per month and will entitle the purchaser to a newsletter, a method of problem reporting and periodic updates. Level 2 will cost $350 ($100 additional CPU) and will supply Level 1 support plus a hotline reporting system (2pm→11pm GMT) with a 2 hour response to problems. Requests outside these hours will cost $100 per hour.

Documentation for System V can be bought separately. The two volumes of product release documents cost $30, three volumes of UNIX Operating system manuals cost $60 and the 7 volumes of Operating System guides cost $145.

AT&T now sell UNIX training, this is based in the USA but they are willing to send an instructor and a course to Europe. There are five courses:

1) Operating System implementor's curriculum
2) Application developer's curriculum
3) Manager/supervisor's curriculum
4) System administrator's curriculum
5) User's curriculum

Bill also announced three new product releases: the S statistical package, the UNIX Writers Workbench and The UNIX Instructional Workbench.

Telephone numbers below are from the USA - I'll leave it to you to decode them for your country.

UNIX System Contacts for AT&T International				
	City	Name	Telephone	Telex
USA	Basking Ridge,NJ	W.J.Murphy	201-953-7581	219345 ATTIUR
UK	London	R.R.Hall	441-930-0691	24660 ATTIUK
Europe	London	A.Ursano	441-930-0691	24660 ATTIUK
Ireland	Dublin	M.Felton	353-1-515-411	25815 TCRNEI
Greece	Athens	S.P.Tsitouris	301-3232-797	219245 AP5GR
Egypt	Cairo	I.N.Shalaby	160-200-726-998	92444 ATTIUN

☞ Tea (but still no Guinness!) ☜

Item 10: 4.09pm **Rich Graveman, Bell Laboratories**

UNIX in the future

Rich's talk gave an overview of the way Bell Labs are thinking about the development of UNIX.

Bell now see UNIX systems as a set of programs and utilities which can be unbundled. Rich showed the following information which shows how the unbundling may take place. A UNIX system comprises:

- The Operating System
  consists of:
  - ☐ The Kernel: which provides a core interface to the hardware; implements system calls; supports the file system structure; and copes with resource sharing and allocation
  - ☐ Device drivers

- ☐ A set of basic utilities: which comprise the Shell; system administation tools to set the system up, maintain the file system and to configure the system for the hardware; directory and file management utilities; and an editor.

- Optional utilities
  - ☐ Functional Command groups: the utilities are split into groups of commands which are the requisite tools for certain well defined tasks. For instance, a set of utilities which allow interuser communication within one machine (**mail, mesg, news, wall, who, write**) may not be required on a single user machine.
  - ☐ Optional device drivers
  - ☐ Program support libraries
- Workbenches
  - ☐ Programmer's workbench: provides a development environment for programming and debugging.
  - ☐ Document preparation workbench: has an input mechanism, a formatter, **tbl**, **eqn** and some form of production capability.
  - ☐ Writers workbench: proofreading and style analysis programs.
  - ☐ Instructional workbench: computer assisted instruction currently supporting 4 courses (this is not **learn**!).
  - ☐ Analyst's workbench: contains the **S** package, data entry and analysis tools.

Rich then talked about the rationale behind the AT&T agreements with the four micro-processor companies. The micro-processor area is the fastest growing segment of the computer industry and it seemed sensible to get an up-to-date UNIX system available on quality chips as soon as possible. There is also a need to standardise ports of UNIX systems in order to encourage the development of applications. The agreements which have been worked out enable the μ-processor companies to port UNIX System V with Bell guidance. At the end of the day, WeCo will own the ported system, and it will be possible to get source licences from WeCo. Binary licences will be available from the μ-manufacturers.

The direction for the future development of UNIX systems hinges on the main goal which is 'to make the UNIX operating system a common basis for applications on a wide range of machines from micro-computers to mainframes.' Current activities to fulfill this goal include: (a) the provision of a single supported UNIX system for both the Bell system and commercial customers; (b) Bell is working with users to achieve a standard; (c) Bell will continue to release new products and (d) intends to unbundle UNIX into functional pieces.

In the unscheduled future, Bell is looking at various areas:

- User friendliness, items of interest here include: a visual shell, some methods of forms input, a help facility, operation aids for running UNIX in a large machine environment and finally, some graphics routines and systems.
- Standards: items here range from a standard for error messages from programs (which is fairly easy to do) to a standard for command syntax (which is very hard).
- Application programs and packages
- Security

Continuing concerns are: reliability, performance and documentation.

There were several questions:

Q. In the future, the main limitation will be the memory requirement of programs. Are there any plans to look into memory utilisation or is memory so cheap that it doesn't matter?

A. It probably doesn't matter - but the splitting of UNIX into several options will help with this.

Q. Will UNIX ever support dynamically loadable device drivers?

A. Not in the short term.

Q. Will the 'Newcastle connection' be supported under Bell UNIX?

A. (Answer from Rob Pike, Bell Labs research) In the research section of Bell labs there are 16 VAX systems with a connected file system (the UNIX system is called Edition 8). The system does not support the moving of processes from one machine to another which Newcastle does. There are NO plans to release Edition 8.

Q. What's in release II of System V?

A. Bug fixes, some enhancements to performance.

Q. Is there a firm date for Bell Labs UNIX systems supporting virtual memory?

A. Not this year.

Q. Will there be an enhancement of the performance of the file system in order to support virtual memory.

A. There will be no change to the file system.

☞ UNIX Training session ☜

This went on in parallel with the talks above. This section of these voluminous notes has been supplied to me by Cornelia Boldyreff.

The official programme included five balanced presentations of half an hour duration. But these began to run over as the afternoon crept on. So that the last presentation was completed just before 5.30pm, when we were due to be thrown out of the room. Despite the rush through, the response of the audience was enthusiastic and positive. In future issues of the newsletter, notes by the various presenters will be published, if they all keep their rash promises to send copies to the editor.

Item 11: 2.00pm **Ben Salama, Sphinx**

UNIX information and Training Survey

Ben reviewed UNIX publications - books, magazines and market surveys. His personal preference for the book: *UNIX - The Book* was applauded by one of the authors in the audience. Ben also reviewed training options - live and recorded.

Item 12: 2.30pm **Alex Osadzinski, Quantime Systems**

UNIX fundamentals for non-programmers

Alex reviewed the history of UNIX and its development as a commercial product. He then let the Bell Labs UNIX stars perform on video: Ken Thompson, Dennis Ritchie, Brian Kernighan *et al.* Great stuff.

Item 13: 3.00pm **Terry Crook, C-Gram Software**

Shell programming

All those features in the Bourne shell which mystify tyros were explicated by Terry. There is a lot to learn about the shell, this session started late and ran on. Some people managed to get tea before the next session.

Item 14: 4.00pm **Andy Rutter, Taurus Technology**

UNIX fundamentals for programmers

This talk was subtitled *UNIX - The Presentation*. In 22 slides, Andy covered: how commands work, system calls, the process environment, file access, inter-process communication, and the standard I/O package. He also managed to answer questions, too, in just over half an hour.

Item 15: 4.45pm **Peter Griffiths, Taurus Technology**

C Language programming

You've heard about the books, seen the movies, been shell shocked, totally immersed in UNIX; so, to complete the session, all there is to C. This was obviously a moving experience for some Pascal programmers in the audience.

☞ End of Day 1 (and onto the Guinness!) ☜

Day 2 - 8th September

It is hoped that most of the people who gave talks will have either submitted a paper or will do so before this wonderous report reaches you.

Item 16: 9.00am **J.Marino, Universitau di Genova**

Experiences with BERKNET

The University of Genoa wanted to connect a VAX11/750 and a number of small PDP11 systems and rather than connecting with **uucp**, it seemed sensible to use some form Local area network (LAN). The main problem with networking is to convince people that they need it, no-one will pay any money towards setting up a LAN until they have seen the benefits. The strategy adopted was to demonstrate the need for a network by installing a low cost one with the expectation that when users need more speed they will be willing to pay for it.

The UNIX systems were all running some version of Berkeley software (4.1BSD on the VAX and 2.8BSD on the PDP11s), and these releases include the software for a LAN system called **berknet**. It seemed sensible to pick up this software and use it.

The **berknet** software is able to drive several types of connection which trade speed of operation with the amount of intelligence which is required in the kernel. The simplest type of protocol uses a standard teletype line in COOKED mode and needs no kernel modification, the next step up involves the installation of the **bk** packet driver as a line discipline in the kernel but still uses a standard teletype line. More expensive links use the **bk** packet discipline via a high speed link like a DMC11.

Problems with the implementation were mostly due to the PDP11s. First, the deamon which runs the network did not fit into the PDP11, this was cured by moving a number of strings into a separate file and also by removing redundant code by eliminating unused routines. This was done in a cunning way by putting all the routines into an archive library and recompiling. The second problem is that the **bk** line driver did not work on a PDP11, this was fixed by recoding and making the kernel *copyout* routine able to return an odd number of bytes.

The result was a network whose raw transfer rate is greater than 100 bytes/sec (on a 1200 baud connection - faster than that and the PDP11s fall over). The network has a good error rate and has proved a good investment in time.

In the future, it is expected to extend the LAN to other machines and then to use an Ethernet connection to improve the speed on the connection.

