AUUGN

Australian Unix User Group Newsletter

Volume 5
Number 3

Registered by Australia Post Publication No. NBG6524

The Australian UNIX* Users Group Newsletter

Volume 5 Number 3

June 1984

CONTENTS

* UNIX is a trademark of Bell Telephone Laboratories.

Editorial

Well, if the number of phone calls and letters that I receive is anything to go by, interest in UNIX is really hotting up. We now have nearly 200 subscribers, up from 30 at the beginning of volume 5, and 580 people on the mailing list.

AUUG Meeting in Melbourne

The preliminary announcement for the Melbourne AUUG meeting has been mailed out, but for those who may have missed out, here are a few details.

The 1984 Winter meeting of the AUUG will be held at the University of Melbourne on Monday August 27 and Tuesday August 28. Keynote speaker for the conference will be Rob Pike, from AT&T Bell Laboratories, co-author of the book "The UNIX Programming Environment".

Papers on any subject related to the UNIX system, or UNIX-like systems will be considered for the meeting. Prospective authors should send a detailed abstract (approximately a page) of their talk to Robert Elz, at the address below, before July 1.

Vendors desiring to display equipment should also contact Robert Elz as soon as possible to reserve space. Space will be allocated on a first come first served basis, and those who request early will obtain prime positions. Vendors should provide details of space, power and any other special requirements with their application. Charges will be based on facilities allocated and will be set individually for each display.

A final announcement will be mailed to people on the AUUG mailing list early in July. Registrants seeking early registration discounts, or reserved accommodation in University Colleges, should reply by August 1, 1984. Further information is available from

Robert Elz
Department of Computer Science
University of Melbourne
Parkville VIC 3052
Australia

+61 3 341 5225

UUCP: decvax!mulga!kre
AUNET: kre:munnari

Contributions

Thanks a lot to all the people who have allowed their arms to be twisted, and have contributed items to this and future issues. I plan to keep twisting, so watch out!

Opinions expressed by authors and reviewers are not necessarily those of the Australian UNIX Users Group, its Newsletter or the editorial committee.

Books

Does anyone out there know who publishes "A Programmer's Guide to UNIX", "A Business Guide to the UNIX System", "A Programmer's Guide to Xenix" and "A Business Guide to the Xenix System"? All are written by Jean Yates and Rebecca Thomas. Hopefully complete information will be available by next issue but for the moment, add the following to your list.

Related Books

5. Pascal Under UNIX
   J.N.P. Hume and R.C. Holt
   Reston Publishing Company

I also have not one, but TWO book reviews. Peter Mason has reviewed "Programming in C" by Stephen G Kochan, and Damian McGuckin has reviewed "Starting With UNIX" by P. J. Brown.

If you want to review some books, drop me a line.

A Review of
"Programming in C"
by Stephen G. Kochan
Hayden Book Company, Inc.

Reviewed by
Peter J. Mason
Australian Graduate School of Management

This is a book of rather large dimensions, 17.5cm by 24cm, and its 373 pages give a thickness of 2.5cm. It's pages open out fully and the book can be made to stay open without too much trouble, making study from it easy. Soft covered and moderate grade paper shouldn't make it too expensive. Unfortunately, no clues to the price came with the review copy, so value is something on which I am unable to comment.

I'd like to have had a text such as this when I was learning C. It treats the language as just another language with no particular reference to UNIX, though the operating system does get mentioned in the Introduction and several Appendices. It should be equally applicable to programming under any other operating system.

The book is most definitely for the novice and intended for classroom use with possibly a dozen or more exercises at the end of each chapter. Much effort has been put into giving good explanations in the body of the chapters, but this does not seem to have flowed over to the exercises with the same enthusiasm. The exercises are frankly boring. They range from "compare the output", and "modify the function", through to "write a program" type questions. For the most part, they're uninspiring, but do exercise aspects from their respective chapters.

All features of the language are investigated with (in most cases) sufficiently wordy explanation to satisfy the curious student, though of necessity, several forward references are made to tantalize at times. The

back cover tells me that there are over 90 examples included in the volume. These appear to demonstrate most of C's wonders and quirks.

This is not a text to inspire good programming habits and well structured design, but as a first course in C it is adequate. Each chapter handles one aspect of C and the challenge to write large programs is not made at all. Data structures beyond the most primitive may infuse the imagination while working through it, but it will be up to the lecturer to enforce some implementation of them.

I would like to present the table of contents here complete with beginning page numbers. This should yield some idea of the space devoted to each.

In summary, we have a good, very easy to read text for its purpose, being to familiarize to reader with the language C. The strength of the book lies in its easy readability. Does this sound like the book for you? It's a text book rather than a reference book, and excellent reading for the learner.

A Review of
"Starting with UNIX"
P.J. Brown
Addison Wesley (1984)

Reviewed by
Damian McGuckin,
Department of Civil Engineering Materials,
University of New South Wales.

Even though this book is aimed at beginners, it is one of those books which everybody should/must read at least once. It is not the sort of book on UNIX which would be kept as a reference, but definitely something everyone could use at some stage when dealing with UNIX. Unlike many books which give a detailed account of how to use UNIX, this book spends a lot of time going into the ideas and concepts behind UNIX. At the same time, it highlights the philosophy or way of thinking one should use to get the most out of UNIX. The author still raises the small-is-beautiful concept of UNIX, a feature that some argue is going out of vogue in the UNIX community of today. Finally, armed with this, one can then go out and (hopefully) use UNIX effectively, or maybe more effectively than one has in the past.

The book begins with covering such basic concepts as timesharing systems in general, files, the UNIX file system and the command interpreter. It then goes into more detail on the shell sh, the editor ed, the online manual, communication between users and document preparation. These are adequately documented with various "SAMPLE SESSIONS". There is a short cursory section on program development covering cc, pascal, f77, lint, make and basic although cursory is the operative word in this chapter of 12 pages, 2 of which are code examples. There is an excellent chapter on problems one encounters when using UNIX. It covers such areas as the affect of a system crash and what to do after it, hung terminals, screenfuls of garbage, what to do if caught in a program which seems to ignore interrupts, and what to do when encountering the (sometimes cryptic) UNIX error messages. Finally, there is the mandatory appendix with a short list of fundamental UNIX commands.

One could argue about the order in which certain topics are raised, (the editor appears halfway through the book), but in general this is really a book that needs to be read cover to cover and then read selectively to get the bits out of it that are wanted. A beginner using it will not feel intimidated as the book is extremely readable, and covers almost every question they may have. The author's wit is also evident as he discusses the seven deadly sins:

"Given that the **passwd** command discourages sloth in typing passwords, and covetousness of your password by others, and that **du** monitors your gluttony for file space, UNIX covers three of the Seven Deadly Sins with just two commands. Although you might argue that **diff**, with its apparently tortuous output, causes anger rather than curbs it, you at least have **man** to help assuage the anger. Envy can be curbed by denying read permission on files, on the basis that what the eye does not see the heart does not grieve for; anyone guilty of pride is reduced to humility when they first try to master **ed** or **nroff**. There are, however, no programs to curb lust."

An experienced UNIX user would benefit from the book just by seeing someone else's viewpoint on the system, and the guru can appreciate it as a manifestation of some of the fundamental philosophies of UNIX itself. It is

really the sort of book that gives you (almost) everything you need in the way of understanding and concepts to be able to master the UNIX programmer's manual.

As a closing remark, and one to highlight what is the author's insight into UNIX philosophy, the following is an excellent quote as to the friendliness (for whatever the word is worth these days) of UNIX.

"If you join a new group of people, the first friends you make are probably the more talkative ones: those who approach you readily and tell you everything you need to know about your new environment. As time goes by, you will find that the chatterers bore you and, after a while, become downright annoying. You make new and more solid friendships with the strong and silent types who perhaps were rather forbidding at first. So it is with making the acquaintance of operating systems. You will find UNIX among the strong and silent ones. UNIX is not especially friendly on first acquaintance, it is not unfriendly either - merely somewhat indifferent. As your acquaintance grows, you will find your friendship blossoms - maybe to become firm enough to last a lifetime."

Nets

The AUUG network database has recently been updated to contain the latest information from the US plus all Australian sites. Queries sent to "auugnetdb:elecvax" over the past few months will be rerun to provide the most up to date information.

The following AUNET sites have come onto the network or have changed the information previously published. A fairly recent network map is presented at the end of this section.

- -

Name: babel
Address:Department of Computer Science
University of Western Australia
Phone: +61 9 380 2878
Machine:
NCR TOWER 1632 (MC68000), 30 Mb Winchester, 8 ports.
Contacts:
Glenn Huxtable (glenn:wacsvax)

- -

Name: csb44
Address:Clinical Sciences Building
The Prince Henry Hospital
Little Bay. 2036
Phone: +61 2 661 6256
Machine:
PDP-11/44, 2x rl02 drives, 1x CDC winchester 80mb, 1x Cypher tape drive
Unix level 7 Ausam
Contacts:
Peter Goadsby (peterg:csb44)

- -

Name: dmsperth
Address:C.S.I.R.O
Division of Mathematics and Statistics
Private Bag
PO
Wembley WA 6014
Phone: +61 9 387 0325
Machine:
DE Unity
Contacts:
John Field (johnf:dmsadel)
Ron Baxter (ronb:natmlab)
Mark Palmer (markp:dmsperth markp:dmscanb)

- -

Name: mugana
Address:Department of Computer Science
University of Melbourne
Parkville VIC 3052
Phone: +61 3 341 5225
Machine:
Unison, V7 UNIX
Contacts:
Robert Elz (kre:munnari)
- -

Name: mulwala
Address:Department of Computer Science
University of Melbourne
Parkville VIC 3052
Phone: +61 3 341 5225
Machine:
Unison, V7 UNIX
Contacts:
Robert Elz (kre:munnari)
- -

Name: mummjeeli
Address:Department of Computer Science
University of Melbourne
Parkville VIC 3052
Phone: +61 3 341 5225
Machine:
Unison, V7 UNIX
Contacts:
Robert Elz (kre:munnari)
- -

Name: mundara
Address:Department of Computer Science
University of Melbourne
Parkville VIC 3052
Phone: +61 3 341 5225
Machine:
Plexus P60, System III
Contacts:
Robert Elz (kre:munnari)
- -

Name: munker
Address:Department of Computer Science
University of Melbourne
Parkville VIC 3052
Phone: +61 3 341 5225
Machine:
Unison, V7 UNIX
Contacts:
Robert Elz (kre:munnari)
- -

Name: murdu
Address:Computer Centre
University of Melbourne
Parkville VIC 3052
Phone:
Machine:
VAX11/750
4.2BSD
Contacts:
- -

Name: musette
Address:Department of Music
University of Melbourne
Parkville VIC 3052
Phone:
Machine:
Unison, V7 UNIX
Contacts:
- -

Name: muwe
Address:Department of Computer Science
University of Melbourne
Parkville VIC 3052
Phone: +61 3 341 5225
Machine:
Unison, V7 UNIX
Contacts:
Robert Elz (kre:munnari)
- -

Name: uacomsci
Address:Dept. of Computer Science,
University of Adelaide
North Terrace
Adelaide SA.
Phone: +61 8 228 5592 (Kevin), +61 8 228 5681 (Kelvin)
Machine:
VAX-11/750, 5Mb, 1*RL01, 2*RA81, 2*RX02, 8 DZ Lines, 24 DMF Lines,
LPS-11, DEUNA.
(+ Ethernet connection to campus LAN. NOTE: uacomsci is a VMS M/C)
Contacts:
Kelvin Nicolle (kelvin:uacomsci)
Kevin Maciunas (kevin:uacomsci)
- -

Name: uqelec40
Address:Electrical Engineering
University of Queensland
St Lucia
QLD 4067
Phone:
Machine:
PDP 11/40
UNIX V7 + PWB + AUSAM
Contacts:
- -

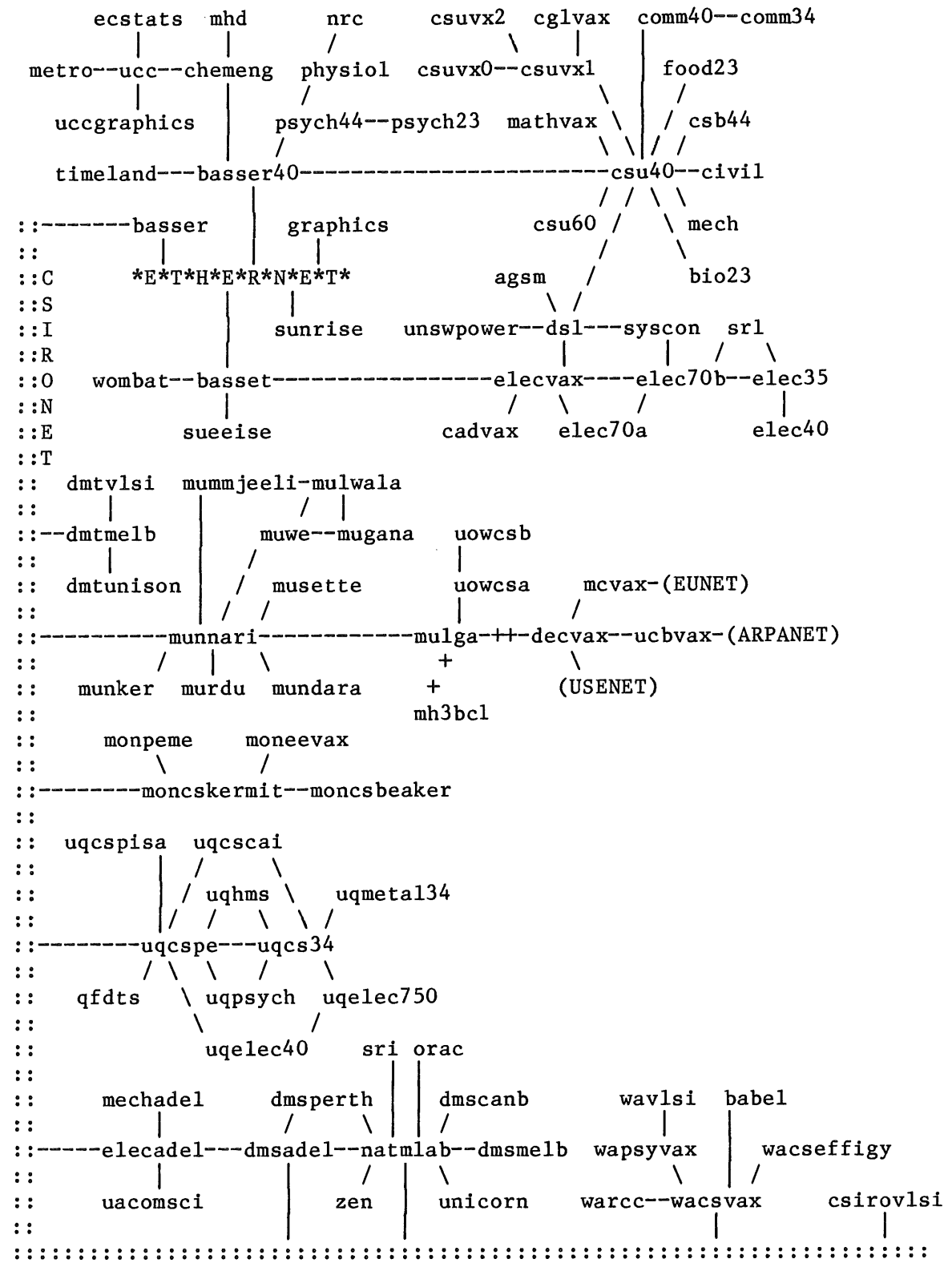
Name: uqelec750
Address:Electrical Engineering
University of Queensland
St Lucia
QLD 4067
Phone:
Machine:
VAX11/750
Contacts:
- -

Name: wacseffigy
Address:Department of Computer Science
University of Western Australia
Phone: +61 9 380 2878
Machine:
PDP 11/23 AUS V7 UNIX. (upgrading to 11/73 mid '84)
CDC 80Mb Winchester, DEC RX02 Floppy Disks, CYPHER streaming tape (tm11),
1xDZV11 4port serial mux, 3xDLV11 serial line
Contacts:
Glenn Huxtable (glenn:wacsvax)
- -

Name: wacsvax
Address:Department of Computer Science
University of Western Australia
Phone: +61 9 380 2878
Machine:
VAX 11/750 4.1zBsd UNIX, RA81, RA60, RL02, TS11, 5xDZ11, Console,
lp and lots of other junk, 3COM 10Mbit ethernet.
Contacts:
Glenn Huxtable (glenn:wacsvax)
- -

Name: wavlsi
Address:Electronic Engineering
University of Western Australia
Phone:
Machine:
VAX11/730 4.1zBsd
Contacts:
- -

Australian Computer Science Network - Bob Kummerfeld 7th May 1984



Unix and the PDP-11/60

Dave Horsfall
Computing Services Unit
University of New South Wales

(dave:csu60)

This article is just a brief summary of the various troubles I had in implementing Unix version 7 (species Elec Eng, UNSW) on a PDP-11/60. At the risk of being defamatory, readers may be familiar with the incomprehensible article on the subject in AUUGN Vol II No 1 (Oct-Nov 79), so I hope this missive is a little more practical. It is not yet finished, as a few aspects still need to be cleared up, such as accessing the micro-registers etc. This will be the topic of a future article.

The 11/60 was acquired some years ago with a view to having it replace the communications multiplexor for the EXPORT batch stations. There was the possibility of later upgrades to improve the performance by increasing the line speed, making it full duplex etc. For various reasons this scheme was abandoned, and the machine sat quietly for over a year, gathering dust and shifting a 1-bit in its display register from side to side. Also about this time, the CSU's 11/40 was slowly being brought to its knees under an ever-increasing load. Not only was it supporting the Unix network to eight other hosts, it also handled VMS/UNIX mail traffic, remote printer output from VMS, Versatec plotting and remote batch. The batch system in particular was used heavily by the network, servicing sites without a direct (or reliable!) batch connection. Oh yes, and while all this was going on there were people trying to use it as well.

Eventually approval was obtained to make the 11/60 the CSU's work-horse, and the 11/40 does nothing but run the network to ten other hosts and handle VMS/UNIX mail. Its disks are currently left-over RK05's, but plans are afoot to improve this. The old DJ-11 makes a fine multiplexor in this application by the way. When all the board-swapping had finished, the 11/60 booted first time (albeit single user) using the 11/40's system. There are no incompatibilities as such between the two CPU's; merely that full use was not being made of the 11/60. It can be regarded as no more than an 11/40 with genuine floating point (which I am told is deficient in the accuracy department), MOS memory, a cache and a 7-segment LED display register. Features such as the user control store and the micro-registers are not used at the moment (I'm too busy), but may be eventually. The point is that a standard 11/40 system will run without change, although I was restricted to single user because the terminal mux's were different.

After a few hours of pounding away on the console duckwriter (mostly to do with configuring for two DZ's instead of one with a DJ and reorganizing /dev) a multi-user system was booted successfully. Further pounding on the 11/40's duckwriter resulted in it being booted multi-user, with a link between the two. It was at this stage that parity errors started showing up several times a day on the 11/60, resulting in it keeling over. Since not much was known about the 11/60, it was assumed that they were memory errors, and the boards and controller were replaced. It should also be mentioned that the

initial installation by DEC was botched, resulting in total destruction of the memory boards, and thereby casting suspicion on the memory system afterwards.

After reading the not very helpful processor handbook, and being directed to the internals manual, it was realised that these parity errors were cache errors, and were in fact the 60's way of telling me that it just recovered from a cache error by going to main memory instead, and interrupted to let me know. Disabling the cache by turning on certain bits froze the CPU, whether it was done via software, /dev/kmem & ADB, or the switches. The cache was replaced and the troubles disappeared. I still can't disable the cache, but at least it doesn't freeze when I try!

I'm told by the way that this is the most common fault on the 11/60; the cache board also contains the memory management and godnoze what else. Perhaps DEC should have suspected this earlier, but they don't seem to be very impressed when their favourite operating system is not the one being used. Someone should also tell DEC that when their engineers make snotty comments about UNIX keeling over at the slightest provocation, this is generally because it's next to impossible getting useful information out of their silly handbooks. It's fine when their own software people have access to inside information, but UNIX implementations are generally done empirically.

The actual software changes were minimal. A new symbol was created ("_1160" for C, ".PDP1160" for AS) which in most cases merely paralleled the existing 11/40 symbol. The display register shows a 1-bit shifting from side to side when idle, as the standard worms, springs etc do not show up too well on an octal display. The FPU floating point unit is included, although nobody except FORTRAN users use it.

The biggest change was in trap.c, to handle parity errors. Although the 11/60 has a cache control register (just like the 11/44), and it lives at address 0177744 (just like the 11/44), DEC in Their Infinite Wisdom did not see fit to make them compatible! The idea is to test for the CPU ABORT bit, and if set indicates that either a memory error was encountered (tough), or it was a cache error and another bit was set saying that you want it aborted. Otherwise it means a recoverable cache error occurred, and this is your big chance to log it and disable the cache if it is too flakey (see above comments about cache disabling). That piece of information was gleaned from several separate paragraphs in the handbook; it would have been nice if there was a section on handling memory/cache errors, but the handbooks are about as mysterious as the processor itself.

There are other minor differences, but they relate to operations, such as booting with CTL/BOOT (we got the RP03 boot installed for the Ampex DM980), and forcing a panic with 207/LSWR/1/MAINT (plants "1" into the PC).

So there you have it; the 11/60 has only minor differences to the 11/40 - the only problem was to do with handling cache errors, and that wouldn't have shown up had the cache been reliable to start with. No doubt if/when I start delving into the micro-registers to handle instruction recovery etc more difficulties will arise, but for the moment everything works well.

Political History of UNIX

Notes on his talk at UniForum, Washington

by Andrew Tannenbaum
MASSCOMP
Westford, MA 01886

INTRO

This isn't a talk about how Ken Thompson and Dennis Ritchie hacked their beloved and renowned PDP-7 to pieces in a tower on high.

I will not mention the extraordinary Bell Labs research environment - crafty computer science gurus who took precious time off from their highly important research efforts to help beat UNIX into shape.

I'm going to talk about some of the more earthly factors that contributed to the UNIX that we know and love today. Perhaps after hearing my story of how we got here, you'll be more able to follow in their footsteps, bringing your own products similar fame and glory, or at least you'll be better prepared to understand the results of UNIX's unusual upbringing.

There is an incredible number of people riding on the UNIX bandwagon, the size of the UNIFORUM audiences is testament to that fact.

THE CABBAGE PATCH OPERATING SYSTEM

I sometimes think of UNIX as the Cabbage Patch Operating System.

UNIX wasn't a completely new idea, it was an amalgam of good ideas.

It's certainly the latest craze, with people foaming at the mouth lining up in crazed hordes to get a peek, sometimes paying ridiculous prices for an opportunity to use the product. Of course, they say, it's worth it, there are lesser pleasures in life which are far more expensive.

Like the Cabbage Patch Kids, UNIX has existed for a while, lying around in a relatively dormant state, waiting for the market to explode.

Like Cabbage Patch Kids, every UNIX is slightly different (I'm sorry to report) but they're all similar enough to be valuable as members of the group.

Cabbage Patch Kids are the product of a company which doesn't specialize in dolls: Lately Coleco (Connecticut Leather Company) has been concentrating on the Adam computer and Colecovision home video games.

Of course, UNIX is the product of another big toy company, AT&T.

WHAT TOOK SO LONG?

You might wonder what forces have tugged at UNIX during the almost 15 years since UNIX was conceived.

Until recently, UNIX was never an AT&T OS product in the sense that VMS was a DEC OS product. UNIX was always, AT&T would claim, a telecommunications support tool, only because AT&T was a restricted monopoly and it was forced by

legal consent decree to stay out of the computer business.

This important idea here is that AT&T wasn't allowed to compete in the computer marketplace, they weren't allowed to sell a product which would compete with IBM or DEC or any of your companies. This caused UNIX to grow up in unusual circumstances, with some benefits and some disadvantages.

In the case of most computer products, a computer company sees that the marketplace yearns for a certain product or service, and then it struggles to create that product or service before someone else does, so that it might make a killing before the rest of the market produces clones. This wasn't the case in the early days of UNIX, though it is certainly the case today.

When UNIX left the caring hands of Thompson and Ritchie, it was soon handed off to an entity called the UNIX Support Group (USG). The people who controlled the progress of USG controlled the future direction of UNIX through the 1970's.

HOW DID BELL USE UNIX?

Within AT&T, remember that UNIX was a tool, used by the Bell System projects which dealt with research and development of different parts of the telephone network - like switching, network planning, service order processing, and directory production.

UNIX was the prize in a tug-o-war between these various projects, UNIX was controlled by the telco projects with the most bucks, projects like BANCS and 5ESS, and the BTL UNIX releases sometimes had special purpose software in them with just these projects in mind. These were not the needs of the programmer, or any other common class of user, they were the specific needs of telco projects.

PROGRAMMER'S WORKBENCH

What was the Bell System going to do with UNIX?

Bell Laboratories had an enormous investment in expensive computer equipment: IBMs, UNIVACs, Honeywells, and such. It was decided that it would be efficient to have a coherent USER interface to all these systems. What I'm trying to say here is that all the drones who punched cards for the 360's got jealous of the UNIX users with their timesharing terminals who didn't have to wait overnight for job turnaround. Today it might strike you as strange that a Teletype Model 33 user was a subject of envy: those days are gone indeed.

Anyway, one of the major UNIX forces at Bell Labs devoted itself to producing PWB, the Programmer's workbench. Programmers would learn one editor (ed) one command interpreter (sh) and be able to submit their batch jobs over rje links from one UNIX terminal. UNIX was to have COBOL syntax checkers and dump analyzers. The idea was to leave the crunching to the big batch machines, and let UNIX front end handle the humans.

Interestingly enough, I heard Bill Joy give a speech in Massachusetts where he claimed that today's UNIX workstations should be used like yesterday's terminals, and that our workstations should be networked to large mainframe CPU's which can do the supercrunching. This philosophy is not at all unlike that of PWB.

Within the Bell System, programmers used UNIX to talk to their IBM, UNIVAC, and Honeywell mainframes.

The Bell System looked at UNIX as a small machine OS. ARPA had different ideas, UNIX would be the successor to TENEX.

UNIX had no competition. Still doesn't. While this is nice, it does tend to make people who protect and defend UNIX pretty soft.

UNIX SUPPORT GROUP

Within BTL, There was a UNIX SUPPORT GROUP, but you never heard about the UNIX DEVELOPMENT GROUP. This is because no Bell System organization had the charter to DEVELOP computer operating systems. There were computer researchers, and there were people like USG who supported telco projects. No one whose charter was development. People in support positions are never given the respect that people in development positions are given.

In the beginning of USG, the name UNIX had prestige, it was a clever research toy. USG likewise had prestige, and there were clever hackers working within USG. As UNIX within BTL became more of a capitalist tool, the hackers likewise had to become capitalist tools, there was less prestige in USG, less clever work to be done, the hackers became disenchanted, and prestige and progressive development within USG disappeared.

There was no screen oriented software or OS hacks because no one had a charter to do it. The political and philosophical powers didn't want it. Sort of like the churches of the middle ages.

BELL AND BERKELEY

The UNIX development time line looked something like this:

research	development	maintenance	panic-explosion
		Berkeley here.	

BTL didn't really have a distribution policy in the early days, you got a disk with a note:

> Here's your rk05, Love, Dennis.

If UNIX crapped on your rk05, you'd write to Dennis for another. Sort of like human fsck.

There are USG UNIX User Meetings every six months, they used to always be in a big auditorium at BTL Murray Hill. For a long time there was always a carnival atmosphere at these meetings, there was always plenty of room in the auditorium, most of the congregants had lots of friends to yak with in the audience, similar to the early days of USENIX.

One of the most amusing parts of the USG UUM's was the discussion of the number of UNIX licenses that have been distributed outside the Bell System. There was always a foil which described the Bell System UNIX support policy:

no advertising,
no support,
no bug fixes,
payment in advance.

This slide was always greeted by wild applause and laughter, and, of course, there were always twice as many licensees this year as the year before. Not much to complain about.

The Bell System was fat and happy with its position in the UNIX market. Send out some tapes, rake in some bucks, not a care in the world. The USG were the Bell System's arrogant fat cats.

Then there was Berkeley. DARPA was UCB's Daddy Warbucks. UCB was a scrappy little alley cat with some street smarts to get the job done in the pinch. Where the Bell System sat on it's haunches, UCB used some crazed grad student slave labor to whip up a product that the Bell System would have taken forever to produce. The incentive just wasn't there in the Bell System.

As long as UNIX ran on PDP/11's, UNIX was never taken seriously by people with "real work" to do. No one really believed that you could actually PORT an operating system, it just wasn't done. The PDP/11 was nice and cheap, but you couldn't run big processes on it.

EXPLOSION

When the VAX came out, you could address memory out the wazoo. Not only that, but a bunch of Bell Labbies actually PORTED UNIX to it, in *NATIVE MODE* without a gargantuan effort. The VAX had virtual memory and number crunching capability, and Berkeley had hackers interested in exploiting it.

When the VAX 11/780 emerged as a UNIX workhorse, 4.1bsd emerged as the UNIX system of choice for the VAX. Outside the Bell System, practically no one ran Bell System UNIX on their VAXes. People would buy a 32v UNIX system from WECo just for the license, they'd never even read the tape, they'd then send their license to UCB and get a copy of the latest BSD release.

Why? The UCB system was more programmer friendly. Eventually, it had paging, job control, faster I/O and higher throughput, support for many non-DEC I/O devices, reasonable though not perfect support software like screen editors and mailers, and compilers for languages like lisp and pascal. UCB users were happy because they got the toys that they wanted to play with.

BELL ATTITUDES

Eventually, smart users throughout the Bell System were running hacker-friendly 4bsd, and USG first responded with "it's not really faster, it only SEEMS faster." Well, the hackers wanted a system that seemed faster.

Why couldn't the Bell System respond? In a nutshell, BTL CS Research was too small and godly. USG was too mundane. There was nothing in between.

Yes, the Bell System does have an incredible record of innovation. They have UNIX, as well as some lesser accomplishments like the transistor and negative feedback, fundamental development of lasers and Nobel prize-winning work in radio-astronomy. They even do some telecommunications work.

Looking at it in another light, Bell Labs has more PhD's than any other organization in the world. If I remember my propaganda correctly, they have well over 7,000 PhD's, and another 20,000 engineers who can actually get the work done. While the Bell System has been a leader in innovation, they haven't done it by making efficient use of their resources, and indeed they were never under any pressure to. For this reason, a small and elite crew of hungry guerrillas like the UCB students who worked on BSD were able to have a major impact on a small piece of the Bell System's workspace.

What would have happened if there was a group of 20 hyperactive hackers developing UNIX over the years?

The BTL environment consists of mostly telephony engineers, EE's with communications backgrounds. They are ed users, 300 baud silent 700's, tek graphics scopes. Well educated, but with poor understanding of state of the art interfaces and very stubborn and arrogant. Ed and sh are just fine, thank you.

Bell was not the SAIL/ALTO/MIT/ARPA LISP/EMACS DEC10 AI environment, which is where much of the winning software from the 60's and 70's came from. Maybe not AI, but at least some good software.

Dennis Ritchie says that BTLCS was waiting for a DEC10 that never got funded. If they got it, UNIX would never have gotten off the ground. What do we owe to the Bell System administrator who turned down the DEC10?

Summary of
The European UNIX-systems User Group Meeting
Nijmegen, 1984

as seen by Red George

As Uniforum was described as a zoo (more like a circus), EUUG brought to mind a peaceful gathering in the park (perhaps reading poetry aloud). There were approximately 300 attendees at the meeting. The technical content was high and the commercial influence was low-key. The conference was held at the University of Nijmegen. The auditorium was a lecture hall in design fitted with high-tech audio/visual apparatus. There was a vendor exhibition held in the promenade outside the auditorium and in a couple modest sized rooms. Although the wares were uninteresting for the most part, there were machines to play with and give-a-ways of buttons, tee-shirts and plastic bags. The lack of a powerful exhibition in no way detracted from the meeting, it enhanced it. There were a couple of interesting terminals. The best was the M2150 from Microcolour Graphics. It has a 4096 colour palette (16 at a time) with hardware (power of two) zoom, area fill, pan and a cross-hair cursor. The graphics resolution was 640x384. There is an independent text plane (80/132 x 80). The price was around $US3000.

The meeting began the evening before the first day of talks as people formed SIGs in the local hotel bars. The European Unix community is not unlike its North American counterpart. It is populated by interesting and knowledgeable personalities (some more than others of course). Perhaps because of my perspective (but more likely because of the character of the meeting) the conference was excellent, technically rich and socially invigorating.

I listened to three quarters of the talks, attended a few more than that and missed a few entirely. I will describe some of the talks.

Emyrs Jones began the meeting by not giving the opening remarks in Dutch.

Mike Kelly from Teletype talked about the 5620. (He also gave a short sales pitch on the 3B series of computers which I found irritating.) One of its features is that it doesn't require you to come up with another version of Unix for your workstation (you just have to run a version that supports the 5620 on your host). It runs on standard system V from AT&T Technologies. It even runs on a VAX, "which is important today, but probably won't be tomorrow." A megabyte of memory makes a more usable system. He described layers, xt and demux functionally. Although he couldn't make a commitment, he wouldn't be surprised if there was a Berkeley port before long. They are looking at colour, peripherals (disks), and compatibility with other Unix systems. A questioner from the audience (from AT&T-BL) suggested a terminal ought to come with the capability to download it. Another asked why Mike was talking about peripherals. Next they'll put Unix on it. Kelly replied that they certainly won't put Unix on it, it's not a suitable architecture. Someone from England with a Canadian accent bitched about non-interaction of the layers and the lack of ability to have multiple processes in a layer. Kelly promised IPC within the 5620 by the next release and RS422 support within a year. Olivetti will be the exclusive distributors of the 3B line and terminals.

After the coffee break, Adrian Freed introduced a couple of talks on the future of Unix at Amdahl and AT&T-BL. He advocated (I think) a merger of System V and Berkeley versions and voiced a plea for the availability of source.

The speaker from Amdahl talked about their product. There was a 13 MIP model and a 20 MIP attached processor model. Lacking was full duplex ASCII support. Their next product would be an implementation of System V. It would also have paging and vfork(). He mentioned that after the third port to new versions of Unix, they used the Bell (sic) source and implemented their changes with ifdefs.

Next Larry Crume spoke of where AT&T-BL Unix was headed in the future. They will continue to provide source, they will continue to provide portability. He said humbly that AT&T-BL has learned, although it took them a while to learn, from Berkeley. He had the right perspective in that Berkeley was a research organization and their output was to be looked at carefully. Some of their work would be incorporated, some of it served as models which would be reimplemented, other things were not necessarily relevant or useful. He mentioned that job control was changed slightly from 4bsd (cough) so user programs don't have to worry about it. With respect to other features, "We will move forward to implement those in an evolutionary fashion." He was concerned over the growth of commands' sizes. Future Unix - A base, the kernel with a minimal set of drivers and utilities (libraries). There would be add-ons such as the C language and other utilities (grep, sort, awk ?). He listed the basic commands. Paging kept cropping up and was eventually addressed. It isn't available yet because AT&T-BL hasn't been satisfied with the performance of their implementation. He hopes that a satisfactory implementation will come out in third quarter of '84 (he hopes many things will be available then). It will provide a large address space and isolate the memory management architecture. There will be no application program changes and it will not hurt users who don't need paging. He also expected record and file locking a la the Unix standards committee.

Lunch was served. I believe it was some sort of chicken along with a meatball soup and fries. Fortunately Dutch Heineken does not taste like swamp water.

Bill Murphy spoke very briefly offering to give people the answers they need. (Is that like giving people the answers they want?)

After the tea break Eric Allman continued with his analysis of databases. I didn't take any notes having spent the entire time trying to get the writing shelf for my seat/desk in place.

The next talk was about an interesting application of Unix. It was to provide a database for kidney dialysis patients and to control the dialysis machines dynamically (e.g., to remove water at a variable rate over time.) Using the computer, new techniques were implemented which otherwise were impractical. The obvious fear from some of us in the audience was the inevitable "Out of blood, core dumped" message. But the machines were very sophisticated and provided many failsafes. Artificial kidneys were passed around. The speaker observed that touch screens were a good gimmick but not really useful for his application.

Andrew Hume talked about integration of various user services. For instance given a mail service and a calendar service, could one send a calendar as mail. Surprisingly often, this is not possible. He talked about his work with multiply typed objects and a bitmapped/mouse interface which overcame these deficiencies.

At some point during the talks a notice appeared on the chalkboard. "This years competition. NOT annoy Rob Pike with new switches for cat. BUT suggest an extension to C which will benefit the community??" The list of suggestions grew throughout the day and was anonymously erased the next morning.

We were on our own for dinner (thank goodness) and a few of us ate in a hotel restaurant. It was fantastic and not very expensive. I did not attend the bar SIGs that evening but dutifully retired to my room (10 minutes away in nearby Groosbeck) to finish some transparencies.

Tuesday morning began with a talk on dynamic profiling from Kirk McKusick. He covered the dos and don'ts of tuning as well.

Next was a discussion of the APSE and unix approaches to software tools. The problems of a large software project were discussed (>100 people, >50K lines, 30+ years lifetime). They were listed as:

1) divergence of intended program behaviour
2) cost estimates exceeded
3) late delivery
4) logical errors/unreliable
5) high maintenance costs (> 70%)
6) duplication of effort

Next the outcome of the software crises was described. Better languages, better methodologies, and support environments. An APSE is a database comprising an information repository for the entire life cycle. Also, communication interfaces (user, system and tool interfaces) and a toolset - integrated set of tools for entire life cycle. The same kernel APSE (KAPSE) is implemented on various operating systems to make the environment look the same. Why a database?

1) tools need not know information representation
2) information can be added without affecting other tools
3) flexible attributes and relationships can be constructed
4) transactions
5) version control
6) integration

As always, there is the conflict between flexibility and structure.

The next talk was about a C Unit Test Harness. It performed automatic test driver generation using a test spec file and did execution logging. The spec must have predicted output. The test then is easily repeatable and can be used with a test coverage monitor (profiler).

After the morning break the following session included a talk about automatic generation of syntax directed screen editors. A bottle of fine Dutch beer was introduced. The beer represented students - full of energy.

The bottle itself was the Dutch bureaucracy. The bottle was shaken and opened, the students energy was impressive. The syntax directed editors are generated from the language definitions as represented in the scanner and parser. A generic editor module is linked to the lex and yacc outputs. The editor incorporates a cursor synched parser, as the cursor is positioned, the grammar is parsed. State information appears just below the cursor. There is a window for output. There is an interactive option which caused the editor to enter a dialogue. Some applications were discussed. A cobol editor ("Cobol is a nice language because it is mostly redundant"). A 'sh' editor. It was noted that that the shell was not a language, the frustrated efforts to generate a BNF for it were discussed. Its inconsistency was described along with an interesting bug. (Try "<<`ls`" on your machine. We did on most of those in the vendor exhibit. The shell either died, with or without a core dump, or hung interminably. In one case a micro was unable to be rebooted after this experiment was performed.) The editor handled errors as best it could. If only one token was possible it was inserted. If more tokens could be used it suggested one. Otherwise it refused the token with an error message.

We were again subjected to lunch (something yellowish served over rice).

A simplistic network implementation was described. It is very much like the 'net' command which first appeared in Unix 3.0 and suffers many of the same deficiencies. Such a limited approach can however be quite useful, although perhaps not completely satisfactory.

Brian Redman spoke at length about the next version of uucp. Europeans are concerned mostly about phone costs and therefore clamor for more efficient use of the phone line and direct X.25 connections.

Sometime after tea there was a general meeting of the EUUG. A draft constitution was voted in by show of hands and an initial executive committee was approved (I abstained). EUUG has unique problems due to geographic considerations. The idea of free student attendances based on personal recommendations was discussed (scholarships of a sort). The prospect of tutorials to subsidize the technical meetings was put forth. (By the way, EUUG does not pay the lecturer in a tutorial.) And in order to bring down the fees, unbundling of the registration was discussed. For instance, don't include lunch. It's interesting to note that EUUG is somewhat like USENIX was before Boston. They are about to struggle with the same problems of dealing with vendors and keeping the meeting down to a manageable size and keeping it technically rich and informal. I trust they can learn a few things from USENIX's tribulations.

Two talks followed emphasising the anguish Europeans have about character sets. The problem is that 128 is certainly not enough, 256 can be made to do. Unfortunately many programs believe characters are only 7 bits. The first talk was technical, the second was delivered by an EEC bureaucrat to the wrong audience. Both talks had a message; if you think there is a rift between Bell and Berkeley, try getting the Europeans to agree on character set standards.

There was a EUUG sponsored dinner banquet at which several members masqueraded as scheduler states (runin, runrun, etc). There was a speech by Theo de Riddler in which he compared Unix to a woman, prostituted, abused, etc. Then it was compared to a famous deceased religious figure whom many worship. As a gesture to the past (glorious in its modesty) Version 5 manuals

were presented to Bill Murphy and Jim Kennedy of AT&T International.

At the bar afterwards there were many interesting discussions with various people. Several of us got into an extended discussion of the future of EUUG (and USENIX) relating to its size and purpose. There was discussion of uucp and ifdefing different algorithms for time and space tradeoffs. There was discussion of a source code stockroom administered by USENIX. Someone from AT&T-BL was very optimistic that all sorts of add-ons would be released soon. There was much interest in V8, the advice was to write letters. There were arguments about the proper placement of network routing algorithms. On the one hand it should be a separate command (mail `route ist`!dt). On the other it was argued that it belongs in uucp (or mail). Along similar lines there was discussion of the feasibility of a dynamic routing database. How do you deal with an anarchistic environment. Nothing will work for long. However any attempt is better than the current situation. There was no tequila and no Amaretto so we were stuck with beer and Grand Marnier. After the management kicked us out from the bar, the diehards reconvened in someone's room. The meeting lasted until after 5 am. I however dutifully retired so that I might prepare for a talk.

On Wednesday morning Joe Carfagno spoke about large systems. A very good presentation describing overall management of a two million line application. The talk covered development, customer interface, management tools, testing, etc. A lot of material covered smoothly and clearly. Interesting aspects of the project described are that it was successful, on time, reliable and efficient. Apparently a key factor was the rich Unix environment.

After coffee Andrew Hume opened his session by briefly admonishing the current practice of criticizing Berkeley merely because it's fashionable to do so. There are plenty of valid criticisms to be made without resorting to aspersions as social rhetoric.

David Tilbrook talked about the five pitfalls of interactive graphics. Boredom, Confusion, Discomfort, Frustration and Panic. He showed a film about "Newshole" (an interactive news copy layout aid he had designed) to make his points.

The next talk was a replay of the "Behind every Binary License" rhetoric from UniForum '84.

After another astounding lunch there were a series of five minute talks on various topics. Andrew Hume talked on /proc in V8. Karrenberg from Dortmund discussed a spooling system implemented in a similar way and at the same time as Berkeley. Greener from Imperial Software Technology gave some EUUG benchmark results. System V is fairly close to 4.2BSD. Speeded up comets (15%) DO EXIST. Tummers from Twente (Holland) described a simple implementation of Dijkstra's semaphores. Mayhew from Bleasdale gave a rudely long talk on implementing IBM bisync on Unix without the vpm. Riddler from Holland described an inexpensive and portable way to benchmark Unix (or most other systems). It consisted of a 68000 master computer controlling a small network of 6809's feeding terminals. Nick Nei from Glasgow gave a fast and furious talk on modeling disk requests. Apart from an unexpected peak at 50Hz, there was no real insight into what is going on.

Then after tea there was a panel discussion of the future of Unix. Each panel member gave a brief statement:

Larry Crume - Need Unix to drive complex micros and loadable device drivers. Predicted people don't want to be given software, that it has to be packaged. Noted that the good old days are gone.

Robert Ragen-Kelly - Concerned that Unix will be made to do things it wasn't intended to do. Urged that Unix be kept small.

Kirk McKusick - Looking forward to receiving his degree. Said that Berkeley will move to function as a research facility. Predicted less formal releases.

Adrian Freed - Ought to be source distributions of everything.

Mike Banahan - People at one time were interested in Unix itself. Now they're interested in what the system can do. Be careful of relying on a single piece of software.

H. J. Thomassen - Unix will go the way of any commercial success. Away from universities towards businesses. Hopes that Unix will provide good administrative packages. Growth of community is fragmented to points where a meeting is meaningless.

Discussion then was involved with who is still interested in Unix in and of itself. The hobbyist market perhaps? The state of Unix was compared to that of the automotive industry in the 40's. A good motor was available, what was needed was bodies and luxuries. Lots of discussion about pros and cons of the availability of source. Discussion of development environments. What about creating them for large teams by stepwise construction from working smaller environments? I've yet to see a really good panel discussion among intelligent level headed participants. This was no exception.

I conclude that the meeting was a grand success. It held my interest throughout and I came home with some ideas and many more contacts. It's too bad Europe is so far away, but I suspect the Europeans like it where it is.

Proposed Syntax Standard for UNIX* System Commands

Kathleen Hemenway

AT&T Bell Laboratories, Murray Hill, NJ 07974

Helene Armitage

AT&T Bell Laboratories, Piscataway, NJ 08854

ABSTRACT

A syntax standard for user and system administrator commands is described in this paper. The objectives of the standard are to improve the quality of the user interface and to simplify development and maintenance of commands. The proposed standard is based on the syntactic features prevalent in the current command set and it is similar to, although broader in scope than, guidelines described in the *UNIX* System User's Manual* and implemented in a command line parser, *getopt*.

The proposed standard has been recommended for use by AT&T Bell Laboratories: new commands should meet the standard and use an enhanced version of *getopt*, and the standard and *getopt* should be used as a basis for regularizing the existing command set. The standard is also recommended for use by original equipment manufacturers and it will be proposed to the standards committee of /usr/group.

1. INTRODUCTION

The syntax of UNIX System commands has led some to criticize the system's user interface (e.g., [NORM81]). The syntax is inconsistent: Although most commands conform fairly well to a single syntactic model (see *intro(1)* in the *UNIX System User's Manual)*, there are subtle variations among them. For example, while some commands allow more than one option to follow a single delimiter, other commands don't: *ls -l -t* may be abbreviated to *ls -lt*, but *lpstat -d -r* may not be abbreviated to *lpstat -dr*. Similarly, the significance of white space between arguments varies among commands: Some commands require white space before arguments to options, whereas others don't allow space, and in others space is optional. *Sort -o file* and *sort -ofile* are synonyms, while *cut -c list* and *cut -clist* are not. In addition to these subtle variations among commands, there are also commands that differ in major respects from the pervasive model--for example, *find* has an atypical syntax. These variations frustrate users when they are learning new commands, and the variations cause users to depend on the manual even for frequently used commands.

Inconsistencies aside, the syntax that is prevalent in the command set has been criticized. Specifically, the use of single letter options has been questioned. Some critics argue that using a single letter takes terseness too far. They argue that because a given letter typically stands for

---

* UNIX is a trademark of AT&T Bell Laboratories.

many different words across commands (e.g., the option -c stands for 'column', 'character', 'copy', etc.), it is difficult to learn and remember options. Also, the use of delimiters for options has been questioned. While a few commands use a plus sign (+) to delimit options that add features, most commands use a dash (-) to delimit all options. Since the dash is frequently conceptualized as a minus sign, this usage seems incongruent.

These issues motivated a project recently completed at AT&T Bell Laboratories. The goals of the project were to identify and evaluate shortcomings of the command syntax and to identify a syntax standard. The standard is intended to apply to new commands and to be used as a basis for retrofitting the existing command set. The standard should exert a constructive influence on the evolution of the command set by establishing a template toward which the command set will evolve. Through its implementation in a command line parser, the standard should also simplify both development and maintenance.

The proposed syntax standard and the reasoning that led to its identification are described in this paper. The standard is presented in Table 1. The processes involved in the identification of syntax rules included in the standard are described in Section 2. In Section 3 the rules are explained and the reasons for selecting the rules are described. Plans for implementing the standard and for related work are summarized in Section 4.

TABLE 1: THE PROPOSED SYNTAX STANDARD

RULE 1:	Command names must be between two and nine characters.
RULE 2:	Command names must include lower case letters and digits only.
RULE 3:	Option names must be a single character in length.
RULE 4:	All options must be delimited by "-".
RULE 5:	Options with no arguments may be grouped behind one delimiter.
RULE 6:	The first option-argument following an option must be preceded by white space.
RULE 7:	Option-arguments cannot be optional.
RULE 8:	Groups of option-arguments following an option must be separated by commas or separated by white space and quoted.
RULE 9:	All options precede operands on the command line.
RULE 10:	"--" may be used to delimit the end of the options.
RULE 11:	The order of options relative to one another should not matter.
RULE 12:	The order of operands may matter and position-related interpretations should be determined on a command-specific basis.
RULE 13:	"-" preceded and followed by white space should be used only to mean standard input.

2. *THE IDENTIFICATION PROCESS*

The proposed syntax rules were chosen on the basis of practical, technical, and user-related considerations. The match between different syntax rules and syntactic features of the existing command set was the primary practical concern: Naturally, the better the match between the new model and existing commands, the easier the transition to the new model will be. Given our commitment to upward compatibility, this was an important consideration. Technically, the command syntax should be consistent with processing accomplished by the Bourne shell, the C shell, and other popular system interface programs. Also, the syntax should be general and robust enough to facilitate argument processing across a wide variety of arguments in many commands. Finally, the syntax should present as simple a model to the user as possible, within the context of the technical and practical considerations. Although we wanted the syntax to be as simple as possible given the constraints, no attempt was made to address the special needs of naive users, because their needs can be more adequately addressed by providing an alternative, simplified interface (e.g., a menu based interface).

These concerns were addressed during a three phase study that led to the identification of the standard. A conceptual and statistical analysis of the syntactic structure of the command set was completed during the first phase. This analysis was primarily based on manual entries, and it included all the commands in Section 1 of the Release 5.0 *UNIX System User's Manual* and Section 1M of the Release 5.0 *UNIX System Administrator's Manual.* The analysis was used to determine how pervasive different syntax rules are among the commands and to assess the extent and nature of inconsistencies. It also served as a basis for defining the parts of a command. These definitions were critical to the standard since it is difficult to specify a syntax or grammar without identifying the "parts of speech". The classification of parts that was used for the standard is summarized in Figure 1.

FIGURE 1: PARTS OF A COMMAND

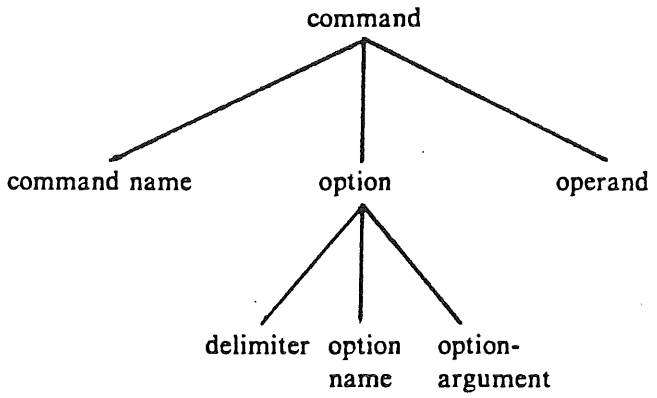


Figure 1. This figure depicts the conceptualization of parts of a command that the standard is based on. Briefly, a command is composed of a *command name* and it may have *options* and *operands*. An option is composed of a *delimiter* and an *option name* and it may have associated *option-arguments*. Operands include filenames and other parameters entered directly on the command line.