Item 17: 9.33am **Peter Jones, University of Essex**

Touch Input speeds up interaction with UNIX tools

With a title straight out of *Electronics* magazine, Peter talked about the work he has been doing with a touch panel attached to the screen of a normal character VDU. His touch panel works by infra-red, although there are many technologies (capacitance, conducting surface, sound, pressure and light beams). The main disadvantages of a touch panel as compared with other pointing devices is that (a) input is physical and tiring, (b) the hand tends to obscure the display, (c) the input has a low resolution (which is no problem on a normal character display), and (d) there is some problem with parallax because the screen glass is thick. Finally, hands are sticky things and the screen tends to get dirty.

However, the pointing is a very natural action and using a touch panel does not involve picking anything else up. Peter seemed really keen on his system.

He has written a screen editor where positioning is done by pointing and other selective inputs are done by having user definable touch buttons on the screen.

☞ Coffee (black, no Guinness) ☜

Item 18: 10.37am **Sam Leffler, Lucasfilm**

4.2BSD, Reflections on a large software project

Sam subtitled his talk '4.2BSD. The morning after'. The main aim was to give some assessment of the work involved in the 4.2BSD development.

Sam started with a chronology of UNIX VAX development:

December 1978	UNIX 32V released
December 1979	3BSD released
	demand paging - Bill Joy
January 1980	DARPA meeting
	the great VMS vs UNIX controversy
	VMUNIX adopted by ARPA
July 1980	Computer Science Research Group (UCB) funded
October 1980	4BSD (later 4.0BSD)
	performance enhancement
	Pascal, long identifiers
	void in C, job control, stdout buffered
June/July 1981	Draft 4.2 system manuals circulated
July 1981	4.1BSD released
	VAX11/750, auto configure,
	VM performance enhancements as a result
	of the Kashtan benchmarks
August 1983	4.2BSD released

He pointed out that there had been a long gap in time between the 4.1BSD release and the issue of 4.2BSD. This had been partially filled by the release to certain sites of intermediate systems (4.1a→4.1c). The 4.1BSD system has several advantages: (a) it is well tuned; (b) it is stable and very robust; (c) except for the virtual memory management and job control, the system is V7 at the system call level; and (d) the MTBF of the system is very high, typically as much as 2→3 months. Various figures are quoted on how many 4.1BSD systems are running. Sam said that he estimates that 15% of all VAXs sold by DEC are running 4.1BSD, estimates vary from 10%→25%.

The goals for 4.2BSD were: to provide the DARPA research community with a standard operating system; to provide a distributed environment at UCB, to maintain the UNIX philosophy†; and to maintain 4.1BSD robustness.

To achieve these aims 4.2BSD has interprocess communication (IPC) primitives, fast network support, the ability to have very large virtual machines, and a fast file system. The file system performance enhancement is required to support large virtual memory for processes.

In 4.1BSD and all previous UNIX systems, IPC is achieved by pipes and multiplexed files. Pipes are not really good enough because they have to share a parent process and multiplexed files are limited to a single machine by their use of the UNIX file system name space. When designing the new mechanism it was felt necessary to have: (a) straightforward semantics, (b) the ability to make an efficient implementation, and (c) the new system must allow higher level objects to be constructed.

In the area of VM enhancements, it was felt that 16Megabyte processes supported by 4.1BSD were too small, and also there were no mechanisms for taking advantage of the large address space.

† Jeers from some members of the audience

The aims for 4.2BSD were to provide a 2 Gigabyte address space and to provide facilities for using this address space by allowing shared files and mapping files into memory. To do this it is necessary to: page the page tables, have a better swap management and to increase the file system performance so that mapped files work efficiently.

4.1BSD had increased the basic file system block size from 512 bytes to 1024 bytes and this had significantly increased the performance of the file system. But there still are problems: the file system fragments very quickly, the disc layout is not very good because it is a linear array of blocks and the I/O transfer size is too small. Solutions to the problems were to (a) implement a larger block size and allow block fragments to minimise disc wastage; (b) to implement a better disc layout policy; (c) to parameterise to allow tuning and (d) to replicate essential information (super blocks) to permit better robustness.

Sam then talked about the 'Good stuff' in 4.2BSD. In the IPC, the user interface is sound and it generates a clear set of primitives. In the implementation, it has turned out that datagrams and 'out of band data'† are useful. It is also possible to make a clean extension from on-machine work to the network environment. The networking interfaces work well and the internal model has proved to be successful (with some caveats). There is a lot of latency and the hardware support works well.

The file system has met the bandwidth requirement and performance is now limited by the hardware (it proves that UNIBUS discs are bad news). 4.2BSD have **mkdir, rmdir** and **rename** implemented as system calls and this is a big performance win. The idea of symbolic links (i.e. a file containing a name of another file) has been very heavily used. The impact of the changes to the directory format (255 character names) has been small.

Having praised the good stuff, Sam went on to point out the 'Bad stuff' in the implementation. In the IPC, debugging facilities are non-existent; he felt that the addressing *vs* naming issue had not been tackled properly; there is a problem with connection shutdown and the documentation is totally insufficient.

The file system is a lot more complex and will take some time for system administators to learn how to use it effectively. A bigger problem is that interaction with the new storage architectures (DSA) is not good.

The signal mechanism is 4.2BSD is totally incompatible with the old system.

Finally, Sam said that it is possible that 4.2BSD may probably be the last UNIX software release from UCB. The people there are getting interested in different areas.

Item 19: 11.30am **Teus Hagen, Maths Centre, Amsterdam**

Pros & Cons for different LAN techniques

This was a free-for-all with Teus saying procative things and attempting to get a discussion going, it was partially successful. I was unable to take notes, which means that I must have been paying attention to what people were saying rather than reading the overhead projector slides.

☞ Lunch (Even if you wanted Guinness by now, there was none) ☜

† *expedited* in 'Yellow book' terms.

Item 20: 1.45pm **Film by Lucasfilm**

Computer Graphics and the Return of the Jedi

This film was a 'short' showing how some scenes for the film had been developed. It proved a bit of an anti-climax because we needed to see the 35mm final scene. Still it was interesting to see the nice hardware/software.

Item 21: 2.01pm **Peter Collinson, University of Kent**

UNIX at UKC

See the paper elsewhere.

☞ Tea (after the last speaker everyone needed lots of Guinness) ☜

Item 22: 3.35pm **Marja-Riitta Kiovunen,Timo Kunnas, Helsinki University**

HUT APL, an APL System for UNIX

Marja-Riitta gave a brief description of the features of APL and Timo talked about their implementation. I quote from their abstract:

HUT APL, an APL system compatible with IBM VS APL, has been developed in Helsinki University of Technology. The system runs under UNIX 4.1BSD running on a VAX computer. It consists of an APL language interpreter, workspace routines and associated commands, facilities to use auxilary processors with shared variables and an editor modified for APL use.

So far, APL systems commonly available for UNIX have been incomplete implementations, lacking some features and often giving wrong results, especially in special cases such as empty arrays. VS APL is the *de facto* standard used by most manufacturers, it also conforms with the ISO and ANSI proposed standards which are being developed. The HUT APL system is downward compatiblewith VS APL, with minor exceptions which are mostly sue to the underlying architecture. It has features typical of many UNIX sub-systems, such as the possibility of using UNIX commands, editors and files; and the ability to start asynchronous or synchronous processes which can communicate with the APL system.

The system is based on the APL available from USENIX, this was originally developed in Yale University and later at Purdue and Berkeley. These implementations are somewhat inefficient. For instance, all numbers are represented internally in 64 bit floating point form, which is neither efficient nor preserves space. Also, the APL syntax analyser is written in **yacc**, and APL syntax is too dynamic for this to work properly. However, it was more important for the developers at Helsinki to generate a system which worked correctly rather than one which worked in a super-efficient way.

The HUT APL system is completely written in C, and can thus be ported to other architectures. It has a clean environment into which any new features of the APL language can be implemented easily. The system is in the public domain and is available on request.

Item 23: 4.05pm **Michael Tilson, Human Computing Resources Corp**

A Tutorial on C language portability

This main aim of this talk was to demonstrate that C is an unsafe language but with a bit of care it is perfectly possible to port C programs between different architectures without a lot of difficulty. Portability is an important issue because software is expensive to write and if the program is 'portable' then it costs less to move it. If software is portable, then the software is not locked into one hardware vendor. Mike also said that portable software is more reliable, since it tends to be

less 'dirty', and use fewer 'tricks'. Errors in type matching often indicate errors in thinking.

Mike was concerned about source code portability and not any related issues such as binary portability, portable library routines, the operating system interface and the command interface.

The talk was illustrated by small examples, all of which have happened in practice and all of which passed happily through **lint**. I won't attempt to reproduce the examples here, but the main points which the examples made are:-

- Type casts and **lint** do not solve all problems.
- There is no guarantee that the bit representation of a **char** pointer is the same as that of an **int**.
- It is not safe to assume that the value 0 is the same as a pointer containing zero. This is especially hard when passing a zero value into a routine which expects a character pointer as an argument. The expression NULL should always be used rather than the constant 0.
- Whenever an **int** variable contains a number which is greater than an **int** on a PDP11, the variable should be defined as **long**.

Mike says that non-portable C cost HCR about $100,000 in the last year.