In the second phase of the study, the rules identified in the first phase were evaluated. Technical aspects of the rules were analyzed as were the human factors aspects, and the rules were compared with popular alternative rules. Syntax rules implemented at the University of Waterloo [GARD83] and rules proposed by various groups within AT&T Bell Laboratories were among the alternatives considered. Due to time and resource constraints, the human factors analyses were analytic rather than empirical--consequently, although there are many human factors arguments in this paper, experimental data are not available to substantiate them.

Finally, in the third phase of the study the strengths and weaknesses of the various rules were weighed against the impact selecting each rule would have on the current command set. A conservative criterion was used in making decisions: pervasive syntax rules were chosen over alternatives unless there was clear evidence that the benefits of making changes to the command set outweighed the costs. As it happened, in all cases where there was a single pervasive rule, it was chosen. Where the criterion of pervasiveness could not be applied, rules were selected on the basis of the technical and human factors considerations.

3. DISCUSSION OF THE PROPOSED SYNTAX RULES

Major factors that were considered in selecting the rules are summarized in this section. The impact of the structure of the existing command set on the selections is described, as are the relevant technical and human factors considerations. Strengths and weaknesses of the rules are presented and the reasons for rejecting alternative rules are summarized. Methods for compensating for the weaknesses and plans for retrofitting existing commands to meet the rules are also described.

3.1 Command Names

Rules 1 and *2* specify that command names should be between two and nine characters long and that they should include lower case letters and digits only. These constraints are consistent with existing command names (only three commands violate the rules), and they are intended to exert a conservative influence on future command names. Single character command names were excluded because they may be confused with option names and with simple editor commands. The nine character maximum length was selected to discourage the use of extremely long command names. Special characters were excluded to avoid conflicts with characters that have special meanings to the shell and upper case letters were prohibited for simplicity: As long as case does not vary in command names, users can effectively ignore it.

These constraints apply to new commands only. Due to compatibility considerations, the names of existing commands will not be changed: The few command names that violate the constraints will be left as they are.

3.2 Option Names

The use of single character option names *(Rule 3)*, is pervasive in the command set: 79 per cent of the commands that have options have single character option names only. Overall, there are roughly 1400 single character names and 130 multiple character names.

Although these data largely determined the selection of *Rule 3*, before selecting it we evaluated its strengths and weaknesses relative to the strengths and weaknesses of alternative rules. Because this rule is controversial, and because its selection influenced the selection of other rules, these analyses are described below in detail.

On the positive side, *Rule 3* has several strengths. First, it is a very simple rule. Second, it minimizes the number of keystrokes and thereby minimizes (a) time to type; (b) length of the command on the command line; and (c) the frequency of spelling errors. Third, it allows bundling of options.

On the negative side, the use of single character option names has been linked to several problems. Two problems are described below: (a) contention over the use of letters; and (b) inadequate descriptive power of single letters.

1. *Contention over the use of letters.*

   A command may have more than one option that is most appropriately represented by a given letter. Typically, this is resolved by using an upper case letter for one of the options (e.g., the *awk* command uses *-f* for 'file' and *-F* for 'field separator'). This causes difficulties for monocase terminals and it causes problems when the commands are ported to monocase operating systems. It is also undesirable because case seldom has any mnemonic significance, and the user is burdened with remembering an arbitrary assignment of case to options.

2. *Inadequate descriptive power of single letters.*

   If an option can only adequately be described by a pair of words, it is difficult to select one letter for the option name. This frequently occurs when an option is most appropriately described by an action and an object. It also occurs when an option specifies the negation or suppression of an action. This leads to choices like whether to use *-s*, *-n*, or *-m* for 'no mail' (i.e., *-s* for 'suppress mail' or 'do not send mail', and *-n and -m* for 'no mail'). Because letters have been used in different ways in different commands, the meaning of the letters varies in confusing ways across commands. For example, *pr* uses *-h* to mean 'take the next argument as the header', whereas *list*, *nm*, and *prof* use *-h* to mean 'no header'.

   The use of a single letter with opposite meanings across commands is a special case of the general problem that most letters have different meanings in different commands. Although a few letters have been successfully reserved for particular uses (e.g., *-T* and *-V* mean 'output terminal type' and 'version number', respectively) most letters represent a variety of different words across commands.

Both of these problems--contention over the use of letters and the inadequate descriptive power of single letters--led us to consider alternative methods for naming options. Two alternatives to single character option names were considered: (a) multiple character option names where the whole name always must be used; and (b) multiple character option names where the minimum number of characters necessary to uniquely identify the option may be used. Both alternatives were rejected because of problems. When multiple character option names are used without truncation, naturally, typing long names is bothersome and spelling the option names is difficult for users. Although spelling an option name does not seem difficult when you consider names like "print" and "copy", it seems much more difficult when you consider that many of the names would be abbreviations. Specifically, some option names would be abbreviations of long words and others would be abbreviations of multiple word terms and phrases.

Using multi-character option names and allowing truncation reduces to the use of single letter option names in most cases, since a single letter will typically be a unique identifier for an option. Consequently, this alternative has most of the same problems as single character option names. The one problem it does not have is the contention problem: since the user may type in as many letters as are necessary to uniquely identify an option, there is no need to have unique single letters to identify options. However, there are several problems with this solution to the contention problem:

- a new option may make a previously unambiguous option ambiguous.
- to be able to identify unambiguous truncations, the user must know the entire set of legal options for a command.
- it makes bundling impractical.

In summary, neither of the two alternatives was clearly better than the use of single letter names. Consequently, we decided to endorse the de facto standard and develop ways of compensating for its limitations. Specifically, guidelines are being developed for choosing names for options. These guidelines will apply to new options only--they will not be used as a basis for changing existing single-letter options. However, selected commands with multi-character option names will be modified to accept single letter names in addition to the existing option names.

3.3 Option Delimiters

Rule 4 prescribes the exclusive use of dash or hyphen (-) as an option delimiter. The use of - was endorsed because it is the de facto standard: 88 per cent of the commands that have options use - exclusively as a delimiter. The use of a second, alternative delimiter was not endorsed because the costs in adding a second delimiter were considered to outweigh the benefits. Briefly, the costs are:

- another special character must be reserved.
- the user must remember when one delimiter is used rather than the other.
- the user must beware of more potential syntactic ambiguities (e.g., in addition to files named *-k*, files named *+k* become problematic).

Because of the demands already placed on special characters by the shell, we did not consider reserving a new character. Since the plus sign (+) is the only alternative delimiter that is used to any degree in the command set, it is the only alternative delimiter we seriously considered.

We decided not to endorse the use of + as a delimiter for several reasons. First, + is used in four different ways in the command set, and none of those usages met our criteria for endorsement. The criteria were: (a) there should be clear, simple, and well-defined rules for determining when + should be used; (b) the rules should be applicable to more than a few atypical commands; and (c) the use of + should complement, rather than conflict with, the pervasive use of - to delimit options. The second reason + was not endorsed was: the different uses of + in the existing command set are likely to remain for historical reasons,[1] and consequently, endorsing a single use of + in new commands could be confusing. Finally, + was not endorsed because it encourages the interpretation of - as 'minus' rather than the more neutral 'hyphen' or 'dash', which is not desirable considering that many options delimited by - seem to add features.

3.4 Bundling

Rule 5 states that options without option-arguments may be bundled behind one delimiter. For example, *ls -lt* may be used in place of *ls -l -t*. Bundling is a convenience that is a benefit of single character option names. It was selected because the costs are small, and the convenience is clear. Also, bundling can be used in shell scripts to make the conceptual grouping of options apparent. For example, the options *t* and *v* can be bundled in the command *cpio -i -tv -B* to indicate that *t* and *v* are related (they both affect the table of contents), and that they are separate from the other options specified.

The rule also states that an option with an argument may not occur in a bundle. For example, the command *foo -klm file* is illegal, assuming *k*, *l*, and *m* are options and *file* is an option-argument. This is not allowed because it makes the binding between an option and its argument unclear.

3.5 Option-Arguments

Rule 6 requires white space before option-arguments. This rule was selected over two alternatives: (a) no white space before option-arguments; and (b) zero or more spaces before option-arguments. Although we don't have reliable data on the numbers of commands that follow *Rule 6* and each of the alternative rules, together they account for 160 of the 163 commands that have option-arguments. In the absence of reliable data, *Rule 6* was selected on the basis of technical and human factors considerations.

Relative to using no space before option-arguments, using a space has these advantages:

---

1. Specifically, the uses of + in *sh*, *set*, *sort*, and *tail* are not likely to be changed. *sh* and *set* use + to turn options off, while *sort* and *tail* use + to delimit numbers specifying a positive displacement from a numeric default.

1. When option-arguments are separated from option names by white space, they are easy to parse visually. For example, it is easier to pick out the components of *-f file* than it is to pick out the components of *-ffile*. Similarly, when option-arguments are preceded by white space, options with arguments may not be confused with bundled options or incorrectly interpreted as multi-character option names. (However, they may be incorrectly interpreted as options followed by operands if there are no subsequent options.)

2. Because the shell performs operations on words (as they are defined by white space), it is important that option-arguments be treated as words by the shell. This is clear in two cases. First, it is clear when the option-argument is a null string: for example, a null argument can be specified by the string *-d ''*, but not by the string *-d''*, because trailing quotes are stripped by the shell. Second, it is clear when filename expansion is applied to the option-argument: for example, filename expansion works appropriately on the string *-f x**. It does not work appropriately on the string *-fx** (*-fx** matches strings that begin with *-fx*).[2]

Clearly, these considerations rule out the alternative of prohibiting a space before option-arguments. However, they don't rule out the alternative of allowing space or no space. We decided to require a space, rather than allowing space or no space, because it is a simpler rule. Also, since the use of space and no space are not synonymous technically, treating them as synonyms is unwise. However, to ease the transition to requiring a space in existing commands (many of which presently allow or require no space), the ability to handle both will be provided indefinitely in the commands.[3] The documentation and training will encourage users to use a space and users will be warned that we are evolving toward requiring a space. Eventually, the commands will be changed (by making a change in the command parser) to require white space before option-arguments.

According to *Rule 7*, option-arguments cannot be optional. In other words, there are only two kinds of options--options that are always used with option-arguments and options that are never used with option-arguments. *Rule 7* follows directly from *Rule 6:* Given the choice to have white space precede option-arguments (and our goal of having simple, uncomplicated rules), option-arguments cannot be optional. This was not considered to be a significant disadvantage of the choice to have white space precede option-arguments, because our study of the command set revealed little need for optional option-arguments.

Rule 8 states that when an option has more than one argument, the option-arguments must either be separated by commas as in *foo -f file1,file2* or separated by white space and quoted as in *foo -f "file1 file2"*. In effect, this rule states that arguments to an option must be included in one word passed by the shell. This rule was selected because complications occur when option-arguments are passed as separate words by the shell and the number of option-arguments following an option may vary.

3.6 Order of Arguments

Rules 9, 11, and *12* reflect de facto standards: the vast majority of the commands require all options before operands and the order of options relative to one another rarely matters, while the order of operands frequently does.

As an alternative to *Rule 9* we considered allowing options and operands to be interspersed with no meaning attached to the order of options relative to operands. This allows the user to enter options anywhere within a command (presumably, the user may remember a neglected option after he or she has entered operands). We did not select the rule because in some commands it is clearly inappropriate to allow options to follow operands (e.g., commands that take other commands as

2. Incidentally, these considerations preclude the use of any character as a delimiter between option names and option-arguments (e.g., =).

3. Since *getopt* presently allows space or no space, it will support this feature without modification.

operands). Also, some commands require both options and operands in a particular, atypical order for sound technical reasons (e.g., *cc*), and those commands can more easily be viewed as exceptions to *Rule 9* than as exceptions to the alternative rule. Finally, command line editing is a better solution to the problem of "neglected options".

As an alternative to *Rule 11*, we considered a rule stating that later occurrences of an option should override earlier ones. This allows the user the flexibility to change his or her mind while entering options and to contradict him or herself. However, this feature is of limited use because there is typically no way to "erase" an option by entering another option. This feature only allows the user to: (a) enter the same option with different option-arguments, and to have the later occurring arguments take precedence over the earlier ones (which may or may not be desirable); and (b) enter mutually exclusive options that have no established precedence relation, and have the later option take precedence over the earlier one. Because the feature is not generally applicable and because command line editing is a better solution to the problem, the alternative rule was rejected in favor of *Rule 11*.

The alternative to *Rule 12*-- that is, to assigning positional interpretations to operands--is to make the operands into option-arguments, identifying by the corresponding option name the role of the option-argument. This alternative was not seriously considered.

The "--" argument, specified in *Rule 10*, may be used to keep operands that begin with - from being interpreted as options. Although operands typically do not start with -, in some cases they do. For example, it is not uncommon to *grep* for a pattern that begins with -. Also, filenames may begin with -, although this is discouraged and there are other ways around the problem (e.g., *. /-k* may be used instead of *-k)*.

3.7 Standard Input

Since many commands need a placeholder for standard input, there should be a convention that is used consistently across commands. *Rule 13* endorses the use of - as a pseudonym for standard input. A placeholder should be necessary only when input to a command is optional and when input is coming from more than one place.

4. GENERAL DISCUSSION

There is little that's new about the proposed standard. It is consistent with the syntax pervasive in the command set and it is largely consistent with guidelines developed in 1978 and implemented in *getopt* (see *intro(1), getopt(1),* and *getopt(3)* in the *UNIX System User's Manual).* After analyzing the problem, we decided that a conservative solution was the only sensible one: Given the software investment all users have in the current command set and AT&T's commitment to upward compatibility, practical impacts must be kept to a minimum. The proposed standard accomplishes that. At the same time, if the standard is successful it will have a positive influence on the evolution of the command set. By ensuring consistency among new commands, it will put an end to the proliferation of variation among commands. Also, by providing a model toward which existing commands can evolve, it will partially correct the existing variability.

The UNIX System Development Laboratory is moving ahead with the standardization effort internally and presenting the proposal for adoption in the larger UNIX System community. It is recommended that new commands conform to the standard and use an enhanced (smaller and faster) version of *getopt*. Mechanisms are being identified for enforcing the standard internally, and for handling "escapes" (cases where good judgment indicates that one or another rule should be broken). Plans for retrofitting the existing command set are also being identified and guidelines are being developed for designing new commands and for enhancing existing commands. These guidelines will address the problems of single character option names by establishing criteria for the selection of option names. A new standard is also being established for manual entries--the new standard will correct format problems with the current manual entries and it will ensure consistency across entries.

Acknowledgments

We are indebted to Jerry Vogel and Aaron Cohen: after evaluating the problem, we arrived at a solution very similar to the one they championed five years ago. We are also indebted to Mike Bianchi and Larry Wehr for bringing their considerable technical skills to bear on the issues addressed by the standard, and to Brian Keene for collecting data about command options. Finally, we are grateful to countless other people whose ideas helped shape the proposed solution.

REFERENCES

NORM81 Norman, D.A. The trouble with UNIX. *Datamation*, 1981, *27:* 12, 143-150.

GARD83 Gardner, J. Notes on command syntax. Paper posted to the net.cog-eng newsgroup. September 7, 1983.

Send Comments To:

Software Sales and Marketing
PO Box 25000
Greensboro, North Carolina 27420

nwuxd!unixsys

Installation for Electronic Address:

Add the following entries to the file "/usr/lib/uucp/L.sys"

nwuxd Any ACU 1200 13122601844 in-BREAK-in-BREAK-in unixml word bellmail
nwuxd Any ACU 300 13122601844 in-BREAK-in-BREAK-in unixml word bellmail

WHAT'S ON IN NEWCASTLE?

Albert Nymeyer
The University of Newcastle

Here at Newcastle University we've been running a PE3220 with Edition VII TWG UNIX for 12 months now. Edition VII is the usual Bell Level 7 plus Berkeley enhancements. I'm told Perkin-Elmer doesn't sell this product any more - they now have their own UNIX. We use UNIX to entertain our 600 odd first year mathematics students on a total of 18 screens.

We received the system late in 1982 (hardware in August, software in November). Like everybody else we decided to run Berkeley Pascal, but like nobody else, we went for vi instead of ed for the students.

Looking back on 1983, by far the major hassle we had was security. On many occasions the system was broken, either crashed, tied up, hung up or just plain screwed up by a small band of dedicated hackers that was formed half way through the year. Their rate of learning and their ingenuity was quite remarkable. Towards the end of the year most of the problem areas in UNIX had been fixed. The system as delivered from the United States is extremely vulnerable to a hostile student base. The second major problem was the huge amount of time a minority of students would spend at the terminals, and this was obviously detrimental to their mathematics. Surprisingly, vi wasn't that much of a problem. It did affect system performance, but not drastically. If anything, the problem with vi was its sheer power. The majority of our mathematics students are not really interested in Computer Science, on terminals they are positively timid. When they got into vi, they got into trouble.

We decided that, in 1984, we would attempt to redress this tremendous imbalance caused by a very small minority hacking day and night, and a large proportion not getting out of the system what they should, which was simply to run a series of small Pascal programmes.

Shed (SHell EDitor) was born, ironically, as a result of the very hackers of 1983 (who have now 'assumed' positions of responsibility). This beast called Shed is now the login shell of all first year students. It is ed-like, but doesn't use metacharacters, and doesn't have things like shell escapes. Apart from deleting and inserting lines, and character substitution, shed can only do a few well defined functions. Students can run a program (i.e. pix), get a directory listing, remove a file, print a file (on terminal or lineprinter) and very little else. Shed is a simple line editor, they can never leave it, apart from logging off. The prompt it uses is the number of minutes a student has left in an allocated weekly total. Each student has a time file in their account which stores this number. It is read on login, and is periodically updated when a prompt is printed on the screen. On logout it is written to the student's time file. If the student exceeds his time while logged in, he is evicted. Once he reaches his allocated time he can not login again for the rest of the week. All the student time files are initialised at the beginning of the week.

The students therefore never see Unix, or a C compiler. Later in the year, the better students who are progressing well, will be allowed to use the normal system, if they wish. The normal system is, of course, available to Computer Science classes.

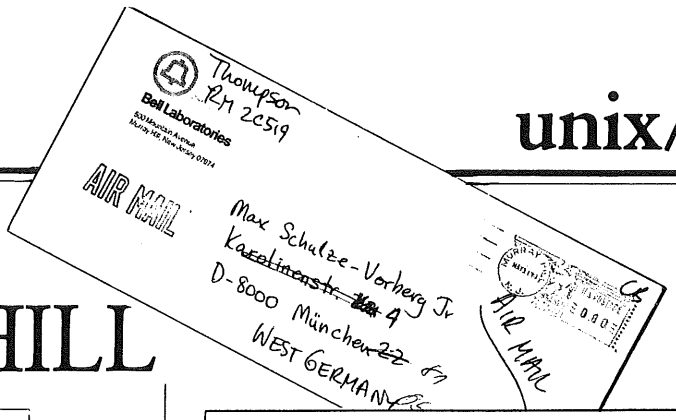
The good thing about shed is that all the students are now at the same level. They are learning Pascal better, and are more conscious of using their allotted terminal time to maximum benefit. Hopefully, by using shed, we can also overcome the problem of a very high failure rate amongst good computer-oriented students, not just in Mathematics I, but in all subjects.

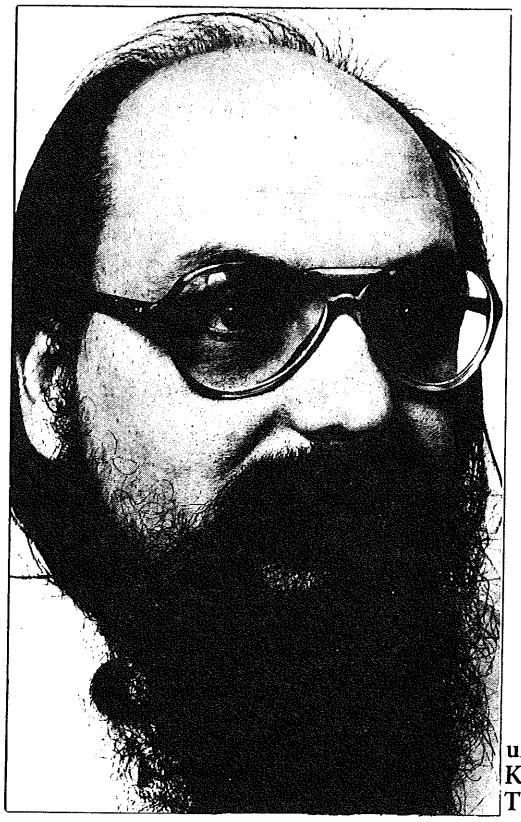
In one fell swoop we have of course, hidden away an essential feature of Computer Science, UNIX. Whether this is too high a price in the teaching of mathematics is arguable.

It is hard to say whether shed will be standing in 1985. A part or whole implementation of AUSAM is possibly our next project. We will see how 1984 unfolds.

mail from . . .

MURRAY HILL





unix-guru Ken Thompson

Dear Mr Schulz-Vorberg,

I have been putting off replying to your letter/telex because I have real trouble playing the rôle of celebrity. I do apologize for missing your deadline. I will try to answer your questions now; but under protest.

1. Can you stand the word "unix"? Sure. I regard UNIX probably the same way you do. I use it as a time-sharing system. At this point, I have no "motherly" feelings toward it. I think of it in the third-person the way any user would.

2. If you redesigned UNIX, what would you do differently? I was first asked this in the question-period after a talk on UNIX in San Diego. I answered then "I would have spelled CREAT with an E.". There are, of course, many things that I would do slightly differently. There are a few things that I would have done to set an early standard if I had had the foresight. Examples are bad-block handling and inter-process communication. Without a standard, there are many different implementations that detract from portability. I am not implying that my implementation would be as good as others, just that it would be first and as such there wouldn't be multiple competing implementations that are not significantly better.

3) What are you working on now? I am still working on computer chess. I have several projects in that field, Belle is still slowly evolving; I am working on data bases for both chess games and openings; and I am working on exhaustive solution to some endgames. In UNIX, I am working on a local area network – as most people are. I hope to have a distributed file system with banks of user machines.

The program and user interfaces would look like normal UNIX. I have ~~also~~ worked recently on subjects relating to aviation — spherical trig, route planning, weather, cartography and things like that.

4) what does success of UNIX mean to you? Its the same as success in anything. Suddenly I do not have to justify working on strange ideas. (example chess). There is a freedom that I didnt have before. Resources are more readily available. I speak with more authority even though I do not deserve it. This last point can be quite a responsibility. I must be careful in technical discussions. ~~so~~ It is ~~p~~ossible to wound colleages with ~~cr~~ criticism where only a difference of opinion was intended.

I hope that I have been responsive.

Regards,
Ken

;login:

The USENIX Association Newsletter

Volume 9 Number 1 **February 1984**

CONTENTS

The deadline for submissions for the April issue of *;login:* is April 16

USENIX Conference Proceedings Available

Washington DC UniForum Conference — Winter 1984

The proceedings of the January UniForum Conference are being produced by /usr/group and will be available in June or July. The price is $30 per copy, plus $15 per copy for overseas postage. They may be ordered from:

/usr/group
4655 Old Ironsides Drive
Santa Clara, CA 95050

Toronto Conference — Summer 1983

Copies of the proceedings of the Toronto Conference are available for $30 per copy, plus $15 per copy for overseas postage. Send your check or money order to:

USENIX Association
P.O. Box 7
El Cerrito, CA 94530

San Diego UNICOM Conference — Winter 1983

Copies of the proceedings of the San Diego UNICOM Conference are available for $25 per copy, plus $10 per copy for overseas postage. Send your check or money order to:

Software Tools Users Group
1259 El Camino Real, #242
Menlo Park, CA 94025

/usr/group Standards Committee Meeting

The /usr/group Standards Committee met on Tuesday, January 17 at the UniForum Conference to review the latest version of the Standards document, which had been edited to incorporate changes agreed upon at the previous committee meeting held at Comdex on Tuesday, November 29 in Las Vegas. No major changes were made at the last meeting. After much discussion, a motion was made to submit the Standards document to the general /usr/group membership for a vote on the adoption of the Standard as it now exists. This was passed by a vote of 24 to 3. The Standards document had previously been submitted to the /usr/group membership for comments.

After a final round of editing and phototypesetting, the Standards document, along with the Reviewer's Guide, will be sent to the /usr/group membership for a vote. This is expected to happen towards the end of February. A two-thirds vote is required for adoption of the Proposed Standard. The documentation will include a comparison with System III and System V, as well as a detailed description of the file/record locking primitive so that members can vote with access to the proper background information.

In further discussions held on January 17, extensions in the areas of terminal handling, networking, real time, interprocess communication and object file formats were considered. It was felt that something had to be done about the *ioctl* system call so that the terminal control capabilities could be specified. After some debate, it was agreed that the System V version of *ioctl* would be the best starting point. A proposal will be generated for consideration at the next /usr/group Standards Committee meeting, to be held in April in conjunction with the Winter Comdex meeting in Los Angeles.

The /usr/group Standards Steering Committee feels that the *proposed* Standard has now received adequate review and is ready for a public vote. The adoption of this Standard will benefit both suppliers and users of UNIX-based and UNIX-like systems. Copies of the Standards document are available for $50 from

/usr/group
4655 Old Ironsides Drive
Santa Clara, CA 95050
408-986-8840

If you would like to participate in the /usr/group Standards effort, or would like to make a proposal, or even review an existing proposal, please write to the Chair of the Standards Committee:

Heinz Lycklama
Interactive Systems Corp.
1212 Seventh Street
Santa Monica, CA 90401

Local UNIX User's Groups

The USENIX Association will publish information on local user groups in *;login:*. Information on local groups giving the name, address (phone number and/or net address is desired), time and location of meetings, special events, etc. is welcome. Please call the USENIX office if you wish to have your group added to the list. Our current list of local groups follows.

The Front Range group meets about every two months at different UNIX sites for informal discussions.

Front Range Users Group
N.B.I., Inc.
P.O. Box 9001
Boulder, CO 80301
Attn. Wally Wedel
(303) 938-2923

Dallas / Fort Worth UNIX User's Group
Advanced Computer Seminars
2915 L.B.J. Freeway, Suite 161
Dallas, TX 75234
Attn. Irv Wardlow
(214) 484-UNIX

There is an informal group that meets in the Washington, D.C., area every two months or so. The current contact for that group is:

Neil Groundwater
Analytic Disciplines, Inc.
Suite 300
8320 Old Courthouse Road
Vienna, VA 22180
(703) 893-6140
npg@lbl-csam

Unigroup is a non-profit organization in the New York City area for users and vendors of products and services for UNIX systems. It plans to publish a quarterly newsletter, a local *uucp* directory, and a directory of local companies offering UNIX system products and services.

Unigroup of New York
G.P.O. Box 1931
New York, NY 10116

Reports on the Presentations at UniForum

These reports were prepared by Karen Summers-Horton with the assistance of Jerry Deroo and Berry Kercheval. Reporter and editoral comments are enclosed in square brackets.

Keynote Address

Reporter: *Karen Summers-Horton*
2843 Valcour Court
Reynoldsburg, OH 43068

Keynote Address

Jack Scanlon
AT&T

Mr. Scanlon spoke on the history of UNIX and the present status of the UNIX revolution. The early conferences in the mid-70's led to the acceptance of UNIX by the university communities. The first commercialization came in the mid-70's by AT&T, and in 1975, System III standardized UNIX.

Today, 70,000 computers run UNIX, with over 300 applications packages available. Mr. Scanlon went on to outline some pressing market needs that exist today. UNIX is moving to versatility, with portability to several different hardware types.

In the short term, there is a need to increase the switch to System V by independent vendors, both high and low end. Mr. Scanlon restated AT&T's commitment to support of System V.

Joint Session

Reporter: *Karen Summers-Horton*

The presentation of the /usr/group corporate sponsor awards went to AT&T, Plexus Computers, Gould, Inc., Zilog, Pyramid Technology, Onyx Systems, Human Computing Resources, Wollongong Group, and Santa Cruz Operation. There were also seven additional new corporate sponsors.

It was announced that the /usr/group Standards Committee had made its final acceptance of the proposed standard for UNIX and UNIX-like systems and that the proposed standard was now being submitted to the general /usr/group membership for a final acceptance vote.

Lou Katz of the USENIX Association announced that the recently voted on bylaw revisions has been accepted. The revised bylaws eliminate ambiguities and restructure the membership classification so all but student members have the right to vote. The next USENIX conference will be in Salt Lake City, Utah, June 12—15, 1984. The sites for future conferences will be: Dallas, Texas, in January, 1985; Portland, Oregon, on June 10—14, 1985; and Atlanta, Georgia, on January 10—13, 1986.

The president of the European UNIX Users Group (EUUG), Emrys Jones spoke on EUNET. One of the services offered by EUUG is a twice annual conference (the next to be April 16—18, 1984 in Nijmegen, Holland). EUNET currently consists of 121 sites. The EUUG structure is such that all of the groups elect a board, of which one member serves on a national board. This board, in turn, elects five members to an Executive Board. The national board has control and jurisdiction over EUUG via the main board member, but EUUG has virtually no control over the national group.

Next, Tom Crowley of AT&T Technologies spoke on AT&T's new products. He restated AT&T's commitment to the support of UNIX System V. Several new products include:

- UNIX, System V, Release 2 (which will be upward compatible).
- UNIX Documenter's Workbench Software (complete text processing capabilities; it is included in Release 2 upgrade for present licensees).
- BASIC Interpreter (compatible with IBM PC Basic).
- MC 68000 Software Generation System (current compiler technology).

Mr. Crowley also spoke of AT&T's new business policies. This includes Level One support for source licensees. This is three months support at no charge with UNIX System V, Release 2, and is available for many processors. Another announced new policy is royalty-free use of run-time libraries for applications developers. There is also more flexible royalty schedules for System V resellers.

Networking

January 18, 1984, 1:30 p.m.
Chair: Thomas Ferrin, University of California, San Francisco

Reporter: *Jerry Deroo*
4075 Pheasant Run
Mississauga, Ontario
Canada L5L 2C2

Driver-Based Protocol Implementations

Greg Chesson
Silicon Graphics
630 Clyde Court
Mountain View, CA 94043

Greg's contention is that UNIX has, until now, stood in the way of network implementations. Networks are now supported by hardware and evolving software. So why not simulate "future" hardware by hiding a layer of software in a classic device driver? This would minimize the complexity of the operating system interface.

The goals of the project were stated as:

- implement a LAN which requires a device driver to implement the "classic" network
- have a simple path for distributed files
- allow bulk transfer
- implement DMA.

They have implemented the socket, or character device side. They have not quite got the block device side running.

They implemented this with about 1200 lines of code in the driver, 1000 lines to implement data transport, and a further 600 lines of code to "glue it together." The system:

- has good character I/O rates
- has no problem with typed lines
- can support block I/O.

They have implemented this on all versions of UNIX from V7 up. To access a virtual terminal, all you do is point *gtty* at a special file.

They have found that not all network protocols work well with this approach. Very complex ones would tend to over-load the system, as much of the protocol code is executed at interrupt time.

They found that UNIX helped, more than hindered, their development efforts.

They would like to suggest that while networks have come a long way, that "technical responsibility is NOT the market driver." They suggest that the "gotcha principle" is still there, whereby once you have chosen a technology, you are pretty well locked into it.

They conclude that

- IPC is not required for a network
- files, terminals and other network objects can be accessed through a device driver
- hardware developments will permit further software simplification as functions can be moved from driver to board level
- standards are still incomplete

The Excelan TCP/IP Protocol Package

Bruce Borden
Silicon Graphics
630 Clyde Court
Mountain View, CA 94043

Bill Northlich
Northlich Computer System & Software
21 Newell Court
Walnut Creek, CA 94595

This talk described the implementation of the DOD standard protocol IP/TCP on an intelligent Multibus Ethernet front-end controller. This move, which requires about 35 Kb on the board, has resulted in a significant off-loading of the host. It simplifies the host/network interface, and improves the portability of the protocol.

They chose TCP/IP for the following reasons:

- it is fully specified
- it is an accepted standard
- it is widely implemented
- it has Berkeley support

They have observed a transfer rate of 12 Kb per second [?] in process to process transfers, and 12 Kb/s in file transfer. The implementation results in the fact that a file transfer over the network takes less system overhead than a local file copy.

There were some complexities introduced into the effort due to hardware peculiarities. They first had to simulate the "inside" of UNIX, to allow them to put the 4.1cBSD version into the board.

Current work is covering these areas:

- profiling the board for tuning
- allowing more than one board on a host
- internetworking
- modifications to go to 4.2BSD
- statistics gathering
- place telnet/rlogin into the board

They are currently able to interface with V7, System III and System V.

Worknet: A Xenix-Based Merged Filesystem Network

Alan Greenspan
Altos Computer Systems
2641 Orchard Park Way
San Jose, CA 95134

This talk covered the high-level design and implementation of their network. It uses the RS-422 and Ethernet transport levels, and can support up to 30 terminals with a maximum distance of 500 feet [not sure if the distance is total or node-to-node].

The implementation places one more tree level into the UNIX file structure, a "super-root" directory, from which machines on the network are accessed as "directory" names. This results in a network model consistent with the model of a single UNIX machine's filesystem.

With the implementation, it is possible to do the following:

- access a remote raw device
- move the current working directory to any other machine
- remote execution of an command. This results in non-network qualified pathnames to be relative to the executing machine.
- with symbolic links, mail, *lpd* and other services can be centralized.
- pipes can span across the network.

CSNET Grows Up

Michael T. O'Brien
The Rand Corporation
1700 Main Street
Santa Monica, CA 90406

Daniel B. Long
Bolt, Beranek and Newman, Inc.
10 Moulton St.
Cambridge, MA 02238

This talk was a "state of the product" discussion. It was the contention of the authors that CSNET was a means by which cost-effective access to Arpanet could be provided. The development group is maturing, and expect to move to a production environment in three years. Currently, CSNET has automated 80% of the site management task.

During development of the system, and in trying to widen its user base, the authors have run into several user problems:

- confusing, and varied, path conventions
- how to find a specific person
- message headers that are not too understandable
- user expectations of the system.

They want something which offers

- the cost of the post office
- the ease of use of the telephone
- and the speed of the telegraph.

It is difficult to meet all of these criteria.

Distributed UNIX

January 18, 1984, 3:30 p.m.
Chair: Alan Nemeth, Prime Computer, Inc.

Reporter: *Karen Summers-Horton*

Software Administration Over Computer Networks — The exptools Experience

Joseph L. Steffan
AT&T Bell Laboratories
Naperville, IL 60566

The experimental tools package (exptools) at AT&T Bell Laboratories in Naperville, Illinois is administered and distributed via computer networks. The exptools package includes over sixty tools, maintained by about twenty people on more than one hundred machines of different types connected by two different networks. There is one coordinator, who controls the division of the workload and the installation of the tools on the various machines.

There are many obvious benefits to this system. It reduces the computer centers work, and enables the tools to be updated fast (a matter of days instead of weeks). Also, one person can administer many machines, because it does not significantly increase his workload.

There have been some operational problems, including the network being down to one or more machines, network jobs disappearing, files being truncated on delivery, and the target file systems being out of space.

There is also the question of security. The system is probably secure enough for the corporate environment, but there is concern about whether or not it is secure enough for the college environment.

In conclusion, it is believed that this system points the way to administration of hundreds of desk-top computers connected with a network, and removes a stumbling block to their acceptance.

A Distributed File System for UNIX

Matthew S. Hecht, John R. Levine, Justin C. Walker
INTERACTIVE Systems Corporation
8689 Grovemont Circle
Gaithersburg, MD 20877

The standard topic for this session seems to have been remote filesystems for UNIX. This was one of the more scholastic of the papers presented. They described a way to share filesystems among homogeneous UNIX systems.

Similar work cited included S-UNIX/F-UNIX at Bell Labs, and a Plexus design by Groff. The LOCUS project at UCLA is more ambitious, because it deals with replicated files and transactions. The COCANET project at Berkeley is also more ambitious since it handles remote processes.

The solution is based on a "remote mount" primitive, that can mount a remote disk onto a local file tree. The work is split across the two machines at the inode level — the client does a remote procedure call of the *nami* function, which is carried out on the server.

Problems that are not solved include dealing with "..", loops in *rmount* structures, and security. No provision is made for sessions or restart detection.

No hard performance data is available. It was expected that access to remote files would be about one half as fast as local files.

Multiprocessor Debugging on a Shared Memory System

Chet Britten, Paul Chen
Metheus Corporation
5289 NE Elam Young Parkway
Hillsboro, OR 97123

This talk described the debugger used to help implement the Metheus Lambda 750 VLSI design workstation. The workstation uses two processors, and a special purpose debugger was needed. The workstation is based on 4.1BSD with Ethernet, virtual memory, color graphics, multiple windows, multiple processors, and an OEM Omega display controller.

The system was ready for beta test in 12 months. The software was done by three software engineers. The debugger more than paid for itself in development time it saved.

Some of the problems encountered were shared vectors, processor identification, different traps on different processors, changing the trace vector, the ID register, unused bits, and stack range.

Some improvements made include an option to continue tracing; indirect addressing capability; counting the number of breaks or traces; knowing about structures; having only the correct processor stop; knowing about the memory map for the user program, which is harder with virtual memory; breaking on a virtual address, which has to know a lot about UNIX; providing a logic analyzer type trigger, to trace instructions on a page fault; and nested triggers.

Having UNIX has proved to be an advantage. It provides job control, shell escapes, and the *nlist* utility.

Panel Discussion: Multiprocessing Issues

The panel consisted of:

Joseph L. Steffan, AT&T Bell Laboratories
Chet Britten, Metheus Corporation
Paul Chen, Metheus Corporation
Monty Picard, Plexus Corporation
Greg Chesson, Silicon Graphics

Issues discussed included lightly coupled versus tightly coupled systems, and the advantages to different approaches tried.

Compilers and Languages

January 19, 1984, 8:30 a.m.
Chair: Louis Salkind, New York University

Reporter: *Jerry Deroo*

The Evolution of the Portable C Compiler

S. C. Johnson, L. Rosler
AT&T Bell Laboratories
Murray Hill, NJ

The original *pcc* was built in the mid 70's. It has been moved to over 30 different machines. In so doing, it has become an effective C standard. The compiler itself has become reliable as much of the code has been re-used in most of the ports.

However, the original

- was slow
- assumed a limited target machine architecture; for example, no provisions for three address instructions, or more complicated address modes than the old reliable, PDP-11
- the porting process was complex. It involved hand-writing many non-trivial C functions to do the code generation. The process was never designed, it just evolved.

Over the last few years, effort has been made to improve the package. The porting process has been better designed. Though not yet a cookbook, it is table driven. The porting process is based on recent theory. The machine description is in a table. The code generation portion is now mostly automated.

They found that the theory does not handle some hardware features very well. Examples cited were:

- register pairs
- condition codes
- multiple register types
- "strange" op-codes

They have made some policy decisions regarding how to implement various features. If there are only two or three choices for how do do something, the choice is controlled by #ifdef's in the compiler. If there is only one way do do something, that way is provided. A user can, of course, roll their own way.

Now that the code generation part of the process of generating a new compiler is simplified [!], there are still some hard-to-implement areas. Examples of these are:

- stack frame layout
- switch code generation, struct copy
- the interface to the assembler
- width and alignment of data-types
- register variables, and register allocation

Developing and testing a new compiler is now a matter of:

- picking a similar target machine, and modifying it. This is possible now that a few machines have been done. This usually takes less than one week.
- the development of the assembler and the loader now dominate the time allocation.

They have, so far, generated compilers for the 68000, and for the VAX. One nice thing that is now possible to do as it is easier to change things, is to build a version of the compiler that will generate statistics about the program it is compiling.

They are looking to use this new version for leverage in the continued struggle to improve the compiler. It has provided a foundation onto which they can add more tools for compiler development. As the compiler is less monolithic, they can clean up ancient code.

Future development will include changes to the language's already renowned error recovery capabilities, the implementation of classes, and the construction of the second generation version of *lint*. Furthermore, they see evolution in the program development environment, including things like a super-*make*, a syntax-directed editor, and support for incremental compiling and debugging.

How to Feel Better About Knowing It is Written in C

Alan R. Feuer
Catalytix Corporation
Cambridge, Massachusetts

The speaker contends that critical programs that are written in C are perceived as dangerous. While UNIX is great for a production software development environment, other environments are better at "some things."

Contention: it should be possible to develop software in C with the safety of Pascal without sacrificing the flexibility of C. The speaker then gave several code examples of problems which the compiler believes, but execute incorrectly. In fact, some of the examples got by *lint*. *lint* can help, but is not enough.

The speaker has implemented a compiler which can place in line run-time code into the program. This would be used mainly during the development of a program, and be turned off once it was ready for production.

The speaker feels that C embodies some old assumptions which now need re-examination. [He did not have time to re-iterate those.]

An Implementation of the B Programming Language

Lambert Meertens, Steven Pemberton
Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam

"The name of the language is called B." Having said that, the speaker stated the design objectives of the language. They were:

- simplicity
- suitable for conversational use
- a tool for structured programming

They have designed the language by iteration, and the language has evolved to one for personal computing. The language is easy to learn, as it has few constructs. It is easy to use, as it is powerful. These "pluses" for the user have made the language not easy to implement for the designer. [Then again, is that not what it is all about.]

The language is command driven. New commands can be added, which allows the development of subroutines and functions. The language is strongly typed, yet variables need not be declared. Their type is inferred from the context in which the variable is used. String operators are built into the language.

The first version was implemented, in C, with about two person months of effort. It was not efficient, but was usable. Linear lists were used for storage. The next version entailed about four person years of work. Attention was paid to efficiency. The implementation was more modularized. B trees were used for the storage management.

The language is interpreted. It embodies an editor which does the syntax checking as the code is being entered. The editor supports command completion.

The product runs under UNIX. It has run on the VAX, an 11/45, a PERQ and a 68000-based machine. The product is available for a nominal cost.

Future work will address

- a fully integrated environment

- more efficiency
- static semantic checks
- graphics support

Object-oriented Programming in C Language

Brad J. Cox
Productivity Products Int.
37 High Rock Road
Sandy Hook, CT 06482

The speaker described the reasons for chosing to implement some extensions to the C language from Smalltalk-80.

With these extensions, it is now possible for one person to develop, in the same time, what using traditional methods would require several people. This is possible because:

- the extensions enhance the reusability of the code that is written
- the extensions make handling abstractions easier, so the "user" is more efficient
- the code is more changeable.

The speaker described the implementation of a tool, a "Screen Application Generator," which was built to generate a model application, an expense account information collection program. The screen application generator, SAG, is given a description of the application and generates C code. This code is combined with a large library of support functions, which are re-usable higher-level functions — an applications library.

This tool allows an application to be described as a hierarchical group of processes, in which lower related processes inherit the context of their "parent."

The speaker stated that this tool results in the reduction of the amount of code which must be generated.

UNIX Directions

January 19, 1984, 10:30 a.m.
Chair: B. E. Redman, Central Services Organization

Reporter: *Karen Summers-Horton*

Behind Every Binary License is the UNIX Heritage

B. E. Redman
Central Services Organization

P. E. Parseghian
AT&T Bell Laboratories

The philosophy of UNIX was discussed: how did it come about, and why did it grow to the state it's at now? The development of UNIX took place over a period of time, and its purpose was to create an environment that would be convenient to program development. Today, there is a new concept of UNIX: the academic UNIX of the 70's has faded away and commercial UNIX is here ("The Cheese").

Money has become very important, and competition is marking the development of UNIX today. The authors are troubled by the "unbundling" of UNIX — each part becoming an option. As more different types of UNIX appear, their value diminishes. Portability has led to incompatibility.

In order to preserve the essence of UNIX, there needs to be two separate UNIX's — the academic UNIX and the commercial UNIX. The authors believe the academic UNIX can be preserved, and the commercial UNIX is not appropriate for their needs. There needs to be a reconsideration of the development of UNIX, and the development of another UNIX, modeled after academic UNIX.

The high visibility and enormous growth of UNIX has taken place largely due to the fact that UNIX is less rigorous, less uniform and less antiseptic. Stability, clarity and uniformity have emerged, and in the authors opinion "paradise is boring."

How Did UNIX Get Here? A Sketchy History of the UNIX Development

Andrew Tannenbaum
MASSCOMP

We can view UNIX as the "Cabbage Patch" operating system. It is the latest craze, but how did it get here? UNIX was always a telecommunications operating tool — AT&T was not allowed to compete so UNIX grew up in unusual circumstances. It was lying in a dormant state, waiting for the market to explode.

The future of UNIX was controlled by USG (UNIX Support Group) inside AT&T, and they controlled the future directions of UNIX through the 1970's. UNIX was originally viewed as an interface to the big machines that would do the heavy number-crunching. Later, the University of California, Berkeley and ARPA would have different ideas. It is important to note that USG was a SUPPORT group, and there was no UNIX development group.

When the VAX 11/780 came about, Berkeley emerged as the system of choice. U.C.B. UNIX was considered more programmer friendly. USG's attitude was that Berkeley UNIX was "not really faster, it just seems faster."

In conclusion, the author believes that Bell never really made efficient use of their resources in relation to the development of UNIX.

A Proposed Syntax Standard for UNIX System Commands

Kathleen Hemenway, Helene Armitage
AT&T Bell Laboratories
Murray Hill, NJ

Past attempts to standardize the syntax of UNIX system commands have not been successful. The authors believe that times have changed, and there are several reasons for acceptance of this standard: there is greater recognition of the importance of user involvement; applications and tools are being developed in more different places; standardization has become a part of the UNIX system world; and there is more concern about the evolution of utilities.

There was a practical concern to be consistent between the proposed syntax and the existing syntax. There were also several technical concerns including compatibility with alternative shells, generality and robustness, and the idea that the syntax should be as simple as possible.

The method used to develop the standard involved three phases. Phase one was a conceptual and statistical analysis of the syntactic structure of the command set. Phase two involved an evaluation of technical and human factors aspects of alternative syntax rules. In phase three the strengths and weaknesses of alternative rules were weighed against the impact that selecting each rule would have on the command set.

A total of thirteen syntax rules were developed:

Rule 1: Command names must be between two and nine characters.

Rule 2: Command names must include lower case letters and digits only.

Rule 3: Option names must be a single character in length.

Rule 4: All options must be delimited by "-".

Rule 5: Options with no arguments may be grouped behind one delimiter.

Rule 6: The first option-argument following an option must be preceded by white space.

Rule 7: Option arguments cannot be optional.

Rule 8: Groups of option-arguments following an option must be separated by commas or separated by white space and quoted.

Rule 9: All options precede operands on the command line.

Rule 10: "--" may be used to delimit the end of the options.

Rule 11: The order of options relative to one another should not matter.

Rule 12: The order of operands may matter and position- related interpretations should be determined on a command-specific basis.

Rule 13: "-" preceded and followed by white space should be used only to mean standard input.

It is hoped that these standards will preserve the time-honored features of the UNIX system and actively guide the evolution of the utilities. It will also, inevitably lead to more standards and guidelines.

Panel Discussion: UNIX Directions

The panel consisted of:

Brian Redman, CSO
Rob Pike, AT&T Bell Laboratories
Mike O'Dell, Group L
Armando Stettner, Digital Equipment Corporation
Andy Tannenbaum, MASSCOMP
Ed Gould, mt Xinu
Mike Karels, University of California, Berkeley

Applications

January 19, 1984, 1:30 p.m.
Chair: Reidar Bornholdt, Columbia University

Reporter: *Jerry Deroo*

MLE (Multi-Language Editor)

Scott Bradner
Harvard University

MLE is an editor for natural languages. It was designed and implemented as a support tool for the humanities.

Some of the difficulties which have to be addressed when designing and implementing such a product are:

- difficulty of representing a language's character set on terminals
- designing an internal representation
- some languages have more than 100 characters; northern European languages have upwards of 200 characters

- accent representation is a problem; even more difficult is placing two accents on the same character

To facilitate the user, several features have been implemented. The keyboard is completely mapable, so that an user can configure the keyboard as they desire. The direction of cursor motion is modifiable, to suit the situation.

Each character is represented in a long integer, which contains all the information required to draw the character. Manuscripts which are imported into the system's database are now translated into the system's representation each time they are accessed. They switched to this method after finding that one-time processing made it difficult to handle the text. If they ran into an inconsistency in the translation, it might go undetected for some time.

The system supports typesetting of the languages. *ditroff* has a separate hyphenation algorithm for each language. *sort* knows about the sequence of characters in each language.

They are working on spelling checkers for each language. They feel that the package is not yet ready for distribution. They expect to have the full implementation of the editor done in June. They are doing some work on doing the fonts on a bit-mapped graphics terminal.

MPS: A UNIX-Based Microcomputer Message Switching System

T. Scott Pyne, Joseph S. D. Yao
Hadron, Inc.
6th Floor
1945 Gallows Road
Vienna, VA 22180

The speakers described their implementation of a message switching system used for communication between a base unit and mobile vehicles. Their mandate was to replace an existing hardware configuration, while retaining the existing mobile units. Their system had to "drop-in."

Due to performance considerations, and timing problems, they did not want to use pipes for inter-process communication. They did not, however, have access to a system which supported ports. They wanted shared memory, which was also not supported. Given that all they had was a binary Xenix, what to do?

They implemented shared memory by patching the kernel to put in their modifications. They implemented ports without having to make any kernel modifications. They used the Rand named-pipe approach, but had to modify it as they could not make any kernel modifications for this. The constraint is that the reader process must keep up to the writer, as there is no write blocking. They are using fixed-size messages.

Their modifications will work on systems from Version 6 up. Currently, their code is being handled by lawyers to determine distribution requirements.

Taming the Beast (An RSX Emulator for UNIX)

Daniel R. Strick
University of Pittsburgh

This package allows a binary built on an RSX system to be executed on a UNIX system.

The speaker was faced with a dilemma: he had a commercial RSX product, and obtained a UNIX system. What should be done with the commercial product?

He needed to be able to construct RSX binaries on UNIX, so he needed an interface which supported the RSX system interface.

The speaker contends that "an emulator for system X should make data translations implemented by the system X binary available on UNIX." This requires that:

- UNIX files can be accessed as UNIX files and as RSX files
- RSX files can be accessed as both UNIX and RSX files
- the emulator process must be runnable in a pipeline

Some of the obstacles which had to be overcome:

- file name mapping: RSX is a flat file system
- file attributes: RSX has a lot of them
- file types: RSX has these; this requires re-formatting from one type of file storage to another on the fly, depending on how the file is to be accessed.
- there is some similarity between RSX and UNIX in the file system representation, the difference being that RSX speaks in terms of a file number, while UNIX speaks in terms of a file name.
- RSX can not handle more than one file name (pathnames), which results in the path */usr/dan/junk* being treated as "file junk on device /usr/dan."

A UNIX Based Color Graphics Workstation

Rex McDowell
Metheus Corporation
Hillsboro, OR

The speaker described the design and implementation of a graphics workstation. The machine is built around three 68000 CPUs and a bit slice display processor. The machine runs 4.1BSD.