Item 24: 4.36pm **Mike Banahan, Taurus Technology**

Benchmarking - report and discussion

Mike had followed up some discussion about EUUG generating a benchmark package which had occurred at Bonn. He announced the release of the *EUUG Benchmarks* on the 16th September. The benchmarks are a complete package consisting of a shell script, a re-written **time**(1), a makefile and a set of benchmarking programs written in C.

The aim of the package is to provide a simple standard set of programs which are easy to run and will not generate too many figures for comparision. The programs are deemed to be the core of the benchmarking package and any further programs will be added to the package. Existing programs in the package will not be removed.

The package currently consists of the following operations:

1) Check that the new **time** works, this uses a human with a watch to ensure that the clock on the machine is running at the correct speed.
2) Size the machine and run some tests to determine the CPU speed. This gives a fiddle factor which is then used to scale the tests so they don't take a very long or very short time.
3) CPU tests: include integer arithmetic, register arithmetic, arithmetic and function calls and some *real* maths.
4) I/O tests: include pipe throughput, pipe throughput using 1 byte writes, some writes to /dev/null and disc writes and reads.
5) Compound load test: run a C compilation sequence once, and then run many in parallel.

The report from the package gives the normalised real/user/system times for each test, plus the actual times for the compilation of each of the programs.

Mike ended with a dire warning; people expect too much from benchmarks. The benchmarks will not predict machine/system performance, nor are they intended to. The main aim is to allow a very broad comparision between machines.

☞ End of Day 2 (and onto the Mansion House for drinks) ☜

Day 3 - 9th September

Well, finally the Guinness and general lack of sleep caught up with me. I must confess and apologise for missing the first speaker. I reproduce his abstract.

Item 25: 9.00??am **H.Ludwigs, Quantum**

Portable Forms Management System

Forms management systems are usually fixed onto one type of host computer and one type of screen device. Therefore, it is impossible to move a program which relies on screen form I/O from one machine to another without heavy changes in the source code. PFMS is a tool which provides a screen interface which will run under UNIX as well as VMS, RSX11-M or RT11.

PFMS is modelled after a widely used system, FMS†, and implemented in Fortran 77.

DEC's FMS is a program to build, maintain and use screen forms on a VT100 or similar terminal. The FMS interface may be used from programs written in different programming languages, the user interface to FMS is language independent.

Interaction with the user program is via subroutine calls which specify the name of a form to be used, the name of a field within the form to be manipulated, and a string where the input or output value of this field is deposited. The actions and the placement of the cursor are specified by so-called field terminators in the FMS interface calls.

Item 26: 9.30am **Rob Pike, Bell Laboratories**

Unix style or *cat -v* considered harmful

I quote from Rob's abstract:

'UNIX is spreading rapidly throughout the commercial computing world. This spread is due largely to UNIX's portability and practicality. Its original popularity is the academic computer science community, however, was due to its embodiment of a number of new and simple ideas, cleanly implemented.

The passage of time and programmers over every line of UNIX source code has made the file system more complicated to learn, to use, and to maintain, and much bigger - the current VAX kernels are about a factor of 10 larger than the 5th edition kernel, but certainly not a factor of 10 better. Most of that growth has not improved the system, but merely added to it.'

Rob's thesis of over complicated growth of the system started by discussing the **cat** command. **Cat** started as a simple command which copied its input to its output, it is a general file copying utility which can be use to concatenate files. However, its major use is probably to display a file on the terminal. It is a simple *command* and then along came Berkeley and made it into a *sub-system* by adding switches to it. The switches which were added were mostly related to **cat**'s use as a general program for screen output. Rob maintains that the switches are not required, since their action can be achieved in other more general ways. The switches are

-s strips multiple newlines, this can be done by **sed**.

-n numbers the lines, this should be done by **pr** or a special utility.

-b which numbers the lines but suppresses blank line numbering is just yucky.

-v prints non-visible characters in a visible way, this should be done by a special command.

Rob's contention was that the original **cat** did not alter it's input in any way and the -n, -b and -v switches simply contavened this basic rule.

Rob then switched his attention to the problem of mis-placed functionality. Most UNIX

† FMS is a trademark of DEC.

systems do not have any mechanism to handle screen scrolling. Instead of tackling this problem by perhaps inserting a terminal driver which did it† or buying a BLIT which has scrolling in the terminal. Berkeley tackled the problem by distributing it thoughout lots of utilities. The **ucbls** which normally pretends to be **ls** on most UCB systems is a case in point. It generates its output in columns, this can be done either by *echo* * or by using **pr**. However, the program now has a problem, in that when you type something like *ls|wc*, you want it still to generate the correct answer. So, **ls** now looks at its output stream and says 'If this is a terminal, I will output in multiple columns; else I will output in a single column as before.'

A second case of this problem is illustrated by the **more** command. This command started out as a program to perform screen scrolling. The problem was that it found its way into other UCB commands, notably the **man** command. There was a problem now when the **man** command was run to a file. So, now **more** looks at its output to find out whether it is talking to a screen or not.

Rob's contention was that a problem which should have been solved simply either by getting a terminal which scrolled or by changing the terminal driver has given rise to a large number of *ad-hoc* solutions which have been applied without any regard to the original problem. So, when you're fixing something make sure you are solving the right problem.

☞ Coffee (no Guinness, thank goodness) ☜

Item 27: 10.36am **Jaap Akkerhuis, Maths Centre, Amsterdam**

Typesetting and Ditroff

Jaap's talk started with a brief history of typesetting from Guttenberg to the present day. I did not really take notes on this bit and I hope he will write it out for the newsletter.

The original **troff** was severely limited by the CAT typesetter at which it was aimed. Device independent **troff** is a recoding of this original program to remove all the CAT dependencies and allow **troff** to be used with other typesetters and typesetter-like devices. Some extra graphics commands have been added to **troff** to allow pre-processors like **pic** to draw pictures in the text. Jaap's experiences with the new package have generally been good.

Item 28: 11.32am **Tom Horton, National Semiconductor Corp**

Virtual Memory management in GENIX

I reproduce the abstract, not because I was asleep, but because the talk was too technical to take adequate notes on.

'We have recently ported UNIX to a NS16032-based system. The port is based on 4.1BSD UNIX and is called GENIX‡. We discarded most of the 4.1BSD virtual memory support code and developed our own, in order to support the virtual memory architecture provided by the NS16082 Memory Management Unit.

A GENIX process has a virtual address space of up to 16 megabytes and is mapped by a two level hierarchy of page tables. Both process pages and second level tables can be paged out of and faulted into memory.

The kernel's address space is mapped by its own set of page tables. Because the user and kernel page tables have the same structure, the kernel manages its own address space with the same algorithm which is used to manage the user address space.

† As the UK people know, we have had a scrolling terminal driver for years because it is very annoying to see your output disappear up and off the screen

‡ GENIX is a trademark of National Semiconductor Corporation.

Sharing of texts is done by mapping executable files. When a file is executed for the first time, the kernel creates a file map for using level 1 and level 2 page tables. The process's page table entries are then set up using the file's map as a template.'

The overall impression is that the hardware looks to be correct and does not have the fatal lack of page reference bits which the VAX-11 processors suffer from. Also, because the hardware is about right, the software implementation can then be resonably elegant.

Item 29: 12.06pm **David Tilbrook, Imperial Software Technology**

Hacking vs Engineering - The UNIX Dilemma

David gave a vastly amusing talk with 35mm slides of Ada, Lady Lovelace and also pictures of sneakers. His main point, which I hope he will expound in these august pages, was that the majority of programmers in the world are working on extremely huge software projects and they really do not have the tools to cope adequately with what they are doing.

The set of UNIX tools which we are used to having around are not viable when scaled up to very large projects and committees which are set up to attempt to solve the problems of scale generate abominations like ADA.

He is getting interested in making tools to solve some of the problems associated with these large projects.

☞ Lunch ☜

Item 30: 1.45pm **Rob Pike, Bell Laboratories**

Video tape on the Blit terminal

The Blit terminal has a bit mapped display, a keyboard and a mouse. The software runs many UNIX shells in separate windows on the screen. The terminal allows local editing of input and output. The user interface is very elegant and watching the tape, it seemed to me that this was the way I want to talk to computers.

Item 31: 2.03pm **Chris Corbett, University of Essex**

Figure processing within Nroff

There are a couple of pre-processors which will allow the user to insert figures into text generated by **troff**. The main problem is that suitable output devices are extremely expensive. Chris has done some work to insert figure drawing capabilities into **nroff** where the output is on low-cost printers.

The system uses a new preprocessor called **fig**, which is used in the same way as other **troff/nroff** pre-processors. The program can either be used to insert (say) raster bit map images into the output from **nroff** or white space can be left if the output device will not draw.

The package can also incorporate images which were specified in **plot(5)** format. This is very useful because the images which are generated can be previewed easily.

☞ Tea ☜

Item 32: 3.35pm **Ed Keizer, Vrije Universiteit**

The Amsterdam Compiler kit

The Amsterdam compiler kit is an attempt to solve the problem of creating compilers for several languages for a wide range of different machines. The solution which has been adopted is to translate the language into an intermediate language (EM) which in turn is translated into code for the target machine.