Each window on the screen is treated as a separate tty by UNIX. *termcap* has been modified to allow the colors of the window to be accessed. This required changes to login, and some low level tty driver changes. The activation of a new window asserts carrier on for the tty.

Window overlay is done by twiddling the color maps. Pop-up menus are stored in off-screen display memory, and copied in when they are needed.

They have implemented a core graphics library, which is a subset of the CORE standard. The system supports the notion of a "graphics entity," which is a display item. The display processor can do scaling and rotation on these entities.

User-Level Window Managers for UNIX

Or: Text Windows Without Kernel Modifications

Robert J. K. Jacob
Naval Research Laboratory
Washington, DC 20375

The speaker has implemented a window management facility that does not require any modifications to the UNIX kernel. It uses facilities that have been supplied by the kernel, from as far back as Version 6, to implement multiple windows on a screen.

The user can move from one window to another at will. Each window has a shell running in it, with a *termcap* entry which describes its size.

The package is implemented as a series of processes connected with pipes. One process listens to the keyboard and directs the input to the appropriate shell controller process, while also sending the input to the display management process, which does the echoing. Each shell control process feeds its shell, and returns characters from the shell to the display manager. The entire package is based on the assumption that everything could be represented as a character stream, and it did not matter where the stream came from or went to. (Things like *vi* fall down here.) Each window emulates a glass terminal.

Versions of the package that run on more recent versions of UNIX can take advantage of some of the new features.

The software has been posted to the net.

Implementations I

January 19, 1984, 3:30 p.m.
Chair: Michael O'Dell, Group L

Reporter: *Karen Summers-Horton*

The UNIX Paging System

Keith A. Kelleman
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974

Work is currently underway at AT&T Bell Laboratories to develop a demand paged kernel for UNIX System V. It is hoped that it will utilize the best features of both 32V and BSD.

Some of the requirements for the System V paging is that it will require no program changes — it will not change system calls. A priority is to not hurt those who don't need paging — there should be no performance loss. Also, as part of the development, there will be a definition of memory management architecture — the architecture must be general enough to support many different memory management units, paging and swapping kernels. A main component of the architecture is the "region." A region is a data structure within a system that represents a potential virtual address space.

The reason for not merely supporting Berkeley paging is because AT&T feels it is too complex and too specific to the VAX (not portable enough). They also do not like the *vfork* call, and claim there is no well-defined architecture.

The status of the project is that there is currently a prototype implemented on the 3B20 computer. The performance is good relative to System V, Release 2. They are currently porting it to a VAX.

Providing a Job Control Facility in UNIX System V

W. P. Weber, Jr., L. S. Weisbrot
AT&T Bell Laboratories
Murray Hill, NJ 07974

The reasons for wanting a job control facility are numerous. They include the limitation of a single terminal, and the sequential nature of the single terminal. BSD solved some of the problems of job control. The authors want the ability to layer the terminal and have a controller that switched them between the layers.

The implementation took place in two parts: the kernel and user pieces. The kernel piece consists of a driver, *sxt*, that provides mapping between multiple virtual terminals and single real terminal. The user piece consists of a user-level utility, *shl*, that provides control through layer control and manipulation.

This model supports the definition of job control, providing multiple environments. The authors see it as "the poor persons Blit." Some distinct advantages are that no changes are required to existing programs to run under the facility, the impact on the kernel is small, and no changes are required to the shell.

Reloadable UNIX Device Drivers

Thomas A. Alborough
Creare R&D Inc.
Box 71
Hanover, NH 03755

This talk was different from the announced talk ("Loadable Virtual Disk Device Driver and Server") because there was insufficient time for a good presentation of the original topic. According to the abstract, the original talk dealt with a local device driver that provides transparent access to a remote device.

They needed to develop a device driver in the UNIX environment. The objective was to reduce development time and provide a better product. The technique used was to overwrite the old driver code and data with new.

Elements of the technique: the driver code and data are of fixed size. The driver code and data are locatable (via *nlist*). The driver vector entry points are vectored, making it easy to change the vectors at runtime. The linker handles all this consistently. There is a device loader utility that checks and loads.

Notes: You must boot the system once. From then on, to change the driver, you write the new code and data onto the kernel using */dev/kmem*. There must be no changes in global data structures. Also, no device activity while reloading. You must understand static versus global data initialization — some things must be reinitialized when reloading, others are automatically reinitialized by reloading the data. The system is at risk — if you mess up badly, it can crash the system. You also must have writable executable code in the kernel.

OSx: Towards a Single UNIX System for Superminis

Ross A. Bott
Pyramid Technology, Inc.
1295 Charleston Rd.
P.O. Box 7295
Mountain View, CA 94039

OSx is a dual port of 4.2BSD and System V to a supermini by Pyramid Technology. It is not a layered system, but full implementation of both operating systems in a single machine.

The philosophy is that both 4.2BSD and System V programs should run under OSx without modification. Users should not suffer any performance penalties because of the coexistence of the operating systems.

The user interface is defined in terms of "universes." One may switch universes simply by giving the command *btl* or *ucb*. You can determine which universe you are in by the command *whereami*. Single commands may be executed in the opposite universe, and you may direct output between universes.

This system provides 4.2BSD and System V compatibility at a detailed level. There are no performance penalties and dual system transparency, essentially providing the best of both worlds.

New ½ Inch Tape Options and Trade-Offs for 4BSD on DEC VAX Processors

Robert J. Kridle
mt Xinu, Inc.
739 Allston Way
Berkeley, CA 94710

The performance of five tape units was measured in typical UNIX tape applications. They were benchmarked under 4.2BSD on a VAX 11/750. The units used were of two types: the traditional start-stop and the enhanced streaming technology tape sub-systems. The author does not recommend or disrecommend any of the units tested.

The traditional start-stop units tested were: Cipher 100X, Kennedy 9300, and NEC TD 1505. The enhanced streaming tape units tested were Cipher M891 and DEC TU-80. These systems cost anywhere from 5-20 thousand dollars.

The things measured included: average continuous data transfer rates (estimated and measured), and the measured time to complete the dump of a small file system. A recommendation for a modification to the 4BSD tape handler for the DEC TU-80 was made, which would improve its performance considerably.

Implementations II

January 20, 1984, 8:30 a.m.
Chair: Joseph S. D. Yao, Hadron, Inc.

Reporter: *Berry Kercheval*
Zehntel, Inc.
2625 Shadelands Dr.
Walnut Creek, CA

A Layered Implementation of the UNIX Kernel on the HP9000 Series 500 Computer

Jeff Lindberg
Hewlett-Packard
3404 E. Harmony Rd.
Fort Collins, CO 80525

Hewlett-Packard has implemented a UNIX kernel on top of an already existing operating system. HP-UX, as they call this implementation, is System III with the ever-popular "Berkeley enhancements," in this case the *ex/vi* screen editor. HP has also included *f77*, virtual memory, Pascal, memory shared between processes, 3-D graphics, and a data-base management system called "IMAGE."

The HP9000 Series 500 computers are based on a proprietary 32 bit microprocessor chip set with a stack architecture. It has user-transparent multi-CPU support, for which they included semaphores and a very carefully worked out scheduler.

HP-UX is based on top of an operating system called SUN (no relation to Sun Microsystems Inc.). The SUN operating system was intended to be a modern operating system to support HP's desk top BASIC system. As designed, though, it turns out to be language independent, and had several other features that made it attractive as a base upon which to build a UNIX implementation. It provided a high-level interface to the machine functions that need not be redone; was written in a high-level modular language [modcal — HP's modular Pascal language; I think it's proprietary but I'm not sure — it may merely be unsupported to customers] and is invisible to UNIX users.

There are four major sections to HP's UNIX kernel. First, memory management, which handles the user's private memory partition, virtual memory and shared memory partitions. Second, process management schedules processes to run and handles real-time stuff. Third, the file system, which needed to handle HP's diverse directory formats. Finally, the device drivers needed to be implemented so that new ones could be added dynamically.

The implementation strategy called for the implementors to layer their UNIX kernel on top of the SUN kernel. The aim was to implement exact System III syntax and semantics. Modcal was used instead of C. The HP-UX kernel was to run in privileged mode to allow access to the hardware and system data structures. Calls to SUN routines were to be made if appropriate. It was not a requirement that UNIX and BASIC co-exist.

The benefits of this approach were several. An existing high-level interface to he hardware would ease the initial "duh — how does this work" stage. Drivers for most devices already existed, and could be used without change. The IMAGE data-base intrinsic functions in the SUN kernel would make the UNIX implementation of IMAGE easy. SUN support for real-time functions, multiple CPUs, reliable error recovery, dynamic memory segments, semaphores and device I/O also made the "layered look" attractive.

Several risks also were apparent. What if they couldn't exactly duplicate the UNIX semantics? To solve this problem, HP developed a validation suite for the kernel that would verify its correctness. The other major worry was "how slow will it be?" Would there be too much translation from UNIX syntax to SUN syntax? As it turned out, performance was good, but tuning of the SUN kernel was necessary. Real time performance was also good.

In the process management section, the match between SUN and UNIX was good. The scheduler had static priorities and provided for system daemons. The major change was adding a *fork()* system call; the SUN kernel had no concept of "cloning" a process. Signals were easily integrated with the SUN trap handlers and periodic clock interrupts.

The filesystem provided the necessary hierarchical directories, but "special files" necessitated major changes to the SUN kernel. Things like *mount()*, *access()*, *pipe()*, and *chown()* all needed to be done from scratch.

The I/O system was a good match. The tty driver needed extensive modifications, though, and the internal keyboard and CRT drivers needed to be made to look like a terminal.

Memory management followed a simple model making the transition easy. The main work involved was in extensions to UNIX to support shared memory segments, mapping of files to memory segments, and virtual memory for large CAD applications — one of the major reasons for building the system in the first place.

Implementing the *exec()* system call was a significant effort, but rested on the SUN kernel operations. Shared libraries were implemented, but are available only for standard commands — they cannot be used by user programs. Other miscellaneous functions — automatic power-on boot, clock traps, multiple CPU support — were added.

Writing Device Drivers for Xenix Systems

Jean McNamara, Paresh Vaish, Richard N. Bryant
Intel Corp.
5200 NE Elam Young Pkwy.
Hillsboro, OR 97123

Mr. Bryant presented this "how-to" talk for novice writers of device drivers. Since Xenix is a direct UNIX port, these comments should be generally useful.

What a device driver does is manage input from and output to a hardware device. There are two types of device drivers, block and character. A block device driver is one that transfers entire blocks at once, such as raw disk or tape drives. Character devices are typically ones that communicate one

character at a time, such as terminals, printers or plotters. In practice though, a character device is anything that isn't a block device. Block devices are generally simple. Most of the work is done by the basic I/O routines. Character devices can be quite complicated; the tty driver is a particularly baroque example.

All Xenix devices drivers are written in three files. A name for the device is invented, such as "tty" or "mt". "xxx" will be used in examples. The first file is called *xxx.h*. It contains the definitions of data structures, device addresses, control register masks, useful macros and other device-specific information. The configuration file, *cxxx.c*, contains the actual instances of the driver's private data. It lives in a "cfg" directory, and can be easily modified by customers with binary licenses who wish to reconfigure their system. The "cfg" directory is a Xenix-specific feature. Finally, the *xxx.c* file contains all the routines that do the actual work of the driver.

In a "cooked" block driver, nine routines are necessary. They were described briefly:

init() initializes the device.

open() opens the device, enabling device interrupts.

close() closes it.

read() handles requests to the driver to read a block from the device.

write() handles requests to the driver to write a block to the device.

ioctl() does device dependent magic things.

strategy()
receives I/O requests from, say read and write, queues them and starts I/O.

intr() handles device interrupts like "I/O complete," and restarts the next I/O request.

start() pokes the device to perform actual I/O transfers.

Raw block drivers are much simpler. *read()* and *write()* handle user requests for I/O, and do it, calling the system routine *physio()*. *ioctl()* does the same magic stuff.

Character devices are more complicated. *init()*, *open()*, *close()*, *read()*, *write()*, *ioctl()*, and *intr()* do similar things to the corresponding block device driver routines. *proc()*, however, is a System III/V requirement; it sets the "line discipline," required of all character devices.

l_open() initializes the tty (or whatever) structure.

l_close() clears it out.

l_write() these two handle clist processing.

l_ioctl() modifies the "tty" structure.

l_input()
places incoming characters into the input queue.

l_output()
takes characters from the output queue and sends them to the device.

In order to make a kernel with a new driver in the Xenix environment, the *xxx.c* file must be installed in */sys/io*, the *xxx.h* file in */sys/h*, and the *cxxx.c* file in */sys/cfg*. The makefiles must be altered to include the new driver, then a *make* in */sys/io* and a *makexenix* in */sys/conf* should do it.

First, though, there are four switch table that need to be updated. *initsw[]* lists the *init()* routines for all block devices. *intrsw[]* lists the interrupt routines. *cdevsw[]* lists *open()*, *close()*, *read()*, *write()*, and *ioctl()* for all character devices, and *bdevsw[]* lists *open()*, *close()* and *strategy()* for block devices.

Quite a few useful routines for the device driver writer exist in the kernel. *sleep()* and *wakeup()* allow process synchronization. Generally when a driver must wait for an event to happen it calls *sleep()*; when the event happens *intr()* calls *wakeup()* and the driver resumes processing.

Mutual exclusion is done with *spl0()-spl7()* and *splx()*, which set and restore the interrupt level. Finally *timeout()* will invoke a function some time in the future.

Mr. Bryant mentioned that the best way to write a new device driver is still to copy an existing one, but that this talk should help unfortunate souls without source licenses.

Porting Xenix to the Unmapped 8086

John Bruno Hare, Dean Thomas
The Santa Cruz Operation
Santa Cruz, CA 95060

At the beginning of his presentation, Mr. Thomas indicated that the paper would be published in *CommUNIXations* and that reprints would be available from The Santa Cruz Operation.

The Intel 8086 can operate in three modes, depending on what kind of memory management is done. The purpose of this presentation was to show that the "Small Model" 8086 is sufficient to support UNIX. Such machines as the IBM PC, Victor 9000, and Convergent Technologies' Integrated Work Station have no memory management system, and must perforce use this model.

The 8086 can access one megabyte of physical memory. Addresses are formed by combining a base pointer, which points to a segment on a 16K boundary, and a sixteen bit offset, which allows segments to be up to 64K long. There are four registers that can be used as base pointers called the "code," "data," "stack" and "extra" registers. Unfortunately, there are no provisions for intersegment protection or segment bounds checking.

In the "Small model" the code, data and stack segments are all the same. This corresponds roughly to the type 407 executables on PDP-11's. The "Compact model" has the text segment distinct from the data/stack segment, and corresponds to the "Separate I/D" executable on some PDP-11's.

The SCO Xenix port supports both of these models. They make relocation easy, as all the kernel needs to do is change the segment registers and move the code. The middle model, in which more than one text segment can exist, must trap intersegment calls so the kernel can change segment registers.

Two problems exist using the middle model, though. If there are any absolute addresses in the code of a program, they must be fixed up at load time. SCO includes a "fixup table" in the a.out module for this purpose. Another problem was swapping. If, say, absolute addresses are on the stack, and the processes gets swapped out and then swapped back in at a different address, it's in trouble. Either some means must be given to detect and fix these at swap-in time [ecch] or the process must be locked in core [ecch]. SCO's solution was to give the executing process high priority to reduce the likelihood of swapping, and they swap to the same physical address. Luckily most programs do not need to use the middle model; two which DO are *vi* and the RMS-Cobol Animator.

Interesting topics arose during questions. A malicious user can still trash the system, though. The kernel checks certain data structures for changes on each system call, and if they have changed, assumes the worst, issues a segmentation violation and kills the process. The large model is not supported, but is not written off yet either.

Transparent Implementation of Shared Libraries

Curtis B. Downing
Bunker Ramo Information Systems
35 Nutmeg Drive
Trumbull Industrial Park
Trumbull, CT 06609

Frank Farance
Systems Theory Design Corp.
555 Main Street, Suite 705
New York, NY 10044

Reporter: *Karen Summers-Horton*

This talk described a long needed feature in UNIX — a way to share large libraries among different running processes. It is not only useful to share big application libraries, but also the standard C library.

Shared libraries are defined as sharing a text segment among different processes. No attempt is made to share data segments. The traditional UNIX shared text mechanism (the **-n** option to *ld*) only allows sharing of the exact same text. Different programs loaded with the same libraries can't be shared. True shared libraries are better.

Various possible methods were discussed.

- Static links done by *ld*: the problem here is that if the code is changed, you must relink everything.
- Dynamic links (like MULTICS): you need hardware and software support.
- Quasi-static links: you must fake undefined symbols.
- Quasi-dynamic links: you must install stubs to shared library routines, and have the stubs dynamically link.

The method used is quasi-dynamic links. They do a lookup for the first call only, after this they must still go through the stub, but the overhead is very low.

The hardware and software required: standard *ld* and *cc*. You must have shared memory. You need a compiler with relocatable code, additional support to produce the stub, segmentation or paging memory mapping.

Things that belong in a shared library include *printf*, *scanf*, and applications support (in this case, a database management system). What does not belong in a shared library include system calls, low usage routines, and high performance routines.

One problem is that, since System V shared memory was used, the user must call a support routine to set it up, so this is not transparent to the user. Also, their compiler did not do enough relocation.

UNIX File Systems Optimization on Microcomputer Systems

David Robboy
Intel Corp.
5200 NE Elam Young Parkway
Hillsboro, OR 97123

This talk discussed different methods for making the UNIX filesystem faster. This has been a problem with UNIX for years, since the original filesystem was one of the major bottlenecks in UNIX. The object is to essentially "make your disk faster." This is even more important now since the new inexpensive winchester disks tend to be slow.

There are three possible bottlenecks in a filesystem. It can be I/O bound, limited by the transfer rate on disk (e.g. backups). It can be CPU bound, e.g. small data requests. It can be access bound, limited by the time to do seek or search operations.

Possible optimization strategies include getting faster hardware; using read ahead (this is good for sequential access in an access bound system, but bad for random access); buffer caching, to re-use data already read; and organizing the filesystem, for access bound systems, to reduce the scattering of files, and to reduce seeks.

They have already tried getting a faster controller, which increases the raw sequential I/O, but not multi-process throughput: real systems are usually access bound. They also tried a faster CPU, but it didn't help in proportion (only a factor of 1.5 or 2 for a CPU that's four times faster).

The 4.2BSD filesystem was considered. Berkeley claims a tenfold improvement in performance over the old 4.1BSD filesystem, due to a larger block size, clustering, and the rotational allocation of

blocks.

Having a large block size is important. It gives the effect of read ahead, reduces scattering of data on the disk (because there are fewer, larger pieces). It also wastes space when small files are allocated, taking an entire block. Berkeley uses fragments, which avoid the wasted space, but are complex. They also cause fewer buffers in the cache (since each buffer must be larger), this is a disadvantage.

Cylinder groups partition the filesystem. This reduces file fragmentation, and arranges that files are closer to inodes and directories.

Rotational allocation may not be useful for multi-user random access. It introduces additional complexity, and must parameterize hardware dependencies.

There are other strategies. You can speed up memory intensive functions, such as memory to memory copies. You can improve buffer cache management, by adding more buffers, or having separate buffers for data and inodes. DMA data transfers can be used to go directly from disk into user space; this is good for program loading, and avoids clogging buffers with programs. You can maintain continuous files, which contributes to fast program loading. You can sort the disk free list.

The speaker concluded that UNIX is access bound, especially on micros. You can focus on keeping file blocks together. Some strategies enhance each other. Filesystems will always be bounded by either access, CPU, or I/O.

Communications

January 20, 1984, 10:30 a.m.
Chair: Mark Horton, AT&T Bell Laboratories

Reporter: *Berry Kercheval*

A UNIX to VAX/VMS Communications Link

Clifford N. Cary
Creare R&D Inc.
Box 71
Hanover, NH 03755

Creare has developed a system which enables a UNIX system to communicate with a VAX/VMS system. Many users can use the facilities for file transfer, remote terminals and virtual disk simultaneously. It was designed to support multiple error-free channels using layered software. It uses a typical UNIX command syntax.

A user process (or "transient" process as Mr. Cary termed them) communicates with a "net layer" background process. The "net layer" in turn communicates with a "data link" background process. The "data link" process talks to a device driver in the kernel which drives the communication line.

The "data link" process and the driver implement the DDCMP protocol over an RS-232 link. [I believe the standard tty driver is used, but I am not sure.]

The "net layer" acts as a multiplexer for multiple users, serializing user requests and ordering data coming from the other end of the link.

The system is unbalanced; that is, the UNIX system is the "master" and the VMS system is a "slave." [Let it be recorded here that the scribe resisted a strong temptation to editorialize.] Named pipes are used for data transfer from transient processes to the net layer, since they provide a "well-known" place to send data. Other architectural details of the implementation were discussed.

With the addition of a special device driver, "virtual disk" service can be provided. The UNIX transient process opens the device, which causes the kernel to spawn an instance of the disk server.

The server opens a channel to the VMS system, and from then on, a VMS file looks like a UNIX block device.

Mr. Cary claimed 400 character per second throughput on a 9600-baud RS-232 line.

Loving *uucp*, or How I Spent my Summer Vacation

P. Honeyman
Princeton University

D. A. Nowitz
AT&T Bell Laboratories
Murray Hill, NJ 07974

B. E. Redman
AT&T Bell Laboratories
Whippany, NJ 07981

Messrs. Honeyman, Nowitz and Redman spent their "summer vacation" rewriting the *uucp* software package, with the aid of a host of helpers — the list looks like a "Who's who on Usenet."

One change to "vanilla" *uucp* is the implementation of dynamically created per-site spool subdirectories. This greatly reduces the load of directory searching — a quadratic algorithm executed *in the kernel*. Also, since the sitename is in the name of the directory, sitenames can be fourteen characters, although this could cause problems for machines still running the older versions.

Logging and administrative information is much improved. Multiple *uuxqt*'s — one per site — are permitted, with the use of a -s flag to tell *uuxqt* to work only on a single site. Locking is improved. Instead of relying entirely on age to tell if a lock is stale, the pid is extracted from the lock file and signal 0 is sent to that process. By examining the error return from *signal()* it can be determined if the process still exists. This is possible only in USG UNIX, though the hack for 4.1 or 4.2 is easy — two lines in sys4.c.

Multiple L.sys entries finally work right. Many modems are supported. Other types of devices such as PBX's can be easily handled with "chat scripts."

cu has also been enhanced to use the same information about modems, switches, PBX's and ACU's that *uucp* uses.

Errors are returned to the originator. More tools are available to check on status of the *uucp* system or individual jobs, or to kill or reschedule jobs. Exponential back off on retry of failed connections has been added, and scheduling has been removed from *uucico*, and put in a separate program.

New Implementations of *uucp*

P. Honeyman
Princeton University

D. A. Nowitz
AT&T Bell Laboratories
Murray Hill, NJ

B. E. Redman
Central Services Organization

Security aspects of the new implementation of *uucp* introduced in the previous presentation were discussed here. Security objectives of the new design were to provide restrictive defaults, more flexible options, a more easily understood format for specifying the options, and to allow command permissions by site instead of compiling it into *uuxqt*.

The default permissions are obtained from the minimal permission file entry "LOGNAME somesite". This allows *somesite* to send files to the public directory, and execute *rnews* and *rmail*. Nothing else.

For *somesite* to receive files, "SENDFILES=YES" must be added to the permissions line. To allow *somesite* to request files, "REQUEST=yes" must be added. To allow *somesite* to write to other directories, "WRITE=dir1:dir2:dir3" is added.

An example of an "open system" would be "LOGNAME=uucpok REQUEST=yes READ=/ WRITE=/". Additionally, commands can be specified per site.

Future work will include encryption on spooled and transmitted data and gateway facilities to other networks.

[Mr. Nowitz later told me that this version of *uucp* was not a released product, and therefore not available outside of AT&T Bell Laboratories yet.]

Mapping the *uucp* Network

Rob Kolstad
CONVEX Computer Corporation
Dallas, TX

Karen Summers-Horton
2843 Valcour Court
Reynoldsburg, OH 43068

Reporter: *Jerry Deroo*

The speakers gave a quick overview of the size and growth of the *uucp* and Usenet network. A quick synopsis of the *uucp* network:

- more than 2100 sites
- sporadic connections
- low cost
- store/forward propagation
- some gateways to other networks
- a relative anarchy

Usenet, which is a subset of the *uucp* network, currently connects about 700 sites in North America, Europe and Australia.

To tame the beast, they are setting out to map the approximately 8000 *uucp* connections in a way which is usable by routing software. They are asking for public information concerning all sites. They anticipate that the project will take from six to twelve months to complete.

Panel Discussion

Following formal presentations was a panel discussion with Steve Bellovin, Peter Honeyman, Mark Horton, Rob Kolstad, D. Nowitz, Armando P. Stettner, Karen Summers-Horton, and Lauren Weinstein.

Participating from the audience were Rob Pike, Erik Fair, Andy Tannenbaum and many others. Several topics relating to the future of Usenet and the *uucp* net were covered with perhaps more heat than light.

Graphics

January 20, 1984, 1:30 p.m.
Chair: Noel Kropf, Columbia University

Reporter: *Jerry Deroo*

UNIX as a Development Base for High Performance Graphics Applications

Thomas Ferrin
Computer Graphics Laboratory
School of Pharmacy
University of California
San Francisco, CA 94143

This talk described work undertaken at UCSF to model molecular structures. Standard approaches to molecular modeling has resulted in models that were very cumbersome, and took a large amount of time and effort to create and change. An effort was undertaken with these objectives:

- allow the display and manipulation of multiple models simultaneously
- allow the selective display of a region of a model
- permit rotation and transformation of a model
- allow for the modification of a model (e.g., bond rotation)
- use color to enhance the representation of the model

To attain these goals, the following things were required of the supporting operating environment:

- an efficient and portable applications programming language
- "real time" response
- tools for optimization and documentation
- good programming examples to guide development

They chose UNIX in 1976, as it was a time sharing system to which people considered "experts" had already switched.

There were some disadvantages to this choice:

- no graphics hardware vendor support for the operating environment
- missing a decent Fortran compiler, which their users were accustomed to having.

They set out to write their own device driver and a graphics subroutine library. They made changes to the kernel to allow a real-time process to exist. The changes for real time process required about 40 lines of kernel code, and took about one month. They designed and implemented their own database and access methodology, optimized for protein and nucleic acid molecules.

The speaker then displayed a series of slides, and a motion picture of the system in action.

DAPS and GSDL: A Procedural Approach to Graphics

Christos Tountas
Graphic Information System Technology Inc.

This talk described the uses of a graphics system that has, as one of its interfaces, a procedural language to describe the graphics objects to display. Repetitive patterns can be generally described once, and then given parameters to indicate where the object is to be displayed.

The system supports many different output devices, and can present one of several different user interfaces.

There is presently a gap between a graphics package that has a collection of pre-cooked graphic objects that the user can chose from, and a package of graphics subroutines that must be enclosed with a complex user-written program before it can actually do anything. This package attempts to bridge this gap.

An example of the short-comings of most systems is that there is no way of obtaining automatic re-generation of an image after the image is changed: the entire image must be specified again. This

suggests that parameterization would be a valuable feature: to enable the user to specify an attribute of the image in terms of other attributes.

The package supports the basic notion of move and draw, and supports a flavor of turtle graphics user-package interface. It is possible for a user to have a library of graphics objects from which other objects can be built.

The package operates in either two or three dimensions. It supports hidden line removal on a user controllable basis.

The speaker has an impressive slide show, demonstrating what the package can do.

The Design of a Dedicated Graphics Subsystem for a UNIX Machine

Or: How to Make Pictures in Real Time

Jack Burness
MASSCOMP
1 Technology Park
Westford, MA 01886

The speaker began by presenting a short history of the evolution of display systems. The path was:

- display lists: image drawn from memory in a "stroke refresh" manner.
  Problem: if display list was too long, flicker was likely.
- frame buffer: since operating system evolution makes display lists in user memory rather difficult, short of giving the display processor its own set of memory management registers (and this does not get by swapping), the display list is moved to the frame buffer.
- put the frame buffer into the terminal.

  These last two approaches make it difficult for the user process to get at the "display list"
- implement a ring buffer in user space that the display processor can access via the system bus. This is a hybrid approach, but works well if the display processor can lock user memory pages for the ring buffer; now the operating system can direct the display processor, since it is essentially doing DMA with it.

The speaker describes a system that is the result of this type of thought process. The system can support up to 15 virtual terminals on one display processor (15 is a magic number!). Each one is a full featured terminal, and is an overlay on the display processor. Each virtual terminal may have up to 15 viewports, or windows. (Again, that magic number.)

The real problem is one of data management: what to do with the data that is overlayed by another window? If you move it somewhere, it is possible to have to move 8 Mb of data. These moves have to be rather quick. And where to put it?

Image Processing Work Station

Dale K. Hensley
TRW
1 Space Park
Redondo Beach, CA 90278

This talk described the evolution of a software package for performing image processing.

The software has undergone a lot of change. As the software was originally environment specific, each move caused a re-write. They finally moved to UNIX, and have stuck to basically V7 system calls, to enhance the portability. Despite their intentions, however, every once in a while device dependence crept into the software. Things like the size of an integer would be hardwired in every once in a while.

Multiple copies of #include files get you in the end, as well.

They ran into one problem, in that no one had ever written anything on how to write a device driver. This results in device drivers for exotic graphics boards that look like line printer drivers, if one is not careful.

Their current configuration is a VAX 780 with three Deanza image processing workstations. Each of the Deanza devices has two Unibuses, so they use one to connect the device to the VAX, and hang an LSI-11/23, through a Qbus/Unibus converter, off the other. The 11/23 runs the 2.8BSD kernel, which buys them a few performance improvements:

- the 1K block filesystem improves throughput
- expanded memory capacity of the 23+ lets them run 2 Mb of memory, gives them a memory disk
- they use memory maps for large copyins/copyouts. This speeds up image manipulation and polygon filling.
- The VAX can access the 11/23 memory to do DMA.

They are now planning to develop a product based on this work, to deliver a low cost image analysis system. The system would cost between 30 and 40,000 dollars, and be made up of:

- LSI-11 (23+ or 73)
- tape drive
- 512 by 512 monitor
- array processor

One aspect of the implementation is that the user is not exposed directly to the UNIX environment running on the 11/23. This means that the users do not require any UNIX knowledge.

This talk was pressed for time as well.

Real-Time UNIX

January 20, 1984, 3:30 p.m.
Chair: Reidar Bornholdt, Columbia University

Reporter: *Berry Kercheval*

Real time Extensions to the UNIX Operating System

Brian Look, Gary Ho
Hewlett-Packard
Data Systems Division
11000 Wolfe Rd.
Cupertino, CA 95014

A real-time system reacts predictably and in a timely manner to events in the "real" world. Applications for real-time systems include process control, machine monitoring, instrumentation or laboratory automation.

The primary problem in integrating real-time processing into the UNIX environment is in management of the real-time process. Basically, the running process must be pre-empted if a higher priority process becomes ready to run, say, in response to a device interrupt that must be serviced NOW, not in fifty milliseconds. Thus the need is for priority based pre-emptive scheduling in which the priorities do not degrade over time. Round-robin scheduling of real-time processes is necessary, not time-slicing, and it must be possible to control the priorities of the real-time processes.

The scheduler must have micro-second resolution, and not drift over time. Dispatching of processes must be fast; the system often cannot wait for a long system call to complete, so system calls must be interruptible.

Shared memory is often useful. Either Bell's segments or Berkeley's mapped files will do. Some things need to be locked in memory: shared segments, memory-mapped files and perhaps very high priority process images.

Process synchronization presents some interesting problems. A guaranteed maximum interrupt response time is necessary, as are reliable signals, shared memory and semaphores. The isolation of critical regions into areas protected by semaphores enables higher priority processes to be dispatched without worrying about whether they will destroy critical data structures.

File system modifications may also be needed. Pre-allocation of disk space would save critical time in servicing high-priority processes that need to write to the disk. Device and file locking is already provided in 4.2. User control of buffering — better than just calling *sync()* — would be nice, as would contiguous files to ensure faster startup. Scatter read/gather write was also mentioned as desirable.

For most real-time applications, user-written device drivers are inevitable, but folks stuck with binary-only licenses are out of luck here. Non blocking I/O with an "I/O complete" signal is attractive in this context. Synchronous I/O multiplexing, like *select()* in 4.2, and preservation of I/O sequencing is important.

Real-time privilege control can be implemented with two kinds of user. Ordinary users are normal users, while real-time users can invoke processes with some or all of these real-time features. Two possible mechanisms for invoking real-time features would be special files that automatically get the features when loaded, and *ioctl()* calls to invoke them.

In summary, the presenters feel that a combination of the best features of 4.2BSD and System V, with real-time priorities, memory locking and user-written drivers will ensure good real-time performance.

UNIX Performance Optimization — UNIX/68000/SKYFFP

Joseph F. Germann
Sky Computers, Inc.
Lowell, MA

The SKY floating point processor can increase performance of both floating-point and integer functions when integrated into a 68000 system. One might think that with all the work done on the kernel over the years, it would be pretty tight, and there would not be much room for improvement. One might be wrong.

The SKYFFP is a microprogrammable 2901 bit-slice floating point processor on a single Multibus Board that can perform a 16×16 bit multiply in 120 nanoseconds — about the time to access some fast memory chips! It runs as a slave coprocessor to the 68000, and has six registers in the I/O page for command, status, microcode and data. It will do complex, trigonometric, logarithmic and long integer — signed and unsigned — operations. The device driver for this beast is not a normal driver. For one thing, it can support multiple users at once.

The basic paradigm for using the SKYFFP is to load the command, load the data, and then read the result when it is ready. An add will thus take a minimum of four 68000 instructions. The SKYFFP will then hold the bus signal DTACK until the operation is done.

Usually floating point arithmetic is done with runtime routines. The arguments must then be saved on the stack, the routine must be called, it does its thing, then saves the result and returns. A typical floating point add may take from 70-150 microseconds, with 18 microseconds for procedure call overhead. The SKYFFP is integrated into the C language using a mix of in-line code generation and run-time libraries. Fast operations are coded in-line (by a special compiler) to avoid the procedure call

overhead. They include add, mul, sub, cvt, and integer ldiv and lmul. Longer functions — log and trig, etc. — are coded in runtime libraries as the overhead penalty is not as high a fraction of the compute time. A 32 bit IEEE FP division can be done in 25 microseconds, for example, while a tangent may take 180 microseconds.

So what's the bottom line? In a Codata 68000 system with the SKYFFP, and five of the most commonly used kernel math operations on long integers microcoded into the SKYFFP, compiles were 100% faster, I/O intensive processes were five times faster, and memory intensive processes were 40% faster. (Measured with the Teus Hagen/Andrew S. Tannenbaum benchmark). The naked 68000 ran 70,000 Whetstones/second, and with the SKYFFP it jumped to 294,000 Whetstones/sec.

Life with UNIX in Real Time

Steven T. Polya, Jeff Barnes
Contel Information Systems

Messrs. Polya and Barnes presented a case study of Contels implementation of a real-time network control system on a Codata 300 68000 system with UNIX. They had a binary license, but could write device drivers. What they felt they needed was good inter-process communication, a commonly addressable memory resident database, priority based scheduling and non-interruptible processes.

A Z80 based network controller neeeded to interface with the 68000 UNIX processor. This was done by using a 128K ram card mapped in with *phys()*. Two FIFO ring buffers were implemented for interprocessor message storage.

Since many processes use net services, IPC was also implemented using ring buffers in the left-over RAM on the card. This enabled them to do away with pipes and signals for this service, as real-time processes should never sleep. Access control for these data structures was added to the kernel.

An interesting dynamic priority assessing scheme was presented.

An event pseudo device driver was implemented to enable a process to sleep on an event. A master process finds the events and signals the sleeping process to awaken.

To eliminate swapping problems, lots of RAM was installed, and processes could be locked into it.

Integrating a Peripheral Floating Point Processor in UNIX

Eryk Vershen
UniSoft Systems
739 Allston Way
Berkeley, CA 94710

Mr. Vershen presented a brief outline of his integration of the SKYFFP into a UNIX system. This board was described in a previous talk.

Maximum speed was desired, so system calls to the SKYFFP were considered too slow, but using *phys()* to access the SKYFFP registers was very MMU dependent, and besides only the super-user can do it. A hook was added to the SKYFFP device driver to allow it to perform the *phys()* for the client process. This enables a malicious user to zap the microcode, but TAANSTAFL.

The system call overhead in the UniSoft kernel was 250 to 400 microseconds. Using simulated floating point, additions took 270 microseconds, so clearly the system call approach was too slow. In-line code gave the best performance, reducing add times to 80 microseconds.

Clippings

The first item in clippings is the rest of an article from "The Gazette", February 1984 printed by Sydney University, accidentally omitted from volume 5 number 2. Under it is an advertisement, taken from "Australian Micro" May 1984, for a product we could all use at times.

A story on AT&T's entry into the computer market appears on the next page, taken from the "Sydney Morning Herald", April 2, 1984. The third page of clippings come from "Computer World" April and May 1984.

The final four pages of clippings are from "Whats New in Computing" February, March and April 1984.

walk out of his office knowing that letters to all the people concerned will *already have arrived* at their 'mailboxes', at the cost of three cents per A4 page.

By the next morning, he will probably have received half a dozen replies, and other replies will continue to come in during the day, whether he's in his office or not. The next time he uses his computer, he will be informed that there is mail waiting for him and be asked whether he wants to read it.

One of the greatest advantages of the Australian system is that only the name of the addressee, and the name of the computer he or she is using (e.g. Bob, Basservax) needs to be typed in. In the American system, by contrast, the names of all the computers in between need the sender and the receiver to be listed so that the message can find its way.

'This is not so bad if there are only half a dozen computers in the network', says Dr Kummerfeld, 'but it is very unwieldy now in America, where they have hundreds of computers linked together'.

The reason for the simplicity of the Australian system is that messages are passed on automatically from computer to computer. As the messages are passed along, each computer works out all the next stages, calculating the fastest way for the message to get to its destination. This is important not only to save time in addressing, but to overcome the problem of one or more of the computers in the network 'crashing'.

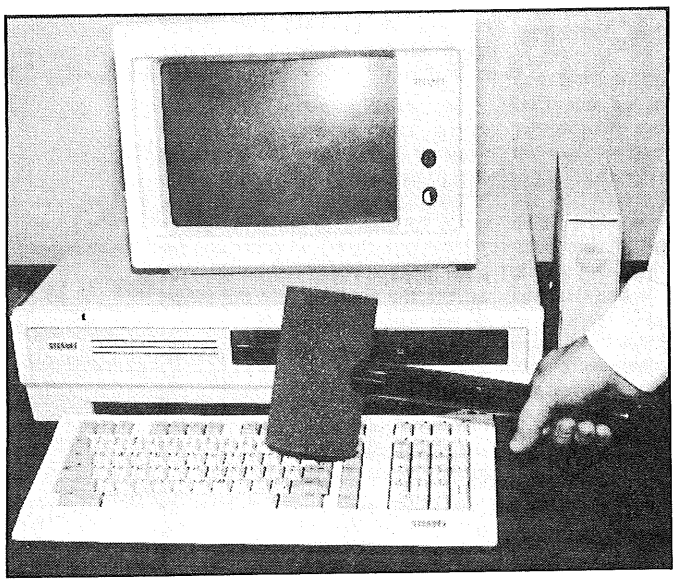
'Networks can change from minute to minute', Dr Kummerfeld explained, 'and if a link is broken because a computer has shut down, a message can easily be lost irretrievably. In the Australian Computer Science Network, the message is simply re-routed around the broken link. The message is also re-routed around areas of heavy traffic, should these occur.'

Further advantages of the Australian system are:

- It can immediately adapt to any new members added to the network.
- It allows users to transfer entire data bases.
- It is much more adaptable to different 'carrier systems'.

As if these advantages were not enough, Dr Kummerfeld and Mr Dick-Lauder, have just finished a major re-design and improvement of the software, and are now testing their new program on computers in the University of Sydney before gradually introducing it throughout the entire network.

The re-designing project has been funded as part of a larger research program made possible by a two-year $90,000 grant from Telecom. Telecom is introducing an Australia-wide computer network, AUSTPAC, and has funded the work at the University of Sydney to help develop computer 'protocols'.



The Bit Banger in operation.

Friendly Bit Banger

A UNIVERSALLY applicable computer peripheral product designed to relieve frustration caused by momentary upset in hardware and software user friendliness has been released in Australia by Insystems Pty Ltd. Called the Bit Banger, it can be used with any system configuration and on any computer from a Cray mainframe to the smallest portable. The unit can be installed by the user, and comes with all necessary parts, such as instructions and hanging straps which will fit any terminal manufactured outside the USSR.

Further information: Insystems, 333 Moray St, Sth Melbourne 3205. Tel: (03) 690 2899.

AT&T not ready to grapple with IBM

By ANDREW POLLACK

THE title bout has been postponed. The challenger isn't ready yet.

The computer industry has been bracing for a battle of the titans between newcomer American Telephone and Telegraph Co and the IBM Corp but many analysts said that when AT&T made its much-awaited entry into the computer business last week, its product announcement only showed that - at least for now - AT&T is avoiding a direct confrontation, and still has some gaps in its strategies.

Still, they said, despite its initial shortcomings, AT&T is clearly one of the major new challengers for a large share of the computer business. The company introduced six models of its 3B series of computers, ranging from a $US10,000 ($10,600) desktop microcomputer capable of supporting several users, to a $360,000 super minicomputer that is fault tolerant, meaning it has backup circuits that allow it to keep operating even if some fail.

The company also introduced two network products that can tie various computers together. One was a local network, 3B-net, that allows various 3B computers to talk to each other. The other was a system that would allow personal computers, including IBMs, to connect to the AT&T 3B computers. This should help AT&T sell its computers to large corporations that are buying thousands of personal computers.

Company officials indicated that in 1984 most of these computers will be sold to the Bell operating companies that already use hundreds of 3Bs. Nevertheless, the initial product introduction was viewed as an important indication of AT&T's strengths and weaknesses as it seeks to enter the new markets it gained access to by spinning off its local telephone companies.

AT&T did manage to convey that it will be a broad-line supplier. Instead of introducing one product, it introduced a range — and made it known that more products will be coming. In addition, the company showed that its line would be compatible from top to bottom. Users will be able to use the same programs on its small and large computers, making it easy for a company to expand its computer capacity.

AT&T also wants to position itself as a reliable supplier, one that because of its size will not fade away into bankruptcy and leave its customers in limbo. "We're into this business in a big way and we're in it to stay," said Mr James Olson, AT&T's vice chairman.

The company has also carefully picked its initial entry points to the market. Its products stress both communications between computers and the reliability that has been a hallmark of the phone system.

The product strategy also clearly avoids the areas in which IBM is strongest, mainly the mainframe market for large corporate computers, and the personal computer business, which AT&T will enter later this year, when more software is ready for its machines.

AT&T decided to pursue the minicomputer market which is machines costing tens of thousands to hundreds of thousands of dollars used largely for scientific and engineering tasks. It also chose to enter the relatively new market known as super microcomputers - the desktop computers that fit between personal computers, used by a single person, and minicomputers that can be used in networks. That will put AT&T into competition with companies such as the Digital Equipment Corp and the Data General Corp.

Nevertheless, the product introduction also highlighted several AT&T weaknesses. The choice of minicomputers shows AT&T's technological, rather than marketing, bent.

The minicomputer market has always been known as one that competed on the basis of computer speed and price, rather than on sophisticated marketing. This seems to fit AT&T's tradition the best. In many other markets, such as telephone equipment, the company has stressed reliability, while other companies have stressed marketing.

AT&T also does not have much software for its computers yet. Instead, it is selling its products initially to other companies: known as value-added remarketers, who will write software for particular industries and resell the machines. It will also sell to large companies that can write their own software.

This is also the traditional minicomputer approach: sell computers to sophisticated users who can write software for them. But companies such as Digital and Data General are now moving away from that strategy of "pumping iron" in search of broader markets.

"The fact that AT&T is doing that limits their immediate impact on the market," said Mr David Moschella, an analyst at the International Data Corp, a market research firm.

Another potential drawback is that AT&T's prices are somewhat high, especially considering that, initially, the company will compete on price and performance alone. The company, which has produced some 3B models internally and even bid for some federal contracts with them, lost out to Data General last year for a major contract from the United States Forest Service.

Analysts said AT&T's pricing was comparable to Digital's, but that Digital's product line is old and its prices are higher than other minicomputer makers. Besides, added Steve Cohen, an analyst at the Gartner Group, a market research firm: "When you're the new kid on the block, is being comparable good enough? I just don't see it being a shot in the gut to DEC".

The top of the new line is the 3B20D, which is a super minicomputer used widely within the telephone network. The computer, priced at $36,200 in a basic system, is really two computers in one, meaning it can keep running if one computer fails. AT&T hopes the computer could be sold for use in reservation systems and other applications where a computer failure could cause major losses.

The 3B20S is a single computer version that can support up to 100 users on terminals and will have an entry price of $244,700. The 3B20A is a more powerful version, up to 1.8 times as powerful as the 3B20S. It sells for $US351,000 for a complete system, and an upgrade to turn the model S into the model A will sell for $128,000.

The 3B5 series are minicomputers based on AT&T's 32-bit microprocessor. The model 100, which can suport 30 users, costs $61,000. the model 200, which can support up to 60 users, costs $78,000.

Analysts were most intrigued by the 3B2 model 300, a desktop computer that can support up to 18 users working on separate terminals. This product also uses the 32-bit microprocessor and has an entry price of $10,590, without the terminals.

(New York Times)

Explosive Unix growth tipped

By Stuart Corner

SYDNEY — An explosive growth in the demand for Unix-based systems, generating a market worth several thousand million dollars by 1988, has been forecast by Robert Marsh, chairman of Plexus Computers Inc.

Speaking at a Unix symposium organised in Sydney, Marsh said the technology of Unix tended to obscure its power as a force in the commercial market. He said there was a rapid growth of interest in Unix during the past year in the US and claimed five per cent of minicomputers running Unix at the end of last year would increase to about 15 per cent by the end of this year.

An impressive array of international celebrities drew about 150 delegates to the symposium at Sydney Opera House. About two-thirds are thought to have been from commercial rather than academic backgrounds.

"More suits than thongs," one commentator said. "But still too many thongs."

The commentator said one speaker who chose to address the assembly while clad in shorts, open-necked shirt and thongs represented an academic attitude from which Unix needed to divest itself if it were to become a success in the commercial world.

Three delegates contacted by *Computerworld* all felt the papers presented were biased too strongly towards the technical aspects of Unix. One said the venue was excellent, the production good and the speakers very poor. He claimed some were inadequately prepared and others rehashed versions of papers presented at the Unix User Group meeting in Sydney in February (CW, Feb 24, page 2).

The opening address was delivered by Dr PJ Hagan, assistant secretary of the planning branch, Technology Development Division, of the Federal Department of Science and Technology. Hagan confirmed the government's intention to support software development in Australia and singled out Unix as being of particular interest because of its independence from individual manufacturer's hardware.

Organisers of the symposium, James Conran Pty Ltd, refused *Computerworld's* request to cover the event. This article has been compiled from information supplied by attendees.

Japanese standard

TOKYO — AT&T Japan is reported to be creating a standard Japanese version of Unix in conjunction with Tokyo University. It is said to support Kanji characters. This will be in competition with the original Japanese version, developed by Toshiba Corp. Sources said standardisation was long overdue because several vendors were producing different and incompatible versions of Unix.

Chinese Unix system

SHENGYANG — Shengyang Computing Institute, which is under the China Academy of Sciences, has developed a timesharing operating system which the institute says is fully compatible with Unix. In addition to all standard commands and routine programs supported by Unix, the Institute's Shenix also supports Fortran 77, C, Basic and Pascal.

OFFICE AUTOMATION SYSTEM

The Fortune 32:16 is a simple to operate integrated office automation package whose word processing is moulded around Wang software but has more than 40 improvements, including the ability to bypass the menus. The simple, easy to follow menus and Fortune's Help facility allow the operator to be taught by the system on-line without reference manuals. Any Wang operator can transfer to Fortune without any additional training and a Wang to Fortune conversion utility enables existing Wang users to transfer their data directly to the Fortune system. Fortune's For: Pro user friendly version of UNIX features a rewritten file structure for improved processing speed and incorporation of System III enhancements and the Berkeley Development Utilities, while retaining functional compatibility with Version 7. Fortune users have a wide range of proven system software products available, including a range of relational database managers, high level languages (C, ANSI, COBOL, C BASIC, SMC Business BASIC, Fortran 77, Pascal and APL), a range of conversion aids and emulators for several popular systems, financial modelling systems, and communications facilities (including asynchronous, batch bisynchronous, interactive bisynchronous 3274/5/6 emulation and networking). In applications, Datacraft has a comprehensive range of Australian packages running on the Fortune 32:16, including a full range of business accounting modules, production systems, share register and purchase management. The Fortune 32:16 hardware features a specially designed dual bus structure and memory management unit designed for high performance of UNIX software on Super Micro hardware, and can be configured as either a professional system with up to three workstations or an extended performance (XP) for up to ten users. Both ranges are compatible and an upgrade kit allows field upgrade from PS to XP. Configurability spans up to 1 Mbyte of main memory, 120 Mbytes of hard disc, cartridge tape back-up, IEEE 4888 interfaces and multiple printer interfaces. Each terminal can connect a printer.

Datacraft Office Systems Pty Ltd.
529 Burwood Road,
Hawthorn 3122

MICROCOMPUTER RANGE

The Sage range of high performance microcomputers can run the following operating systems and their wide range of application packages: IDRIS (a UNIX system), UCSD p-system, CPM/68K, BOS, MIRAGE (runs MICROAPL), PDOS, HYPERFORTH and MODULA 11. It is also possible to run IDRIS, CPM and other operating systems simultaneously. Two models are currently available, the Sage II and the Sage IV, both based on the Motorola 16 bit 68000 chip with a clock speed of 8 MHz and offering multi-user capabilities. The smaller Sage II, designed for single or two user situations, comes with either one or two built-in low profile floppy disc drives and can be configured with 128 Kbytes to ½ Mbyte of memory. Three variations are available in the Sage IV system, offering simultaneous multi-user facilities for up to six operators. The smallest Sage IV has a low profile 8000 Kbyte floppy disc drive and a matching 6 Mbyte fixed Winchester disc drive, the medium sized IV is equipped with a 640 Kbyte floppy and matching 40 Mbyte Winchester drive, and the largest Sage IV system can be configured with one or two 800 Kbyte floppy drives plus up to four Winchester drives in the same cabinet and offering up to 160 Mbytes of storage. System support, including regular seminars, is provided by Rakon Computers.

Rakon Computers.
114 Alexander Street,
Crows Nest 2065

UNIX SOFTWARE FOR VMS

Version 2.3 of Unity from ATAC is a complete implementation of the UNIX System which runs as a user process under VMS and uses only documented features of VMS, thus ensuring continued portability to future VMS releases. Release 2.3 of Unity features more than 100 utilities, a high degree of integration between the VMS Command Language DCL and UNIX, the ability to re-direct input and output of UNIX commands while working in DCL, and provision for up to 64 simultaneous users.