EM was designed to be an efficient intermediate code for many languages (Pascal, C, Plain, Algol 68 and Ada) which easily runnable on very many target machines (8080, Z80, 6800, 6809, 8086, PDP-11, Z8000, M68000 and VAX). Generating target code is also easy since EM virtual architecture is a stack machine and all instructions have a single operand. It is also easy to add new machines and compilers because the programs which comprise the package are all table driven. There is also an EM optimiser which is machine and language independent. The package comes with good debugging tools and many test programs.

As distributed, the package consists of a version of **cpp** which can drive either the C→EM or Pascal→EM 'front-end'. This phase can be followed by an EM peephole optimiser and (in the future) an EM global optimiser. The EM is then fed into a number of backends which can generate code for the various target machines. This code is then assembled and linked with libraries.

The Pascal compiler has some deviations from the ISO standard: only the first 8 characters of a name are significant, standard procedures are not allowed as actual parameters in procedure calls, and the scope of a variable begins at the declaration. There are also some extensions to the standard: program modules can be compiled separately, assertions have been implemented, as have mark and release.

The restrictions on the C compiler are: the stack size must be less than 4K bytes, automatic variables must not occupy more than 4K and parameters must not occupy more than 4K bytes. Identifiers have 8 significant characters and external identifiers have only 7 significant characters. The standard says that the type of the **char** can be left to the implementor and for the Kit, **char** is unsigned.

The Kit is now available to academic institutions only for $500, although contacts to the address below from commercial sites will be greeted with the usual outstretched hand. The address is:

A.S. Tanenbaum,
Vrije Universiteit,
Dept. of Maths & Computer Science,
Postbus 7161,
1007 MC Amsterdam, The Netherlands.

Item 33: 4.35pm **Euug Committee**

Panel Session

This was a discussion on the conference, asking people what they liked and disliked about the proceedings and attempting to canvas opinion on what should go on a EUUG conferences. Again my notes are very thin but these two suggestions were made.

There was a *yes* vote for the starting of 'Birds of a Feather' sessions. There should be something in future conferences about artificial intelligence and functional languages.

The End

All that remains is to hand out the awards for the conference. Taurus Technology get *Tee Shirt of the Conference* award. For their tasteful little rose, Zilog get the *Best badge or Logo* award. Tobias from Holland gets the *Most revolting and obscene badge* award. Dave Tilbrook gets the *Most interruptions during presentations* award, this was a close thing. Guinness get the *Most drunk drink*

of the conference award.

Seriously though, thanks should go to Helen Gibbons and Tim Murphy for good organisation of a a good conference.

CALL for PAPERS

EUUG Spring Conference 1984, Nijmegen

IMPORTANT NOTE !!!

The CORRECT date for
the Meeting is

16th-18th APRIL 1984

The Spring 1984 conference will be held on 25th-27th April, 1984, at the University of Nijmegen in The Netherlands. Conference arrangements are being handled by the NLUUG, group IVV, KU Nijmegen, Toernooiveld 1, Nijmegen, Netherlands, tel +31 80 558833 x3125 or mcvax!kunivv1!sjors.

There is a call for papers and call for abstracts for less formal talks. The accepted papers will be published as official EUUG conference proceedings. The papers should be technical in nature and be of interest to the UNIX community. Suggested topics include (but are not limited to):

Systems	New UNIX kernel implementations, networks, network connections, distributed processing, real time kernels, kernel enhancements,...
Environments	Tools integration, embedded editors, new languages or developments,...
Applications	Database systems, graphics, mail systems, computer integrated manufacturing,...
Issues	Standards, security, reliability,...

For the call for papers, authors must submit an abstract of at least 1K words by the 15th of January 1984. The abstracts will be evaluated by a committee chaired by Dave Tilbrook. Abstracts must include a separate paper with title, name of author, institution or company and surface mail address (and network address). Abstracts written in machine readable form (ms-macro package) are preferred.

Abstracts can be submitted to the EUUG Secretary.

CALL for PAPERS

EUUG Spring Conference 1984, Nijmegen

Closing 15th January 1984

Benchmark Programs
Release 'A'.

Mike Banahan

RESTRICTIONS

Runtime requirements.

This software must be run on an otherwise idle machine. Use ps(1) to check that nothing but the normal system processes are running before starting a benchmark run. In particular, disable logins on all terminals but your own and kill off cron, update and so on. Virtual memory systems will of course still need the page daemon - process no. 2 in Berkeley implementations. If you can't run the programs on idle hardware then the results will not be of much use.

Components.

This is the simplest release of all of the benchmark software. There are three groups of tests: the cpu group which test processor and memory speed; the io group testing I/O rates; and the load group measuring performance loss under increasing load. The load group is even more dubious than the others and should be treated with much suspicion. Artificial loads are just that: artificial. The cpu tests are all written as short loops and exhibit high locality in their behaviour, so machines with cache will score very well on these tests. Machines with a mixture of fast and slow memory will give considerably different results on these tests depending on where the programs get loaded. Later benchmark releases will attempt to address these thorny problems.

The body of code involved is deliberately small. There could have been many more actual benchmark programs in this release: we aren't short of ideas, but we deliberately cut the number of tests to a minimum. The benchmarks will be no use if they are not adopted by everyone, and the results are easily compared. Few people will read page after page of benchmark results and manage to extract anything of use from them, so this release performs very simple measurement of some parameters which are expected to have considerable bearing on the performance of UNIX systems. The precise measurement of specific features, if it can be done at all, will only be found in later benchmark releases.

Running the benchmarks.

Create a suitable directory and install all of the benchmark software. There should be no problems running the software on standard V7 or System III UNIX systems - if there are, let us know. System V is an unknown quantity at the moment.

Check that the system date is set accurately - you will need this to check that the timing information is correct. The shell command file to automatically run the benchmarks is called 'run'. Start the run by typing 'run', there will be a delay while some programs get compiled. Take careful note of the first few lines of output - they will look like this:

```
check that this takes 40 seconds!
Thu May 26 09:57:03 BST 1983
a.out   40.00   0.00   0.06
Thu May 26 09:57:44 BST 1983
this too!
Thu May 26 09:58:05 BST 1983
a.out   41.00   39.48   0.10
Thu May 26 09:58:46 BST 1983
```

You may find that the lines shown in the example are preceded by a message from the 'make' command which is run first. In particular the message

'time_a' is up to date

will appear if you have already run the benchmarks once. It can be ignored.

The two lines containing 'a.out' should be bracketed by dates in which the two times are different by 40 (plus or minus one) seconds. The 'a.out' is followed by three figures: elapsed time, time spent in user mode and time in kernel mode for that process. The first should be as shown - almost exactly 40 seconds elapsed, practically no user or kernel time. The second should also be as shown - 40 seconds elapsed, 40 in user mode, negligible kernel time. (Although the first figure is shown with two digits after the decimal point, they will always be zero. The elapsed time is accurate to one second only: the meaningless extra precision is simply there for consistency with the other numbers and to aid possible post-processing of the results. The other two times are accurate to within one clock tick, whatever that means on your machine).

If the results that you get are different, there are two possibilities: your UNIX doesn't provide reliable timing information, or the header file 'bm_a.h' contains the wrong definition for the '#define HZ' line. The only likely possibility that we know of is machines with a 60 HZ clock instead of the 50 HZ used in the U.K. Try changing the definition and repeating the run - if you've cured the fault then great, otherwise something is definitely wrong.

If all looks well with those figures, reply 'y' to the question

all ok?

and the run will continue. Replying with anything else will just stop the run. At the end of the run, check that the date and time are still correct to make sure that the machine is getting its timing right, and cat the contents of the file called 'results' which will contain the results of the run.

Interpreting the results.

The results are arranged as Appendix A shows. These are the results of a run on an Onyx C8000 with 512kb of memory and a 10Mb winchester. The first line of the results file is rather interesting, telling you the size of the 'loop factor' for this machine. This is worked out by the benchmark suite, which finds out how often a simple loop is executed in a given period of time and then incorporates that figure in all the other tests. It is used to ensure that the tests have a sensible degree of self-sizing about them; if the cpu-bound loops take less than a few seconds, then there is a considerable element of inaccuracy introduced into timing them. With a wide variation in basic cpu cycle times across different machines it is not possible to build a fixed number of loop traverses into the benchmarks. The loop factor is intended to compensate for this, and is accounted for when printing the execution times.

The cpu and i/o results are reported in columns with appropriate headings. The meanings of the column headings are as follows:

prog	name of benchmark program.
actual	actual real time taken.
real	real time divided by loop factor.
utime	time spent in user mode divided by loop factor.

stime time spent in system mode divided by loop factor.

The 'real', 'utime', and 'stime' columns have had the loop factor included in their calculation and can therefore be directly compared with the results of runs on different machines. For the simplest of comparisons between runs, the 'real' column is the one to look at.

Incidentally, because of the way that the measurements are done, don't expect the 'real' figure to be exactly the 'actual' figure divided by the loop factor. They ought to be pretty close, though - the 'real' figure will simply be contaminated by the time taken to load and start the test, the 'actual' isn't. The longer the test runs, the less the apparent difference will be.

There will be one line present corresponding to the program name of '/bin/cc'. This line does not contain any loop factor dependency and simply reports the total of the times taken by the 'cc' command to compile the benchmark programs themselves. It gives a rough guide to the performance of the C compiler on this particular machine when compiling short programs. The figures will probably be dominated by the time taken to load the compiler, assembler and loader phases from disk, and not reflect the speed of the compiler itself. Whilst it is hard to be sure just what the figures measure, they are so easy to collect that it would be churlish not to report them. At least they refer to the running of large, non artificial programs, unlike the benchmarks themselves.

There will finally be the results of the artificial load test. This is not a very good artificial load, but it gives some indication of the way that the system responds to a good thrashing. The test tries running first one, then two, then four, then eight etc. processes containing identical i/o and cpu demands. The figures reported are the real time taken, the number of such processes, and the relative time taken normalised on the result of running one process. Typically the figures improve with increasing load to begin with (the more you give the machine to do the better it can overlap i/o), then there is a more or less sudden collapse as overload sets in. The test stops when this happens to the extent of each process taking three or more times as long as expected. The test only goes as far as attempting to run 64 concurrent load processes, and warns with the message 'fork failed' if the system refuses to allow it that many.

What do the tests do?

The command 'make index' will provide a description of each benchmark program by extracting identifying comment from the source code of each one. It would be nice to use the SCCS tools to do this, but not everybody has access to them yet. Appendix B contains the index for the 'a' release. Obviously, if you want to find out exactly what each program does, then you'll have to look at its source code and work out what is going on.

Acknowledgements.

Although they may not realise it, many people have provided input to these benchmarks. More sadly, many may feel that they have themselves done more work and that their work has been ignored. We can assure you that this is not the case - much of the work involved in producing this release has been selecting what to exclude, not what to put in. The 'b' release will be a much larger collection of programs: think of this as the foundation, not the complete building!

Appendix A

Results from a run on an Onyx C8000, 512kb, 10Mb disc, 23 September, 1983.

loop factor 83

prog	actual	real	utime	stime
cpu_a_001	53.00	0.64	0.63	0.00
cpu_a_001a	31.00	0.37	0.36	0.00
cpu_a_002	157.00	1.89	1.89	0.00
cpu_a_003	13226.00	159.36	4.93	153.51
io_a_001	1049.00	12.65	0.16	6.13
io_a_002	758.00	9.14	0.14	4.29
io_a_003	201.00	2.43	0.13	2.29
io_a_004	667.00	8.04	0.14	7.78
io_a_005	4392.00	52.94	0.15	10.33
/bin/cc		256.00	65.08	49.44

load_a_001.c

31	load =	1 relative	= 1.0000
62	load =	2 relative	= 1.0000
116	load =	4 relative	= 0.9355
631	load =	8 relative	= 2.5444
3875	load =	16 relative	= 7.8125

Appendix B

Brief Description of Each Benchmark Test or Subprogram.

cpu_a_001.c	Simple integer arithmetic
cpu_a_001a.c	Integer arithmetic in registers
cpu_a_002.c	Integer arithmetic and function call
cpu_a_003.c	Real arithmetic
io_a_001.c	Pipe throughput test
io_a_002.c	Pipe throughput - short writes
io_a_003.c	Writes to null device; throughput test
io_a_004.c	Disk reads; throughput test
io_a_005.c	Disk writes; throughput test
load_a_001.c	Increasing artificial load. Probably the least valid of all tests.
sig_a.c	check signals arrive when expected
size_a.c	Calculate approximate speed of cpu; this is included in other tests
sleep_a.c	Check that sleep(2) times properly
time_a.c	Rewrite of time(1) to give consistent results and output format as needed

The United Kingdom Unix Systems User Group

John Shemeld, University of Kent

History

Thanks to the foresight and admirable international outlook of the small group that met in Glasgow in 1976, the EUUG now covers most of Europe. As Teus Hagen has pointed out, a European group cannot hope to cover adequately such issues as national keyboards, national research policies etc. and a number of National User Groups have been formed to fill this niche - the Dutch and Swedish were the first. The EUUG has in turn responded to this development by moving towards a structure based on representation from these National Groups.

The Problem of the UK

The EUUG meeting in Dublin was thus faced with a problem - who represents the UK? The obvious answer - to go away and form a UKUUG on exactly the same basis as the Dutch NLUUG - presents some difficulties. The membership of the EUUG is about 70% from the UK. The Officers and Committee are predominantly from the UK. Was there enough interest to sustain another group? It turned out that there was enough interest for about 20 people to turn up to a subsequent meeting in London organised by Dave Tilbrook at Imperial Software Technology on 23 September, to discuss the founding of a UK group.

Names and Aims

This meeting decided to adopt the obvious name - UKUUG (as in the title) - and identified the objectives of the group as:

1. To provide UNIX information services (membership and installation information, software availability, legal issues, press cuttings - were all mentioned)
2. To provide a forum for discussion.

Money

The meeting then ran into the problem of finance. The existing national groups collect ALL subscriptions, passing on 40% to the EUUG Central Administration. It seemed a little absurd to suddenly divert 60% of 70% (equals 42%!) of the EUUG income to this fledgling organisation - especially when the UK Group is in a position to use the services of the (UK based) EUUG Secretariat. The task of telling 70% of the membership that although they had joined one organisation (the EUUG) they were now being transferred to another (the UKUUG) was also considered to be not worth the hassle.

The Proposal is, therefore, that the UK Group should collect its fees through the EUUG Secretariat, calling on not more than (and probably a LOT less than) 60% for funding of UKUUG activities. It certainly offends against my sense of tidiness to form a National Group with an anomalous relationship with the EUUG, but practical considerations seem to dictate it. The final decision is the EUUG Board's.

Internationalism

The old international outlook was much in evidence again when the question of Southern Irish members was considered. It was felt that National Groups would have to concern themselves with one National Government each, no matter how artificial or illogical its boundaries might be but, judging by the tone of this meeting, Southern Irish members need have no fear that they will just be left to fend for themselves as best they can.

Volunteers

Four volunteers were sought - and found - to form a interim committee to organise the bootstrapping of the new organisation at a one day meeting. This has now been fixed for December 16. They were:

Duncan Campbell	Meeting Organiser for the first meeting at Westfield College, London.
Mark Rafter	Interim Chairperson
John Shemeld	Interim Secretary
Simon Kenyon	Interim Information Officer

The present position is that the first meeting has been organised and the program is being circulated, a draft constitution is under discussion and will be presented to December's meeting, and what happens next depends on you.

EUUG Services

Teus Hagen

EUUG Distributions

The EUUG is now distributing five tapes. The first three (EUUG D1 - EUUG D3) have licensed material, so at least a UNIX Version 7 (or System III, System V) source licence is needed for these tapes. The following distributions are available:

EUUG D1 R6 — The EUUG D1 Release 6 distribution is a UNIX V7 system specially made for small DEC PDP's (11/23, 11/40, etc). The kernel supports the UK terminal driver. A paper describing how to boot from tape is supplied. A copy of the source license agreement for at least UNIX Version 7 should be enclosed with the request for distribution. The distribution is a 2400 feet tape (1600 BPI).

EUUG D2 — This distribution tape has the early Pascal compiler of the Free University in Amsterdam. A copy of the source license agreement for at least UNIX Version 7 is needed.

EUUG D3 R2 — This distribution has the UNIX networking software: news release 2.10.1, fully updated with bugs found up to date of shipment, and some other auxiliary programs. The tape contains also the news received on the Continent for the last year.

If a copy of the source license agreement for at least UNIX Version 7 is included with the request, a fairly debugged version of UUCP with some accounting enhancements will be added to the distribution tape.

The distribution is a 2400 feet tape (800 or 1600 BPI).

EUUG D4 — This distribution tape has the main UNIX Software Tools. The tools are sampled by the STUG (Software Tools Users Group). Most of the software is written in Ratfor, for which a Fortran support tool is included. If you want to have UNIX tools available on your non-UNIX system this tape is a must.

The tape is available in different formats: DEC RSX, DEC VMS, UNIVAC, IBM VMS, UNIX tar, MIT (Machine Independant Tape) line feed format, MIT card format (80 columns). The distribution is a 1200 feet tape (800 or 1600 BPI). Please mark what kind of tape format is needed (MIT line feed format is default). The STUG group receives a $10 profit per tape from the EUUG automatically.

EUUG D5 — This 600 feet tape contains a collection of benchmark programs. The software is in the public domain. No license agreement is needed to receive this distribution.

To receive an EUUG distribution the following is needed:

- your address (and institutional order number)
- the tape number (and format identification for EUUG D4)
- a copy of your source license (EUUG D1-D3).

Please do not send an exchange tape to us, this will only increase delays and postage costs! As soon as your request is received and checked for licensing and EUUG membership the tape is sent out (normally within two days). A bill for tape and postage costs will be sent separately (20-30 UKL).

Requests for EUUG distributions can be sent to the EUUG secretary or the EUUG distribution centre:

EUUG Distributions
Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam
Netherlands

EUUG UNIX Micros Catalogue

The EUUG UNIX Micros Catalogue (edition July 1983) is available now from the EUUG Secretary. The book costs UKL 5.- (UKL 10.- for non members). The catalogue gives a detailed description of all the UNIX computers available on the market.