ATAC.
55 Lavender Street,
Milsons Point 2061

32-BIT Q-BUS UNIX SYSTEM

The MDB Micro/32, a new low-cost 32-bit computer system, features the MC68000 microprocessor and Regulus, a powerful operating system which accommodates all other UNIX software, and one that encourages future system expansion. Regulus features user source compatibility with UNIX V6, 7 and System III, and offers complete support of all UNIX kernel features, multi-key B-tree ISAM and VAX/PDP-11 cross support, and a host of operating systems and command functions not available in other UNIX systems. For example, Regulus supported software drivers allow use of Q-bus line printer controllers, disk and tape controllers, 8 or 16 channel multiplexers, synchronous serial interfaces and high speed DMA interface modules. The MDB Micro/32 contains a single quad size CPU board, designated the MDB-M32, which utilises the Motorola MC68000 processor with 32-bit data and address registers, coupled with dual MC68451 memory managers having 64 segments of associatively mapped memory and 512KB RAM parity memory. The MDB-M32 also has four asynchronous serial ports, two of which have modem control, and a parallel Centronics interface printer port. System memory can be expanded by use of dual size 512KB memory modules up to a total of 4 Mbytes. The 5¼ in high RETMA rack mountable Micro/32 includes a 22-bit addressing backplane/card guide assembly that accommodates 8 quad or 16 dual size modules. A switching power supply provides +5V at 36 amps and +12V at 5 A to the backplane, as well as all power to the disc drives. All I/O cable connections are made to the rear cable distribution.

MDB Systems (Australia).
PO Box 384,
Neutral Bay 2089

ELECTRONIC SPREADSHEET

The Unison spreadsheet system, Viewcomp, features easily specified worksheet dimensions (from 127 columns x 127 rows to 2 columns x 5460 rows), a write protect feature, an easy to use "what if" facility, a wide range of output formatting options, (such as parentheses for negative values, currency character, decimal point or comma, bar graphs made up of asterisks or plus signs, columns three to 60 characters wide, a joined format to stretch titles and data across multiple columns up to 255 characters, and left, right or centre adjustment within columns), up to four windows on the terminal, powerful editing facilities without changing modes, a variety of escape options to reduce typing, natural column and row addressing, file compatibility with the Sironix file system and the use of Berkeley termcap files for terminal independence.

Email Computer Systems.
PO Box 154,
Mooroolbark 3138

DIBOL COMPATIBLE PROGRAMME LANGUAGE

SIBOL is a commercial programming language developed by Software Ireland which is compatible with DEC's DIBOL (supported on the PDP-11 and VAX ranges of computers), and allows existing DIBOL programmes to run under UNIX Version 7, UNIX System III and UNIX lookalikes supporting the full range of system calls and providing the standard I/O library of C functions. The Sibol package consists of a compiler, a run-time interpreter, a symbolic debugger, a library of external utility subroutines. The entire package is written in the C language which is standard on all UNIX systems. The only visible difference between source programmes in SIBOL and DIBOL is that SIBOL expects file specifications in the UNIX format, which is different to those of CTS-300 and CTS-500. A standard ISAM file management system and a symbolic debugger are provided.

Z Systems Pty Ltd.
196b Vulture Street,
South Brisbane 410001

MULTI-USER TIME-SHARING SYSTEMS

The Zilog System 8000 models 21 and 31 are multi-user, time-sharing computer systems which feature: a 6 MHz Z8001A CPU which has 16 general purpose registers, an 8 Mbyte address space, and capability to perform 8, 16 and 32-bit operations; 6 MHz Z8010A memory management units; 6 MHz Z80B CPUs; 1 — 4 Mbytes of ECC controlled memory; a 32 Mbyte Winchester drive; an 84 Mbyte storage module disc drive; intelligent tape and disc controllers; a Z-Bus backplane interconnect; serial RS 232C and parallel interfaces for I/O devices; a modular design; no special cooling or power requirements; and options which include an industry-standard 9-track tape, support for a large number of users and a variety of peripherals.

ENHANCED OPERATING SYSTEM

Cromemco's new version of the 68000 CROMIX operating system (version 20.52) features support of more users on a Cromemco D-Series system with expanded shell buffers and process tables, a device driver that allows RAM memory to act like a disc drive and several other utilities. Up to four RAM disc drives in increments of 64 Kbytes can be allocated, up to the full 16 Mbyte RAM capacity allowed by the 68000 processor. A file structure can be built in the RAMDISC and then mounted and unmounted in the same manner as a physical disc drive and checksum error checking is provided for data integrity. The new CROMIX has 30 shell buffers and 30 process tables, (previous versions had a maximum of 10 shell buffers and 10 process tables) and allows up to eight users to use the system simultaneously with up to 16 terminals connected. New utilities include the Make utility for automating construction of executable programmes from separate modules, a revised version of Default (the flush utility for writing system I/O buffers to disc every specified number of seconds) and an enhanced version of the List utility. CROMIX version 20.52 for Cromemco's 68000-based multi-user system is available on 8 or 5¼ in diskettes.

Adaptive Electronics Pty Ltd.
418 St Kilda Road,
Melbourne 3004

The Zeus operating system, an enhanced version of UNIX with features from system III, featuring: a choice of languages (COBOL, BASIC, C, Z8000 assembler, PLZ/SYS, FORTRAN 77 and Pascal); a hierarchical file structure; compatible file, device and interprocess I/O; separate code and data address space; and user configurability; programming tools which include languages, libraries, a symbolic debugger, text processing software and more than 180 other utilities; and system utilities which include the command interpreter, data communications, and file maintenance, status enquiry and system accounting programmes. Other system features include easy future expansion and the ability to become part of a local area network.

Z-Systems Pty Ltd.
PO Box 59
Paddington 4064

ENHANCED UNIX

The SIRONIX operating system from Email Computer Systems is a multi-user interactive time sharing system which is based on UNIX V7 with Berkely enhancemennts and features: a hierarchical, tree structured file system that is addressable to 1 Gbyte, uses simple naming conventions, has file linking across directories, has automatic file space allocation/de-allocation, has flexible directory and file protection modes, and provides device independence; security via password protection in user log-on and file access; a powerful, easy to use command language which can be tailored to specific needs; a range of text processing and document preparation systems; interprocess communication via pipes; sophisticated desk calculator packages, full support of languages such as C, PASCAL, FORTRAN, COBOL, BASIC Plus and ASSEMBLER; a wide range of debugging aids such as symbolic debuggers; a source code control system to aid programme source control during software development; a network communication facility with other UNIX systems; automatic resource accounting for user processes; the capability for system activity to be monitored on-line to provide resource utilisation monitoring; the ability to execute sequential, asynchronous and background processes; graphics facilities; a computer assisted instruction facility for new users; provision of on-line documentation entries; and a terminal capabilities database which allows the inclusion of a new terminal without needing to change existing software.

Email Computer Systems.
PO Box 154,
Mooroolbark 3188

DATABASE MANGEMENT PACKAGE

Southdata's Superfile database management package features a multi-user architecture with record locking, two supporting packages (Superforms and Supertab), interfaces to high-level language upon request, the storage of data on disc in the form entered (no space padding or empty fields), a free-form structure (records can be different and access different logical files), a phonetic matching search, a wild card character search, range checking of numbers, the counting of records matching a criterion. The Superforms package allows the creation and alteration of screen forms, user defined data checking through forms and field calculations (such as multiplying two fields together or dividing a field by a constant), while Supertab permits the creation of reports on-screen (with automatic page numbering, lines containing left-justified, right justified and numeric fields from the database, separate defining of totals and sub-totals, the use of any field/s as sub-total triggers, etc), the creation of new fields by calculations between fields, the ability to select a group of records at print time, the simultaneous sorting of up to 36 fields and the creation of MailMerge files for use with WordStar. Superfile is available for 8-bit machines (in Z80 assembler for CP/M, TurboDOS CP-Net, Sig-Net, etc) and 16-bit machines (in C for MS-DOS, PC-DOS, CP/M-86, UNIX, IDRIS, etc, and is upward compatible with 8-bit databases). Superfile can use CP/M's full 8 Mbytes on 8-bit machines (extra databases can be opened under programme control) and is only limited to available disc space on 16-bit machines. A programme is also available to convert existing databases to Superfile format.

Sunshine State Scientific Systems.
16 Niddrie Drive,
Toowoomba 4350
Insert G786 on Information Feedback Card

UNIX COMPUTER SYSTEMS

MDB Micro/32 UNIX computer systems feature a quad size CPU module, a Motorola MC68000 processor with 32-bit data and address registers, Motorola MC68451 memory managers with 64 segments of associatively mapped memory, 512 Kbytes of memory with byte parity (expandible to 4 Mbytes), a 64 Kbyte user programmable EPROM capability, dual counter timers, date and time clock with battery backup, four asynchronous serial communications ports (programmable 110 - 19200 baud), a parallel Centronics printer port, DMA access to DEC LSI-11 Q-bus compatible backplane, a 10.4 Mbyte 5¼ in Winchester disc system (RL02 software transparent), a 1 Mbyte 8 in dual floppy diskette system (RXV21 software driver compatible and RX02 media compatible, a single dual size MDB controller for both drives, front panel indication of dc power on a CPU run/halt functions, a Q-bus backplane/cardguide assembly for up to eight quad or 16 dual size modules, a switching power supply providing disc drive power plus +5 V dc at 36 A and +12 V dc at 5 A to the backplane, a FCC compliant 5¼ in high rack mountable chassis and a variety of available modules (multiplexers, line printer controllers, disc and tape controllers etc). Configurations are also available with the 10.4 Mbyte RL02 Winchester replaced by a 20 Mbyte RL02 or RP02 Winchester. The Regulus operating system features user programme source compatibility with UNIX V6, 7 System III, support of al UNIX kernel features, a variety of software development tools, a real time scheduler, flexible inter-task communications, dynamic allocation of file index records. comprehensive file locking, multi-keyed B-tree ISAM and VAX/PDP 11 cross support.

MDB Systems (Aust) Pty Ltd.
200 Pacific Highway,
Crows Nest 2060

XENIX FOR NS16032

The Xenix operating system will soon be available for National Semiconductor's new NS16032 microprocessor. Xenix is Microsoft's licensed version of AT and T's Unix operating system specifically designed for the microcomputer market place to provide multi-user, multi-tasking capability. Xenix on the NS16032 is a super set of existing microprocessor versions of Xenix and will provide such features and virtual memory support. Xenix software developed for the NS16032 will also be available for the NS32032. Software developed for both chips will be upward and downward compatible. The complete NS16000 chip set is currently in production and is available.

Microsoft Pty Ltd.
PO Box 98,
Terrey Hills 2064

'C' LANGUAGE INTEGRATED BUSINESS SYSTEM

The Tetra plan "C" language integrated business accounting system from Tetra Office Systems consists of the following modules: Order Entry, Invoicing and Sales Analysis; Accounts Receivable; Accounts Payable; General Ledger and Management Reporting; Stock Control. Tetraplan provides a fully integrated accounting system with all the user's ledgers and data held 'on-line' for instant use. Tetraplan can automatically post information to other ledgers, avoiding duplication of data entry, and providing instant and detailed updating of transactions. Tetraplan is particularly suitable for use with powerful multi-user Unix based systems but can also be run on stand-alone systems with a 'C' compiler. By providing essential multi-user facilities such as record locking, Tetraplan easily copes with the most sophisticated multi-user environments. Tetraplan also demonstrates exceptional multi-user performance with instantaneous response at several screens. Tetraplan programmes are in 'C', structured in a logical and modular form, and because each module can run as a stand-alone system, the phasing problems associated with installations are substantially reduced. Many system variables such as the general ledger code structure, period dates, the number of decimal places to be printed in a report etc. can be changed without changing programmes. All Tetraplan programmes have been subjected to high volume testing in real working environments. Tetraplan features the facility to release a programme from the control of the screen, thus making the screen available for other work. The Tetraplan spooling system makes use of this facility, and all major report runs offer the option to the operator to run the programme in background. In practice, this means that even a single screen computer can run a number of jobs concurrently.

Tetraplan Office Systems.
PO Box 384,
Neutral Bay 2089

MULTIUSER PRODUCTIVITY SOFTWARE

The software packages iWORD (a word processor), iPLAN (an electronic spread sheet) and iMENU (a programme for developing screen menu-driven applications) are now separately available for Intel's Xenix based multiuser systems for OEMs (previously only available as a seamless set of integrated software on Intel's Database Information System, iDIS 735). Intel's iWORD is a menu-driven screen-oriented word processor that supports all standard text editing, storage and formatting functions. iWORD, an enhanced version of Horizon word processing, features definitions by a set of file drawers, command menus and an on-line help facility, English commands, the ability to visually format documents and print them as they appear on the display screen as well as format them using the text formatting facilities of Xenix, editing

and viewing two documents simultaneously, a spelling checker and correction capability, extensive on-line dictionary and a mail/merge feature. iPLAN Spreadsheet is a multipurpose tool capable of a range of applications including financial modelling, planning, forecasting and calculating engineering formulas, and also supporting what-if decision modelling through a two-dimensional matrix. iPLAN, a tailored version of Multiplan, provides a work space 63 columns x 255 rows, allows work sheets to be linked to receive or transmit data, provides up to eight windows for each work sheet, can be vertically or horizontally scrolled, permits different areas of large work sheets to be viewed concurrently, allows windows to be aligned, scrolled together, opened or closed, and has a cell-locking feature to protect work sheets from unauthorised access. Intel's iMENU is a hierarchical user interface and application development tool that ties together Xenix-based application software to create an integrated operating environment. iMENU, a version of Schmidt's/ menus, is written in C, and allows software developers to design seamless, logical interfaces to Xenix applications.

Intel Australia Pty Ltd.
Level 6,
200 Pacific Highway,
Crows Nest 2065

64 MBYTE MICRO

The Universe 68/67T, a computer from CRDS is available with a Fujitsu 8 in, 84 Mbyte Winchester drive with a formatted capacity of 64 Mbytes and an average access

time of 20 ms (40 ms max), and a 45 Mbyte streaming cartridge tape drive which reads and writes at 70 in/s. The Universe 68/67T has: a 12.5 MHz processor with 4 Kbyte cache; a full 32 bit data bus; up to 5 Mbytes of memory (16 Mbytes when new technology RAMs are implemented); the ability to accommodate slower Error Correction and Control Memories with little reduction in execution rate; a separate terminal I/O processor; high speed data transfers to peripherals with little loading on the main CPU, allowing the support of a number of terminals; the Unix System V (with Berkeley enhancements) or CRDS's Unos operating system; availability of a variety of software, including the Unify Data-Base Management System LEX-11 word processing, Supercomp-20 spreadsheet packages, and the Intelec range of application software for manufacturing and integrated accounting; optional hardware including an Ethernet networking interface (with driver software), a hardware floating point unit, a high speed graphics controller and an array processor; and a high speed Versabus (20 Mbytes) which can accommodate a wide range of modules suited for Versabus, Multibus, VMEbus, (DEC) Unibus and (Motorola) Micro-bus via bus adapters and or bus translators.

Intelec Data Systems.
217 Blackburn Road,
Mt Waverley 3149.

ENHANCED MULTI-USER UNIX FOR IBM PC/XT

International Data Services' enhanced multi-user implementation of Unix system III for the IBM-XT and IBM-PC features most of the popular Berkeley enchancements in supermicro Unix versions (such as Shell, More and VI), a PC-shell which emulates PC-DOS (allowing the running of PC-DOS programmes from the Unix environment, and supporting PC-DOS commands such as DIR, COPY, ERASE, DEC, REN) and the Unix utility UUCP (allowing an IBM PC to act as an intelligent Unix workstation with Unix to Unixhost communications, thus relieving the Unix mainframe). The enhanced Unix requires 5 Mbytes of disc space and minimum memory of 256 Kbytes, although more is recommended.

Atac Business Systems Pty Ltd.
55 Lavender Street,
Milson Point 2061.

Intel 310 and 380 supermicro systems are based on a choice of two powerful microprocessors; the 8086/8087 16-bit microprocessor set, or the powerful 80286/80287 set.

The iRMX 86 operating system is a modular real-time multitasking system which supports a variety of languages (including PL/M, FORTRAN and PASCAL) and support for the universal development interface. Also available is the XENIX 86 operating system, a complete multiuser interactive software base which is a derivative of Bell Laboratories' UNIX version 7 with a set of commercial enhancements.

Intel Australia Pty Ltd.
200 Pacific Highway,
Crows Nest 2065

ATAC GROUP OF COMPANIES

INC IN NSW
ATAC PTY LTD
ATAC WSA PTY LTD
ATAC FEE PTY LTD
ATAC LEASING PTY LTD
ATAC DATAPROCESSING PTY LTD
ATAC BUSINESS SYSTEMS PTY LTD
ATAC COMPUTER SERVICES PTY LTD

PB4222.PKU

21st March, 1984.

Mr. Peter Ivanov,
The Editor AUUGN,
C/- School of E.E. and C.S.,
University of New South Wales,
P.O. Box 1,
KENSINGTON. N.S.W. 2033.

Dear Sir,

Re: AUUGN Vol. 5 No. 1.

I was surprised to read what appeared to be a review of UNITY appearing in the Newsletter. The August 1983 meeting and its overview of UNIX in Adelaide contains the following.

"Adelaide Uni Computing Centre: VAX 780 with VMS Bad experience with EUNICE.....UNITY not much better....."

As far as I am aware Adelaide University has not tried UNITY although we offered them a 30 day trial.

We supplied the University with a user manual for UNITY under VMS however, it would not have been possible for comments such as "expensive to run, slow and files incompatible with VMS" to be derived from that documentation. Indeed, on the information we provided it is apparent that UNITY under VMS runs at 90% of the speed of native UNIX (stand alone UNITY) and that its file structure is compatable with VMS.

The University responded that whilst they were of the opinion that UNITY was a much more comprehensive and better engineered product than EUNICE they could not justify purchasing it when they already had paid for EUNICE.

SYDNEY OFFICE: 11TH FLOOR 55 LAVENDER STREET, MILSONS POINT N.S.W. 2061 AUSTRALIA. PHONE:02-436 2477.TELEX:AA23862.

Since that time a SYSTEM V based version has been announced with a significant price reduction for educational institutions already holding a UNIX licence and Adelaide University are understood to be reviewing their previous response.

It is unfortunate that the report of the August 83 meeting is, at best, ambiguous in respect of UNITY under VMS and, at worst, a misrepresentation of the evaluation process undertaken.

UNITY is provided with Fortran 77 as well as C and includes SCSS as well as may other items as standard which may make it considerably less expensive than other offerings.

Yours faithfully,
ATAC Software (N.S.W.)

Peter Braun,
Executive Director.

SYDNEY OFFICE: 11TH FLOOR 55 LAVENDER STREET, MILSONS POINT N.S.W. 2061 AUSTRALIA. PHONE:02-436 2477.TELEX:AA23862.

Netnews

I have reproduced below some of my network mail and a few "netnews" articles that I thought may be of interest to Australian UNIX users. I have deleted some of the less meaningful data generated by various mailers and news programs. No responsibility is taken for the accuracy (or lack thereof) of anything below.

I find collating all this stuff rather a pain and so would like some feed-back as to whether I should waste my time or not. Please write to me (electronically or not) if you have strong opinions either for or against inclusion of netnews.

From kre:munnari Tue May 15 16:49:18 1984
From: kre:munnari (Robert Elz)
To: netgurus:basser
Subject: US mail status

I had many replies to my query, all suggested the creation of a new newsgroup, that has been done, its called ´aus.netstatus´ and is intended to carry reports about the status of any of the important network links around the country.

Those of you not currently receiving news should probably think about starting to do so. Original (US) news software is available from me, pgn:moncskermit, chris:basser, or jenny:natmlab.

Michael Rourke at UNSW has rewritten the whole thing, you may prefer his version (smaller, less processes, etc) especially on a small machine. Enquiries to michaelr:elecvax.

nb: news can be used as an internal message distribution service (keeps /etc/motd smaller, and allows messages to be grouped by subject unlike the older ´msgs´ programs), or for nationwide, or international network news. You can arrange to receive any subset of the available news that you require, getting aus news will not tax anybody´s net links - there has been less traffic there than mail to netgurus. US news is another matter!

kre

From dave:csu60 Thu Apr 12 14:56:38 1984
To: auugn:elecvax
Subject: sched bug

Peter,

You might give this some prominence ...

There appears to be a bug in sched() which shows up on small machines i.e. non sep-id. Basically, where it looks for someone to swap out, it assumes that it has memory to burn, and can afford to ignore processi that are sleeping p_pri < PZERO. It has made the unwarranted assumption that a process that cannot be signalled also does not want to be swapped out, with the disastrous result that on a very busy machine, memory is eventually filled with sleeping procs that cannot be woken up until some swapped-out process can come in and do something, which it can't ... The result - a deadlock!

These are preliminary observations only, and I stand corrected if necessary, but I noticed the same bug when V7 sched() was implemented on V6. The system used to regularly deadlock when firing up getty's on a reboot, with everyone sleeping on PINOD except the getty that can unlock the inode, which in the meantime got itself swapped out!

The fix? Simple - just ignore the test on p_pri in the loop looking for someone to swap out. The system may thrash more as a result, but at least it won't freeze several times a day.

Perhaps there could be conditional compilation - using the priority test if running on a Vax/70/45/44/23+, but not on 60/40/34/23 etc.

Dave Horsfall (dave:csu60)

From: gwyn@Brl-Vld.ARPA
Newsgroups: net.unix-wizards
Subject: Bugs in System V math library
Date: Fri, 30-Mar-84 13:55:05 AEST

Gary Moss uncovered a family of bugs in the UNIX System V (Release 1) math library. The matherr() function was being passed the whole exception structure rather than a pointer to it, in the following files:

```
gamma.c line 57
j0.c    line 210
j1.c    lines 155 & 203
sin.c   line 48
```

To fix this, just put an ampersand & in front of "exc" in these calls to matherr().

From: geoffw@elecvax.SUN (Geoff Whale)
Date: Wed, 23 May 84 15:23:36 AEST
Newsgroups: net.bugs,net.bugs.usg,net.bugs.v7
Subject: fgrep
Organization: EE and CS, Uni of NSW, Sydney, Australia

The following bug can be found in many releases, notably Level 7
and system V (fgrep version 1.2).
Try matching the patterns "roar" and "arrow"
against the subject "arroar".
It will fail to match due to a problem in setting
fail links in cfail() - this was probably never noticed because
of the highly obscure and unstructured nature of the implementation,
which looks more like Fortran than C.
The expedient fix (and the poor quality of the original code)
is evident from the diff listing below:

```
317c317
<                *rear++ = s->nst;
---
>                *rear++ = s;
339,346c339,351
<                floop:  if (state == 0) state = w;
<                        if (state->inp == c) {
<                        qloop:  q->fail = state->nst;
<                                if ((state->nst)->out == 1) q->out = 1;
<                                if ((q = q->link) != 0) goto qloop;
<                        }
<                        else if ((state = state->link) != 0)
<                                goto floop;
---
>                        while (state != NULL && state->inp != c)
>                                if (state->link != NULL)
>                                        state = state->link;
>                                else    /* no more alternatives at this level */
>                                        state = state->fail;
>                        if (state == NULL)
>                                state = w;      /* first level node */
>                        else
>                                state = state->nst;     /* success link */
>                        do {
>                                q->fail = state;
>                                q->out |= state->out;
>                        } while ((q = q->link) != NULL);
```

The program has been completely rewritten at UNSW to use malloc()
to obtain tree nodes when needed (the original version declares
an incredible 96076 bytes of data!), to avoid useless leaf
nodes in the tree, and to accept the other grep options -s and -y.

-- Geoff Whale (geoffw:elecvax)

```
From: aef@shell.UUCP (Art Feather)
Date: Wed, 9-May-84 05:08:39 AEST
Newsgroups: net.unix-wizards
Subject: Re:  IBM and Univac Unix

UTS for "Amdahl-compatible" machines will be available as a "native"
system with the release of System V UTS.
It may not operate in ALL non-Amdahl systems due to the reduced error
handling that exists at present.  In IBM VM systems, the core
operating system is relied upon for most of the restarting, etc.
--
Art Feather [713-663-2335] {ihnp4,pur-ee,ut-sally,sequent,psuvax}!shell!aef

From: lee@unmvax.UUCP
Date: Mon, 30-Apr-84 18:30:31 AEST
Newsgroups: net.bugs.4bsd
Subject: 4.2 BSD occasionally trashes files

Index:  4.2BSD kernel
Description:
        Occasionally files will get trashed. This was reported
        earlier but the only "fix" was a stop-gap measure which controlled
        the bug by issuing a panic. These mods were made to iput() and
        closef().
Repeat-By: Hard to, it looks to be a race condition.
Fix:
        The routine irele() would lock an inode without first checking
        to see if such was already locked. Below is a diff of
        /sys/sys/ufs_inode.c. I suggest you leave in those panics
        for awhile just to make sure....
RCS file: RCS/ufs_inode.c,v
retrieving revision 1.2
diff  -r1.2 ufs_inode.c
2c2
<  * $Header: ufs_inode.c,v 1.2 84/02/12 22:47:45 root Exp $
---
>  * $Header: ufs_inode.c,v 1.3 84/04/30 02:12:19 root Exp $
3a4,11
>  * Revision 1.3  84/04/30  02:12:19  root
>  * Think I found the bug with trashed files that the stop gap measure
>  * (the panic in iput() and closef()) controlled. In irele() it locks
>  * the inode but does not check to see if it has already been locked.
>  * I am leaving in the stop-gap panics for now but we should know
>  * for sure in a month.
>  *                              --Lee
>  *
252c260
<                 ip->i_flag |= ILOCKED;
---
>                 ilock(ip);
--
                        --Lee (Ward)
                        {ucbvax,convex,gatech,pur-ee}!unmvax!lee
```

From: ado@elsie.UUCP
Date: Thu, 12-Apr-84 09:45:51 AEST
Newsgroups: net.lang.c,net.bugs.4bsd
Subject: i = i * f vs. i *= f

I compiled the following program with the 4.1bsd C compiler:

```
main()
{
        int     i;

        i = 100;
        i = i * .2;
        printf("%d0, i);
        i = 100;
        i *= .2;
        printf("%d0, i);
}
```

I got this output:

```
20
0
```

Is this a bug?
--
UUCP: decvax!harpo!seismo!rlgvax!cvl!elsie!ado
DDD: (301) 496-5688

From: crp@ccivax.UUCP (Chuck Privitera)
Date: Wed, 18-Apr-84 09:40:39 AEST
Newsgroups: net.bugs.4bsd
Subject: Re: Re: i = i * f vs. i *= f

The bug was reported (and fixed) back in January by randvax!edhall.
I have installed the fix here and have not had any problems since.
The fix follows:

Subject: integer op= floating evaluated incorrectly by C compiler
Newsgroups: net.bugs.4bsd,net.bugs.usg

This bug may affect all pre-5.0USG C compilers, and perhaps earlier 5.0USG compilers as well. The below fix only works for PCC-derived compilers (such as the BSD VAX C compiler).