EUUG UNIX Software Catalog

At this moment we are busy to compose an inquiry form for setting up an UNIX Software Catalog. The plan is to send this form to software houses in Europe. To keep the price down it is possible to have an advertisement in this catalogue. We shall try to have the catalogue published for the April EUUG conference in Nijmegen. If you want to have your software product(s) published in the catalogue please send your request for the inquiry form to:

Teus Hagen
ref. Software Catalog
Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam
Netherlands

or to the EUUG Secretary.

Circular UNIX

Timothy Murphy

Trinity College Dublin

Is there any way of writing "personalised" circulars or form letters under UNIX? Perhaps refer can be used for this purpose? I found it impossible to hack my way through the refer documentation, which is impenetrable even by UNIX standards.

We have been using the little program form.c below to send circulars. The usage is

```
form letter < list
```

The file letter contains the form letter, with macros to be expanded signalled by '%' followed by the 1-character macro name. Here is an example:

```
%T %N
%A

Dear %F,

    I am writing to invite you
to an  Ecumenical Bottle Party.
...
```

The list file contains the macro expansions for each recipient. The different "records" are separated by a blank line. For example:

```
%O +
%T Rev I
%N Paisley
%F Ian
%A House of Commons
London

%O +
%T Mr G
%N Adams
%F Gerry
%A House of Commons
London

...
```

Only those letters with a "+" in the macro %O are output. This avoids large-scale surgery on the list-file. For example, some letters inevitably get screwed up. So after printing, the "+"s are all changed to "-"; the few baddies are then changed back to "+", and form re-run.

Two additional "features".

Firstly, the 3 characters %Xn (where n is a digit) in the form letter denote the nth line in the macro %X. So one might write the address in the letter above as

```
%T %N
%A1
%A2
%A3

Dear %F,
...
```

This has the advantage of keeping the page length constant. If the address has only 2 lines, the 3rd line is blank.

Secondly, it is sometimes necessary to make allowance for the length of the expanded macro, eg if the macro appears in the left-hand of 2 columns, or if it is boxed. The character "~" is used for this. It will be "swallowed" until the "debt" on the current line is paid off; thereafter the "~"s are replaced by spaces. For example, an address might be boxed with

```
%T %N ~~~~~~~~~~~~~~~~~~~~~~~~~~
%A1 ~~~~~~~~~~~~~~~~~~~~~~~~~~~
%A2 ~~~~~~~~~~~~~~~~~~~~~~~~~~~
%A3 ~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

The use of the "filler" character ensures that the right edge of the box will remain in the same place.

One final point. The C-preprocessor cpp can be used to save space in long repetitive lists. For example, suppose the list contains a large number of MPs. Then a header might be added:

```
#define HC House of Commons\London
%O +
%T Rev I
%N Paisley
%A HC

%O +
%T Mr G
%N Adams
%A HC

...
```

The usage is now

```
/lib/cpp -P list | tr '\ \' '\
' | form
```

(Of course all this can be put in a command file. But there seems no way of expanding macros including newlines with the C-preprocessor; hence newlines are represented by '\' in the macros, and then translated into newlines with tr.)

Any suggestions for alternative approaches, or improvements, gratefully received ...

Listing of form.c

```
#include <stdio.h>
#include <ctype.h>

#define MACRO '%'
#define FILLER '~'

FILE *form;
char store[01000],*ptr=store,*mac[0200];

main(argc,argv)
  char **argv;
{
  form = fopen(argv[1],"r");
  while(gets(++ptr) != NULL){
    switch(*ptr){
      case '\0':
        writeform();
        ptr = store;
        break;
      case MACRO:
        mac[*++ptr] = ptr+2;
        ptr += strlen(ptr);
        break;
      default:
        *(ptr-1) = '\n';
        ptr += strlen(ptr);
    }
  }
}

writeform()
{
  register char ch;
  register char *str;
  register debt=0;

  rewind(form);
  if(index(mac['0'],'+'))
    while((ch=getc(form)) != EOF)
      switch(ch){
        case MACRO:
          str = mac[getc(form)];
          debt -= 2;
          if(isdigit(ch=getc(form))){
            for(debt--;--ch>'0' && str;)
              if(str=index(str,'\n'))
                str++;
          }
          else
            ungetc(ch,form);
          if(str)
            while(*str && (ch!='0'||*str!='\n')){
              putchar(*str++);
```

```
                debt++;
              }
            break;
          case FILLER:
            for(debt--;debt<0;putchar(' '),debt++);
            break;
          case '\n':
            debt = 0;
          default:
            putchar(ch);
        }
    for(ch=' ';ch<'~';mac[ch++]=0);
  }
```

Joining EUNET in the UK.

Mike Bayliss

Computing Laboratory
University of Kent

Introduction.

After the article "How to Connect to EUNET" appeared in the last issue, many sites have asked me the same questions about hooking into UKNET[1]. This article is an attempt to answer some of the questions, and hopefully make life easier for sites in the UK planning to join the network.

Equipment.

Modems:

For 300 baud transmission the British Telecom No. 1 modem can be used.

Diallers:

The diallers that we can help you support are:

i) Racal VA811S + VA831

ii) Master Systems IAD

This should not be interpreted as *any other dialler will not work,* but as *if you use any other type of dialler you could have problems.* There are other types of diallers available, which are cheaper, and **might** work, one example being the Racal MPS1222AD, which is supposed to be exactly the same as the MPS1222 + VA811S, but cheaper. We expect to have code for this dialler soon.

When you talk to suppliers be very careful, one supplier forgot to mention the V25-V24 converter and the customer thought he was getting a real bargain!

If enough sites are interested, and everybody decides to use the same type of modems and diallers we can try to arrange some sort of bulk ordering.

Software.

In the near future we hope to have a UK software tape put together. This will include the EUUG news distribution, the news batching system currently used in the UK, yet another version of UUCP and some network monitoring utilities. There will also be information on how to use one line for both in-coming and out-going calls.

The new version of UUCP is the main reason for producing this tape. It will contain a mixture of the UUCPs currently running at Edinburgh, Kent, Trinity College Dublin and Warwick. It will include a reworked conn.c with support the diallers mentioned above, the performance enhancements distributed with news 2.10, an accounting package, various bug-fixes and the startling new innovation — protocols that work over X25 networks.

Who do I contact?

Initially, you contact me at the address given in the back of the newsletter.

I'll put you in touch with the closest site to you, and make sure you get as much of the software as licenses allow me to let you have.

What will it cost?

Once you've bought your modem and dialler the largest part of the cost will be your phone bills, but you can (British Telecom willing) get something called *A Midnight Line.* This means that all inland calls you make between midnight and six a.m. are free, but you have a larger annual rental. Before you rush off to get your midnight line however, check that the sites you want to call are

1. Yes, we've given the UK part of EUNET its own name!

prepared to take calls in that timeslot.

Also, you'll be asked to pay your fair share of the Trans-Atlantic and European connection costs. Currently we estimate this will be about 20 to 30 pounds a month with the current size of UKNET and volume of transatlantic traffic. Obviously as the net gets larger, this cost will drop.

The site you connect to might make some extra charges as well, particularly if they are handling a large number of other sites and have considerable overheads. Typically they'll ask for a connection fee to cover the work involved in getting your link running and a regular service charge to cover their overheads.

In addition, if you're starting from scratch you should allow about 2 man-days work from a reasonably competent programmer, and probably some time for technicians to sort out hardware problems with your modem and dialler.

X25

Earlier I mentioned X25 support in the new UUCP, what exactly are we planning to support?

The X25 protocols will let you connect to another site through an X25 PAD, and use UUCP in the same way as it is used over phone lines. Obviously both sites must have the new UUCP, however, the same code will work on normal dial-in lines.

If you're currently on X25 and can find a mail path to *UKC* from your system we can get news through to you now. We hope to be able ship news over X25 using NIFTP, and the changes to news for that will hopefully be on the UK tape.

PSS, SERCNET, JANET, Grey Book and NRS.

If the heading of this section means nothing to you, then consider yourself to be one of the fortunate few and skip this section.

There are now two separate networks in the UK, the network I've talking about so far and the X25 network of which JANET (Joint Academic NETwork) is a part. (JANET includes SERCNET and academic sites that used to be on PSS.) If you are on the X25 network you'll probably know already that there are standards for your network that are nothing like the rest of the world, in particular Grey Book says your mail must conform to a standard based on ARPA standards, but different enough to pose problems when crossing network boundaries, and NRS says your name is backwards compared to the rest of the world.

There is already a primitive gateway between the networks running at UKC but, we don't expect to have a fully reliable, full function gateway for several months. The gateway software will be made available to anybody with the appropriate licenses, when it is working. However, the gateway software will be large and consume vast amounts of processor and disc space, and unlikely to fit on anything smaller than a VAX.

So our plan is to improve on the current mail gateway at *ukc*, and then run it at *edcaad* as well. This means that you won't need to worry about doing your own gatewaying (but there's nothing to stop you if you want to) and you'll only see UNIX standard mail and news.

Off the Net

These articles appeared on the UNIX network. Some of the boring header info has been taken out and a little formatting added.

From: *rbbb.rice%rand-relay@sri-unix.UUCP*
Newsgroups: *net.unix-wizards*
Subject: *Re: 2nd Unibus request revisited*
Date: *Sat, 1-Oct-83 18:51:15 GMT*

From: David.Chase <rbbb.rice@rand-relay>

This is Mike's account of the changes. They were written by Mike Caplinger and Scott Comer ("wert"). These guys have both dropped unix wizards because of the high volume and low content, and so must be contacted directly or through me.

drc

This is a really abysmal presentation of what changes to make to add dual Unibus support for 750s. I would have done a diff, but the original source is gone and I'm too lazy to get it back.

What we're doing here is adding a second group of 128 Unibus interrupt vectors to the end of the scb. This is put in with the changes in locore.s and scb.h. When the vectors are set, we need to tell the 750 it can use two Unibi; these are the changes in autoconf.c. Finally, we make the cosmetic change in Locore.c, though this is only for lint's benefit.

All the references to UNI1vec and uba1int are new and need to be added. I've tried to include enough context to make the changes locatable. Feel free to call me if you have problems, and let me know if it works.

- Mike Caplinger
(713) 527-9964
mike.rice@rand-relay

---

autoconf.c in sys/dev:

```
        /* Did something special for the second unibus on 750's   */
        /* (really, just allocated another fixed interrupt vector */
        /* table at the end of the scb).          wert 11/20/82 */

        switch(cpu) {
                case VAX_780:
                        if (numuba == 0)
                                uhp->uh_vec = UNIvec;
                        else
                                uhp->uh_vec = (int(**)())calloc(512);
                        break;

                case VAX_750:
                        if (numuba == 0)
                                uhp->uh_vec = UNIvec;
```

```
                        else if (numuba == 1)
                                uhp->uh_vec = UNI1vec;

                        else
                                printf("No support for more than 2 UBAs on a comet0);

                        break;
```

---

locore.s in sys/sys (this is used to be +8, I think):

```
#define _rpbmap_Sysmap+12                    # scb, UNIvec, rpb, istack*4
```

Locore.c in sys/sys:

```
int     (*UNIvec[128])();
int     (*UNI1vec[128])();  <----

        printf("Write timeout");
        (*UNIvec[0])();
--->    (*UNI1vec[0])();
```

scb.h in sys/h:

```
        int     (*scb_ipl17[16])(); /*  "           "   IPL 17 */
        int     (*scb_ubaint[128])();       /* Unibus device intr */
--->    int     (*scb_uba1int[128])();      /* Unibus device intr */
```

****
Note changes added 8/2/83!
****

ubavar.h in sys/h:

```
        extern  int (*UNIvec[])();          /* unibus vec for uba0 */
--->    extern  int (*UNI1vec[])();                 /* unibus vec for uba1 */
```

scb.s in sys/sys:

```
                .globl  _UNIvec
        _UNIvec:        .space  512             # 750 unibus intr vector
                                                # 1st UBA jump table on 780's
--->            .globl  _UNI1vec
--->    _UNI1vec:       .space  512             # 750 second unibus intr vector
```

From: *msc@qubix.UUCP (Mark Callow)*
Newsgroups: *net.unix-wizards*
Subject: *4.2 is finally here*
Date: *Tue, 4-Oct-83 20:29:04 GMT*

Our copy of the 4.2bsd distribution has just arrived via Federal Express. Look for yours any day now. There are 2 tapes, a floppy disk, a data cartridge (for a 750) and a huge box of documentation.

I'll let you know how it goes. We will start converting our Vax in about 2 weeks.

--

Mark Callow, Saratoga, CA.
...{decvax,ucbvax,ihnp4}!decwrl!
...{ittvax,amd70}!qubix!msc
decwrl!qubix!msc@Berkeley.ARPA

From: *jim@mcvax.UUCP (Jim McKie)*
Newsgroups: *net.unix-wizards*
Subject: *Re: 2nd Unibus request revisited*
Date: *Wed, 12-Oct-83 15:16:36 GMT*

There is something missing here, and that is the definition of how many UBAs are allowed on each processor type. That goes in ubareg.h, where the code should look something like

```
        #define NUBA780     4
--->    #define NUBA750     2           /* Used to be 1 */
        #define NUBA730     1
        #if VAX780
        #define MAXNUBA     4
        #else
--->    #ifdef VAX750
--->    #define MAXNUBA     2           /* Used to be 1 */
--->    #else
        #define MAXNUBA     1
--->    #endif VAX750
        #endif
```

MAXNUBA is used in a number of places, for array sizes in drivers, but most noticeably in allocating space in the system page table for access. So it will still crash on boot.

You can get a GENERIC system to boot on a 750 with two UBAs if there are no devices on the second UBA.

****NOTE****
Although 4.2BSD has very similar code to support two UBAs (the RPB is also moved to before the SCB so it is in a fixed place), it also has the old code in ubareg.h, at least in the pre-release version.

We found this out after the 4.2BSD pre-release testing was over,

so this may be the first 4.2BSD bug on the net?

Jim McKie Mathematisch Centrum, Amsterdam ..{decvax!philabs}!mcvax!jim

From: *keld@diku.UUCP (Keld Simonsen)*
Newsgroups: *eunet.general*
Subject: *KULUUG Univ. of Copenhagen group formed*
Date: *Thu, 13-Oct-83 01:34:53 GMT*

A local Unix users' group at University of Copenhagen has
been formed. Its name is KULUUG. KULUUG has the following
members:

Institute of Datalogy (DIKU)
Indre By-terminalen (IBT)
Institute of Phonetics (IPUC)
Institute of Applied Matemathical Linguistics (IAML)
H. C. Oersted Institute Edp-department (HCO)
Regional Edp Center of University of Copenhagen (RECKU)

The group will have about two meetings a year.
The contact person is Keld Simonsen, mcvax!diku!keld

From: *mike@taurus.UUCP*
Newsgroups: *net.general,net.unix-wizards*
Subject: *Benchmarking Package*
Date: *Thu, 20-Oct-83 15:07:41 GMT*

The European Unix Systems User Group (EUUG) announces the distribution
of its benchmarking package. This is a suite of programs designed to
provide a standard for performance evaluation of UNIX systems.
The package contains a number of programs which test various aspects of
system performance; these programs are compiled and run, the time taken
to run them is noted and reported in a standard form which can be directly
compared with benchmark runs on other machines or implementations.

The software is available to members of the EUUG - enquiries to mcvax!teus.

The EUUG will collate results of benchmark runs and publish them as
aprropriate. Occasional updates will also be made to the package as it is
felt necessary. If you want more details, I will reply by mail unless the
number of enquiries is high enough to make it worth broadcasting a more
detailed description on the net.

Chairman of the EUUG benchmarking committee,
--

Mike Banahan
{ENGLAND}!ukc!root44!taurus!mike

From: *keld@diku.UUCP (Keld Simonsen)*
Newsgroups: *net.unix-wizards*
Subject: *Bell rescinds Educational S licenses*
Date: *Thu, 20-Oct-83 13:03:05 GMT*

This should have gone to nel.usoft.s or net.general,
but we don't get them here ...
Bell has recently rescinded our Educational software
agreement on the statistical software package S,
and offered us a real Educational license instead.
This will cost US $ 400.- *pr* *CPU* instead of normal
Bell Educational licenses which cover all CPUs of an
institution. There is no initial scheme like System V
where you can have all your CPUs on the initial agreement
for the same fee, and the additional fee is the same (US $ 400)
as the initial fee (I reckon they will see a lot of initial
requests and not so many additionals -).
Well, it was all in the agreement, with small letters,
but you certainly can feel the overwhelming charms of Ma Bell!
And you can imagine what will happen when Bell unbundles !

Keld Simonsen, DIKU, University of Copenhagen

From: *ber@enea.UUCP*
Newsgroups: *eunet.general*
Subject: *The Swedish Unix systems Users Group*
Date: *Sun, 23-Oct-83 22:31:45 GMT*

The Swedish Unix systems users group has now been established.
The group is called EUUG-S, which is sightly different from
the standard for other national groups, but the name SUUG
was already used by the Swedish Univac users group.

The first annual meeting was held in Stockholm on October 19,
with Teus Hagen as invited speaker. The group currently has
43 members (29 institutional and 14 individual members) and
also includes members from Norway and Finland.

Sites connected to Eunet has expanded from zero in April to
a total of 10 in October including a link to Finland.

Bjorn ("Chairman of The Swedish Unix Systems Users Group") Eriksen
mcvax!enea!ber

Letters

Dear Sir/Madam,

Our company has recently started to use Unisoft's Uniplus for in-house system development, and propose to use the same operating system as the basis for some of our OEM products.

It is my responsibility to provide user documentation for our customers, many of whom are not computer oriented, and I have not yet been able to find any documentation on system error messages. Although most of the books about UNIX which I have read claim that the brevity of the error messages is user-friendly, I suspect that short, punchy messages may not be much help to anyone who is not a professional programmer.

I would be most interested to know whether any other members of the User Group have encountered any problems decoding UNIX error messages, and whether any error documentation exists.

Glyn Watson, Product Documentation

Racal-Redac Ltd.
Newtown, Tewkesbury
Gloucestershire GL20 8HE
England

tel: +44 684 294161

Dear Jim,

Last Thursday saw the inaugural meeting of the informal London area UNIX local user group. The main business of the meeting (it transpired) was the consumption of a substantial amount of beer, followed by a curry and then some more beer. In view of the success of the meeting in completing its agenda, the ad hoc solidarity committee voted to meet on a more regular basis. Such meetings are now to be held on the last Thursday of each month, in the Lyric pub, Great Windmill Street, London W1 (just off Shaftesbury Avenue, about 200 yds. from Piccadilly Circus).

The business in these meetings is NOT meant to be of a very serious nature, but all are cordially invited to attend; deadly hacking is to be discouraged, but a spirit of (very) free and easy discussion is expected to pervade. Meetings start at 17:30 local time and generally stop doing anything useful at about 21:00. The business of the meetings is usually complete at 23:00, although the meeting's annexe (Bow Street Magistrates' Court) may be called into use to arbitrate on particularly contentious points at issue.

Mike Banahan (Convenor)
Taurus Technology

Some Important Addresses

The most important address is that of the EUUG Secretary. From there you can probably get information on anything and anyone, like how to join, who you can turn to for problems, etc.

EUUG Secretary
Owles Hall
Buntingford
Herts SG9 9PL
England

tel: +44 763 71209
UUCP:

Any articles or material for the Newsletter, except for advertising which is handled by the Secretary above, should go to

Jim McKie
Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam
The Netherlands

tel: +31 20 5924147
UUCP: mcvax!jim

NOTE: unless it comes in machine readable form, it has little chance of being included. What do you have a computer for anyway?

There are now some National UNIX Systems Users Groups in existence. The addresses of who to contact are

For Denmark (DKUUG)

Keld Simonsen
Institute of Datology
Sigurdsgade 41
DK-2200 Copenhagen
Denmark

tel: +45 1 836466 x14
UUCP: mcvax!diku!keld

For the Netherlands (NLUUG)

Marten van Gelderen
NIKHEF-K
Postbus 411
1009 AJ Amsterdam
The Netherlands

tel: +31 20 5922030
UUCP: mcvax!ikogsmb!csg!marten

For Sweden (EUUG-S)

Bjorn Eriksen
Box 232
S-182 23 Taeby
Sweden

tel: +46 8 7567220
UUCP: mcvax!enea!ber

For the UK (UKUUG)

John Shemeld
The Electronics Lab.
University of Kent
Canterbury CT2 7NT
England

tel: +44 227 66822 x285
UUCP: mcvax!ukc!jds

EUNET, the European UNIX Network, now spans some 12 European countries with electronic mail and news, and has links to North America. For information on how to hook up, contact

In Continental Europe:

In the UK:

Teus Hagen
Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam
The Netherlands

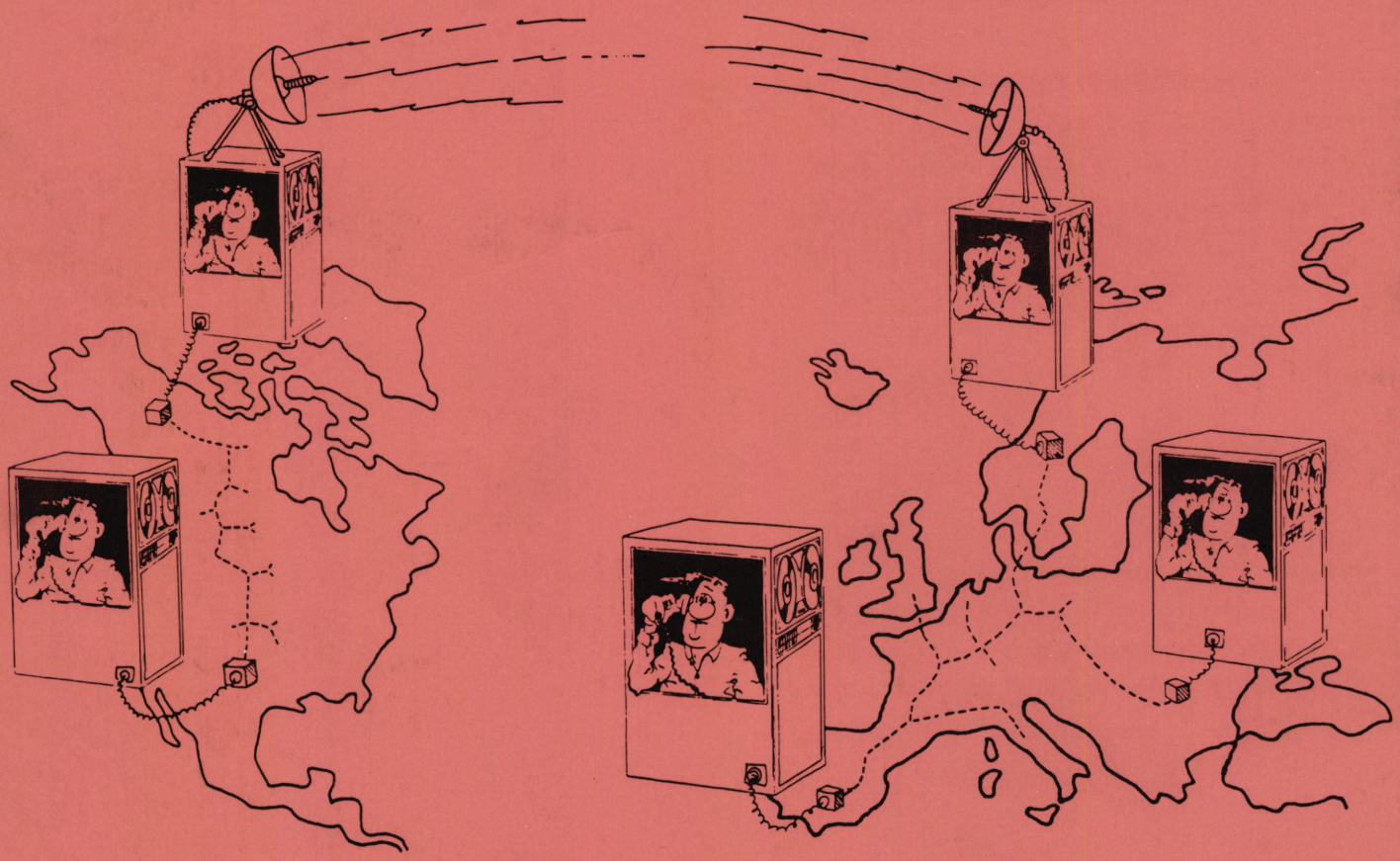
tel: +31 20 5924127
UUCP: mcvax!teus

Mike Bayliss
Computer Laboratory
University of Kent
Canterbury CT2 7NF
England

tel: +44 227 66822 x7615
UUCP: mcvax!ukc!mjb

EUNET LOGICAL MAIL MAP - November 1st, 1983

```
decvax   philabs
   |       |
   |       |                                          sus suadb erix
   |       |                                           |    |    |
   |       |                                           +==enea===+-taycs
   |       |                                           |    |    |
   |       |             tnoa68-tnod68 uvapsy       forward |   ttds
   |       |                |            |                |    |
   |       |         ics24 |     mcpdp34 | uvacs       kvport |  ttdsa
   |       |           |   |      |  | | |   |          |    |
   |       |   dutesta |   | mcpdp45 | | |   |    kunivv1 |    |   ibt
   |       |      |    |   | | |    | | |   |       |    |    |    |
   |       |  vub |    |   | bartok | |   | riv02 |    |    |   diku
   |       |   |  |    |   |    |   | |   |   |   |    |    |    |
   +=======+===+==+==+=+=+=+++++==mcvax=+=+=+=+==++===+=====++===+====+
               |     |   |  |||    |       |    ||   |    ||   |    |
               |     | IM60 |||  sara70 ikogsmb || nlgvax  ||   |    |
               |     |      |||    |       |    ||         ||   |    |
+---vax135     |     |      ||+---vu44    csg   ||         ||   |    |
|     |        |     |      ||    |  |          ||         ||   |    |
|     |        |   philmds  || vu45--vu60       ||         || zti1   |
|     |        |            ||                  ||         ||   |    |
|     | qtlon  |            ||                  ||         |+-pcsbst |
|     |   |    |            ||                  ||         |         |
|++===+===+=edcaad===+===+=+||                  ||         |         |
|||   |              |   | |||                  ||       unido       |
||| caad24 csastand dice | ||+------ircam       ||       | | |       |
|||             |        | ||                  || unidoi1 | unidoi3 |
||| glhocus csbstand     | ||    smsol  iiecnam ||         |         |
|||      |      |        | ||      |       |    ||      unido60      |
||| gl45-+==+=glasgow=+  | ||      +====vmucnam=+|                   |
|||         |         |  | ||      |       |     |                   |
||| edee60  | edmirv  |  | ||   lsicnam sm0cnam  |                   |
|||   |     |    |    |  | ||                  |                   |
||+--edee===+-edmiru  |  | ||                  |                   |
||          |    |    |  | ||                  |                appias
|+-logica  edai-edmrc=+==+ ||                  |
|           |              ||                  |
| west44--root44-ist       ||                  |
|   |       |              ||                cern45
| westf     |   ru-cc44    ||
|           |      |       ||
| ucl-cs    |   ru-cs44    ||
|   |       |      |       ||
|   | roger | regi |       ||
|   |   |   |  |   |       ||
+===+===+==ukc=+===+=======++
    |       |      |
 tcdmath  comet  hirst1
    |
 tcdvax
```



The Secretary
European Unix® Systems User Group
Owles Hall
Buntingford, Herts.
SG9 9PL.
Tel: Royston (0763) 73039.