Index: usr.lib/ccom 4.2BSD 4.1BSD 3.0USG

Description:

When assignement operators such as *= are used with an integer Left-Hand Side and a floating-point expression on the Right-Hand Side, results are incorrect. For example:

```
int i = 6;
i *= .5;
```

leaves a value of 0 in i, rather than 3. The +=, -=, and /= operators are similarly affected.

Caused by:
	Conversion of RHS of assignment to type of LHS before application
	of the operator.

Fixed by:
	The fix is in two parts. First, the automatic forcing of type
	conversion to the LHS of an assignment op must be shut off in
	appropriate circumstances. This requires a change to tymatch()
	in mip/trees.c:

```
1031c1031,1035
<       if( t != t2 || o==CAST ) p->in.right = makety( p->in.right, tu, 0, (int)tu );
---
>       if( o==CAST || (t != t2
>         && ( (dope[o]&(FLOFLG|ASGOPFLG)) != (FLOFLG|ASGOPFLG)
>         || t != INT || (t2 != DOUBLE && t2 != FLOAT) )) ) {
>               p->in.right = makety( p->in.right, tu, 0, (int)tu );
>               }
```

	This causes certain assignment ops (+=, -=, *=, /=, i.e. the ones
	appropriate in floating-point) to remain in the parse tree without
	`balanced' operand types. When these get to code-generation the
	compiler would break unless the template table had the proper pieces
	added to it. Thus, in pcc/table.c:

```
712a713,729
>
> /* begin new stuff */
>
> ASG OPFLOAT,     INAREG|FOREFF|FORCC,
>       SAREG|AWD,       TWORD|TCHAR|TSHORT,
>       SAREG|AWD,       TDOUBLE,
>               NAREG,   RLEFT|RESCC,
>               "        cvtZLd  AL,A1 OD2      AR,A1 cvtdZL  A1,AL0,
>
> ASG OPFLOAT,     INAREG|FOREFF|FORCC,
>       SAREG|AWD,       TWORD|TCHAR|TSHORT,
>       SAREG|AWD,       TFLOAT,
>               NAREG,   RLEFT|RESCC,
>               "        cvtZLf  AL,A1 OF2      AR,A1 cvtfZL  A1,AL0,
>
> /* end new stuff */
>
```

	-Ed Hall
	Rand Corporation
	Santa Monica, CA
	decvax!randvax!edhall	(UUCP)
	edhall@rand-unix	(ARPA)

From: usenix@ucbtopaz.CC.Berkeley.ARPA
Newsgroups: net.unix-wizards
Subject: submissions for USENIX tape wanted
Date: Mon, 2-Apr-84 19:14:22 AEST

One of the primary goals of the USENIX Association is to provide mechanisms for the sharing and distribution of software, especially licensed code, within the UNIX community. Because of the somewhat complicated licensing structure imposed by AT&T, it is difficult (and often illegal) for sites to distribute their newly developed code. USENIX, however, has mechanisms for verifying licenses and dealing with the problems inherent in distributing software.

USENIX is eagerly soliciting code or products suitable for distribution to its members. These include device drivers, additions to the C or system programming libraries, bug fixes, enhancements to the kernel or utilities, research projects available for testing, and new tools or packages implemented on UNIX. Either licensed or non-licensed code may be submitted. Software will remain the property of the submitter, and will be distributed with a right-to-use only agreement.

Contributed software will eventually be grouped by license restrictions and distributed to USENIX Institutional and Supporting Members. In addition, we are attempting to expand our distribution coverage through agreements with foreign UNIX user groups. Arrangements have already been made to have the 1984 distribution(s) redistributed in Europe through the European UNIX System User Group (EUUG). Thus contributed software has a potentially world-wide distribution.

If you would like to take advantage of this opportunity to share your creative talents with your fellow UNIX lovers around the world, your contributions may be netmailed to the USENIX Association at ucbvax!g:usenix or sent to the USENIX Office via regular mail (PO Box 7, El Cerrito, CA 94530). Either UNIX licensed or non-licensed software may be submitted. The deadline for submissions to the 1984.1 Distribution is June 30, 1984.

For further information, contact the USENIX Office or Deborah K. Scherrer, Tape Committee Chairman:
ucbvax!lbl-csam!scherrer

From: Jeff@decwrl.UUCP
Newsgroups: net.bugs.4bsd
Subject: 4.2 (and 4.1) Vax pcc generates bad code for real comparisons
Date: Tue, 3-Apr-84 05:29:28 AEST

Description:

When pcc (the C compiler) generates code to compare a float and a double, it first converts the float to a double but doesn't realize that this takes two registers, not one.

If more than one temporary register is in use while evaluating such a comparision, one of the register may be trashed.

This bug applies to 4.1BSD and probably all earlier Vax pcc versions.

Repeat-By:

compile this C program:

```
double  dd[] =  { 0.0, 2.0 };
float   ff[] =  { 0.0, 1.0 };
int i = 1;

main(){
        if( ff[i] >= dd[i] ) printf( "wrong0 );
}
```

If you run it, it will print "wrong" because the code that is generated (cf. cc -S) for the comparison is:

```
movl    _i,r0
movl    _i,r1
cvtfd   _ff[r0],r0      # trashes r1
cmpd    r0,_dd[r1]      # index is random
jlss    L19
```

Fix:

The fix given here may be suboptimal, but it seems to give correct code. (I can't remember where I got the fix from; it's not mine.) It tells pcc to use 2 stack longwords as a temporary instead of one register.

On 4.1BSD, the file is /usr/src/cmd/pcc/table.c; it's identical to the 4.2 file.

```
*** table.c.bad Wed Dec 15 13:25:17 1982
--- table.c     Mon Apr  2 16:05:08 1984
***************
*** 1,4
  static char *sccsid ="@(#)table.c     1.1 (Berkeley) 12/15/82";
  # include "mfile2"

  # define WPTR TPTRTO|TINT|TLONG|TFLOAT|TDOUBLE|TPOINT|TUNSIGNED|TULONG

--- 1,9 -----
  static char *sccsid ="@(#)table.c     1.1 (Berkeley) 12/15/82";
```

```
+ /*
+  * 2 April 1984       Jeff Mogul      Stanford
+  *    Re-installed fix to prevent compare of indexed float and
+  *    indexed double from clobbering the index register.
+  */
  # include "mfile2"

  # define WPTR TPTRTO|TINT|TLONG|TFLOAT|TDOUBLE|TPOINT|TUNSIGNED|TULONG
***************
*** 228,234
  OPLOG,          FORCC,
        SAREG|AWD,       TDOUBLE,
        SAREG|AWD,       TFLOAT,
!               NAREG|NASR,      RESCC,
                "        cvtfd   AR,A1    cmpd        AL,A1OP",

  OPLOG,          FORCC,

--- 233,239 -----
  OPLOG,          FORCC,
        SAREG|AWD,       TDOUBLE,
        SAREG|AWD,       TFLOAT,
!               2*NTEMP,         RESCC,
                "        cvtfd   AR,A1    cmpd        AL,A1OP",

  OPLOG,          FORCC,
***************
*** 234,240
  OPLOG,          FORCC,
        SAREG|AWD,       TFLOAT,
        SAREG|AWD,       TDOUBLE,
!               NAREG|NASL,      RESCC,
                "        cvtfd   AL,A1    cmpd        A1,AR0P",

  OPLOG,          FORCC,

--- 239,245 -----
  OPLOG,          FORCC,
        SAREG|AWD,       TFLOAT,
        SAREG|AWD,       TDOUBLE,
!               2*NTEMP,         RESCC,
                "        cvtfd   AL,A1    cmpd        A1,AR0P",

  OPLOG,          FORCC,
```

From: chris@basser.SUN
Newsgroups: net.bugs.4bsd
Subject: Re: 4.[12] Vax pcc generates bad code for real comparisons
Date: Fri, 6-Apr-84 14:48:25 AEST

Here is a better fix to pcc for the problem described by Jeff of DEC western research labs. The real code generation problem is that the result of the cvtfd instruction can't be shared with its 'float' source. Of course, the double intermediate value needs 2 AREGS.

In fact, the second table entry is unnecessary, as there is a rewrite rule which will convert the left operand to double (correctly) and then compare it. I leave it here mostly to keep the linenumbers right (and in case I overlooked something).

```
====== "table.c" line 228 - 238
    OPLOG,      FORCC,
            SAREG|AWD,  TDOUBLE,
            SAREG|AWD,  TFLOAT,
-                   NAREG|NASR, RESCC,
                    "   cvtfd   AR,A1   cmpd        AL,A1OP",

    OPLOG,      FORCC,
            SAREG|AWD,  TFLOAT,
            SAREG|AWD,  TDOUBLE,
-                   NAREG|NASL, RESCC,
                    "   cvtfd   AL,A1   cmpd        A1,AROP",
======
    OPLOG,      FORCC,
            SAREG|AWD,  TDOUBLE,
            SAREG|AWD,  TFLOAT,
+                   2*NAREG,    RESCC,
                    "   cvtfd   AR,A1   cmpd        AL,A1OP",

    OPLOG,      FORCC,
            SAREG|AWD,  TFLOAT,
            SAREG|AWD,  TDOUBLE,
+                   2*NAREG,    RESCC,
                    "   cvtfd   AL,A1   cmpd        A1,AROP",
======
```

From: kiessig@idi.UUCP
Newsgroups: net.news,net.net-people,net.mail,net.unix
Subject: UUCP Network Directory: Availability announcement
Date: Thu, 5-Apr-84 02:14:13 AEST

The first issue of the UUCP Network Directory will be coming off of the presses on about May 1, barring any unforeseen disasters. Since the first press run will be limited, those of you wishing to obtain copies should place your orders as soon as possible. The price is $10.95 if you are either listed in the directory or if you provide enough information to be listed in the next issue (the minimum required to be listed is your name and a valid UUCP address). If you don't want to be listed, or if you don't have access to UUCP, the price is $15.95. You must include your UUCP address with your order to qualify for the $5 discount. There is $1.25 shipping and handling, and sales tax in California. Subscriptions are available for $40/year if you're listed, $65 if you're not, including shipping and handling (plus tax in CA) - that's for 4 quarterly issues. We will not knowingly provide copies of the Directory to potential abusers.

The first issue will have over 800 people listed, and should be over 30 pages long. It will be typeset in 5.5 x 8.5 inch format. The entries consist of people's names & UUCP addresses and optionally their work phone, home address, home phone, and one line about them. We are hoping to provide some site and route information in future editions - the exact timing and content depends to some extent on the public domain mapping efforts now underway by Rob Kolstad et. al. (cbosgd!uucpmap).

Over the next two weeks or so, we will be mailing out "final confirmation" notes to everyone who is listed in the Directory. This is to give people a chance to correct any mistakes in their directory entry before publication. It also gives us a chance to make sure there is a valid mail route and mailbox for everyone listed, and lets everyone who is listed know for sure that they are going to be listed. People who send in new entries before April 10 are guaranteed to be included in the first edition. Send your entry to idi!uucpdir, and include as much of the information mentioned above as you want listed - or just type "r-" after this article and I can send you a template to fill in, if you'd rather.

Orders should be sent to the following address. No purchase orders, please (except by special arrangement).

Intelligent Decisions, Inc.
Directory Order
P.O. Box 50174
Palo Alto, CA 94303
408-996-2399

--
Rick Kiessig
{decvax, ucbvax}!sun!idi!kiessig
{akgua, allegra, amd70, cbosgd, harpo, ihnp4, ios, qubix}!idi!kiessig

From: laman@sdcsvax.UUCP
Date: Wed, 11-Apr-84 13:16:25 AEST
Newsgroups: net.bugs,net.bugs.usg
Subject: Bug in System V and System V.2 login.c

There is a bug in both System V and SystemV.2 login.c programs. There are two calls to strncmp with ONLY TWO arguments in terminal(). We changed ours to "strcmp". You can change yours by adding a sizeof for the third argument. Either way will suffice. A contextual diff follows.

```
*** login.c     Wed Apr 11 12:21:58 1984
--- oldlogin.c  Wed Apr 11 12:21:23 1984
***************
*** 563,570
                        continue;
                if(passl && db.d_ino != fsb.st_ino)
                        continue;
!               if (strcmp(&db.d_name[0],"syscon") == 0 ||
!                       strcmp(&db.d_name[0],"systty") == 0)
                        continue;
                (void) strcpy(rbuf, dev);
                (void) strcat(rbuf, db.d_name);

--- 563,570 -----
                        continue;
                if(passl && db.d_ino != fsb.st_ino)
                        continue;
!               if (strncmp(&db.d_name[0],"syscon") == 0 ||
!                       strncmp(&db.d_name[0],"systty") == 0)
                        continue;
                (void) strcpy(rbuf, dev);
                (void) strcat(rbuf, db.d_name);
```

Mike Laman
UUCP: {ucbvax,philabs,sdccsu3,sdcsla}!sdcsvax!laman

```
From: mccallum@nbires.UUCP
Date: Thu, 12-Apr-84 16:02:57 AEST
Newsgroups: net.unix-wizards,net.bugs.4bsd
Subject: Re: kernel bug in flock

Subject: flock panics kernel when given invalid parameter
Index:  sys/sys/kern_descrip.c 4.2BSD

Description:
        The flock system call can cause the 4.2 kernel to panic when
        given an invalid second parameter.  This occurs only when the
        file in question is already locked with LOCK_SH and a second
        call to flock where the second parameter does not contain any
        of (LOCK_UN|LOCK_EX|LOCK_SH) set.

Repeat-By:
        The problem can be shown with:
                ...
        flock(fd, LOCK_SH);
                ...
        flock(fd, 0);
                ...

Fix:
        The following context diff of kern_descrip.c prevents the panics:


*** kern_descrip.c      Mon Apr  9 08:16:14 1984
--- /sys/sys/kern_descrip.c     Wed Mar 28 14:35:47 1984
***************
*** 405,414
                u.u_error = EOPNOTSUPP;
                return;
        }
-       if ((uap->how & (LOCK_UN|LOCK_EX|LOCK_SH)) == 0){
-               u.u_error = EINVAL;                     /* ??? */
-               return;
-       }
        if (uap->how & LOCK_UN) {
                ino_unlock(fp, FSHLOCK|FEXLOCK);
                return;

--- 405,410 -----
                u.u_error = EOPNOTSUPP;
                return;
        }
        if (uap->how & LOCK_UN) {
                ino_unlock(fp, FSHLOCK|FEXLOCK);
                return;
----------
        Doug McCallum
        {ucbvax,allegra,amd70,hao}!nbires!mccallum
```

From: donn@sdchema.UUCP
Newsgroups: net.bugs.4bsd
Subject: Ghastly f77 bug in common subexpression elimination -- IMPORTANT
Date: Tue, 27-Mar-84 02:18:12 AEST

Index: usr.bin/f77/src/f77pass1/optcse.c 4.2BSD

Description:
F77 considers two variables from the same COMMON block to be the same variable for the purposes of common subexpression elimination. This is almost always wrong.

Repeat-By:
Copy the following program into a file com.f. Compile it with the optimizer turned on.

```
        program com

        common /x/ a, b, c, d
        integer result, a, b, c, d

        a = 2
        b = 3
        c = 4
        d = 5

        result = a * b + c * d

        print *, result

        stop
        end
```

The expected result of running the program is `26'. What the program actually prints is `12'. To see why, here is the code produced (comments and other prettification added):

```
        .globl  _MAIN_
        .set    LF1,4
_MAIN_:
        .word   LWM1
        subl2   $LF1,sp
        jmp     L12
L13:
        movl    $2,_x_
        movl    $3,_x_+4
        movl    $4,_x_+8
        movl    $5,_x_+12
        mull3   _x_+4,_x_,-4(fp)
        addl3   -4(fp),-4(fp),{result}  # Mistake!
        pushal  v.3
        calls   $1,_s_wsle
        pushl   $4
```

```
        pushab  {result}
        pushal  {1}
        pushal  {3}
        calls   $4,_do_lio
        calls   $0,_e_wsle
        pushl   $0
        pushal  {00,00}
        calls   $2,_s_stop
        ret
        .align  1
_com_:
        .word   LWM1
L12:
        moval   v.1,r11
        jmp     L13
```

The two multiplies were treated as common subexpressions, and the same temporary [-4(fp)] was used to store both results.

Fix:

I agonized over this one. Since this is a very urgent problem I have decided to can my earlier inconclusive changes and make a single, one-line change that has the effect of making COMMON variables pretty much ineligible for CSE (actually it gives them the status of arrays, which is convenient since COMMON blocks look just like arrays to the compiler). This is fast and has the advantage of introducing few new bugs. The change is to optcse.c in routine scantree():

```
***************
*** 564,570
                                ap = (Addrp) p->exprblock.leftp;
                                idp = findid(ap);
                                killdepnodes(idp);
!                               if( ! ap->isarray ) {
                                        if(rnode->is_dead)idp->assgnval=idp->init
                                        else idp->assgnval = rnode;
                                }

--- 584,590 -----
                                ap = (Addrp) p->exprblock.leftp;
                                idp = findid(ap);
                                killdepnodes(idp);
!                               if( ! (ap->isarray || ap->vstg == STGCOMMON) ) {
                                        if(rnode->is_dead)idp->assgnval=idp->init
                                        else idp->assgnval = rnode;
                                }
```

This bug is responsible for many seemingly unrelated errors, such as the previously reported 'exponentiation' error where 'x1 ** alpha + x2 ** alpha + x3 ** alpha' always turned out to be '3 * x1 ** alpha'. You should fix this as soon as you can...

Donn Seeley UCSD Chemistry Dept. ucbvax!sdcsvax!sdchema!donn
32 52' 30"N 117 14' 25"W (619) 452-4016 sdcsvax!sdchema!donn@nosc.ARPA

From: donn@sdchema.UUCP
Newsgroups: net.bugs.4bsd,net.lang.f77
Subject: Re: problem with mixed-mode in f77
Date: Fri, 6-Apr-84 20:27:44 AEST

Subject: Multiplying something by the constant 0 can cause f77 to crash
Index: usr.bin/f77/src/f77pass1/expr.c 4.2BSD

Description:
This bug is the same as the one reported by trq@astrovax.UUCP with the subject 'problem with mixed-mode in f77'. Basically it randomly applies to f77 programs that have an expression in which a value of a type other than INTEGER is multiplied by the constant 0. Admittedly few real programs explicitly multiply by zero -- the main sources of examples are programs which have PARAMETER variables that are set to zero.

Repeat-By:
Put the following program in a file named constbug.f:

```
        program constbug

        integer iwrblk

        iwrblk = 1 + 0 * 1.0
        print *, iwrblk

        stop
        end
```

If you compile this program with the optimizer on, the compiler says:

```
constbug.f:
   MAIN constbug:
Compiler error line 5 of constbug.f: Impossible type 0 in routine mkconv

compiler error.
```

If you compile the program with the optimizer off, it prints '0'. (What you would expect it to print is '1'.)

Fix:
What is happening is that f77 is stomping on its data structures. F77 wants to convert '0 * 1.0' to '0.0' when it evaluates constant expressions. The data structure for the '0' is freed after the conversion to '0.0', but the compiler doesn't pay attention to the new value and continues to use the old data structure for the '0' instead. Since the compiler is

free to reuse this space, it later seems to behave strangely (in this case the '0' shares with the optimizer data structure for the same statement it appears in!). This is trivial to fix -- the change is in function mkexpr() in expr.c:

```
------------------------------------------------------------------
RCS file: RCS/expr.c,v
retrieving revision 1.1
diff -c -r1.1 expr.c
*** /tmp/,RCSt1015888   Fri Apr  6 20:08:55 1984
--- expr.c      Fri Apr  6 19:10:53 1984
***************
*** 1601,1607
                        {
                        if(rp->constblock.const.ci == 0)
                                {
!                               mkconv(etype, rp);
                                goto retright;
                                }
                        if ((lp->tag == TEXPR) &&

--- 1601,1607 -----
                        {
                        if(rp->constblock.const.ci == 0)
                                {
!                               rp = mkconv(etype, rp);
                                goto retright;
                                }
                        if ((lp->tag == TEXPR) &&
------------------------------------------------------------------
```

Donn Seeley UCSD Chemistry Dept. ucbvax!sdcsvax!sdchema!donn
32 52' 30"N 117 14' 25"W (619) 452-4016 sdcsvax!sdchema!donn@nosc.ARPA

From: donn@sdchema.UUCP
Newsgroups: net.bugs.4bsd
Subject: Constant exponents sometimes fail in f77
Date: Mon, 2-Apr-84 13:05:13 AEST

Subject: Bug in copy propagation causes constant exponents to break in f77
Index: usr.bin/f77/src/f77passl/bb.c 4.2BSD

Description:
        An f77 program that uses certain constant integer exponents greater than 4 will not get the proper results if the optimizer is turned on. In general the values produced are much larger than they should be.

Repeat-By:
        Compile the following program with the optimizer turned on and run it:

```
------------------------------------------------------------------
        program powbug

        integer i
```

```
        i = 10
        i = i ** 5

        print *, i

        stop
        end
```

If your f77 is broken it will print `1000000` instead of `100000`.

Fix:

F77 handles constant integer exponentiation with inline multiplies. The way that inline multiplies are implemented requires two temporaries, one to hold the original value and one to hold a copy which gets repeatedly squared; if the exponent is not a power of two, then the first temporary is multiplied into the second temporary the necessary remaining number of times. The copy propagation code notices that the first temporary is initialized from the second; unfortunately it misses the fact that the second copy is modified and reduces the two temporaries to one, eliminating the assignment from the one temporary to the other. The reason it misses the modification of the second temporary is that the squarings are done with an OPSTAREQ operator instead of an ordinary OPASSIGN. The solution is to make the copy propagation code know about the other assignment operators. The changes are in routine ckexpr() in bb.c:

```
rcsdiff -c -r1.1 bb.c
*** /tmp/,RCSt1015453   Mon Apr  2 12:47:44 1984
--- bb.c        Fri Mar 30 05:02:44 1984
***************
*** 343,348

  {
  Tempp lp,rp;

  if (expr->exprblock.opcode == OPASSIGN)
        {

--- 343,349 -----

  {
  Tempp lp,rp;
+ int   oc = expr->exprblock.opcode;

  if (oc == OPASSIGN || oc == OPPLUSEQ || oc == OPSTAREQ)
        {
***************
*** 344,350
  {
  Tempp lp,rp;
```

```
! if (expr->exprblock.opcode == OPASSIGN)
        {
        lp = (Tempp) expr->exprblock.leftp;
        rp = (Tempp) expr->exprblock.rightp;

--- 345,351 -----
  Tempp lp,rp;
  int   oc = expr->exprblock.opcode;

! if (oc == OPASSIGN || oc == OPPLUSEQ || oc == OPSTAREQ)
        {
        lp = (Tempp) expr->exprblock.leftp;
        rp = (Tempp) expr->exprblock.rightp;
***************
*** 349,355
        lp = (Tempp) expr->exprblock.leftp;
        rp = (Tempp) expr->exprblock.rightp;
        if (lp->tag == TTEMP)
!               if (rp->tag == TTEMP)
                        enter (lp->memalloc, rp->memalloc);
                else
                        enter (lp->memalloc, ENULL);

--- 350,356 -----
        lp = (Tempp) expr->exprblock.leftp;
        rp = (Tempp) expr->exprblock.rightp;
        if (lp->tag == TTEMP)
!               if (rp->tag == TTEMP && oc == OPASSIGN)
                        enter (lp->memalloc, rp->memalloc);
                else
                        enter (lp->memalloc, ENULL);
```

Thanks should go to Peter Gross at the High Altitude Observatory for bringing this to my attention...

Donn Seeley UCSD Chemistry Dept. ucbvax!sdcsvax!sdchema!donn
32 52' 30"N 117 14' 25"W (619) 452-4016 sdcsvax!sdchema!donn@nosc.ARPA

From: donn@sdchema.UUCP
Date: Fri, 13-Apr-84 01:56:18 AEST
Newsgroups: net.bugs.4bsd
Subject: f77 disallows floating point constants less than -2.59e+33

The distributed f77 will not accept floating point constants that are smaller than roughly -2.59e33, but going by the representation one would expect to be able to use constants as low as -1.70e38. The following program fails in the way shown:

```
% cat tst.f
        program tst

        real    s
        data    s       /-1.7e38/
```

```
        print *, s

        stop
        end
% f77 tst.f
tst.f:
   MAIN tst:
Error on line 4 of tst.f: data value too large

Error.  No assembly.
```

Positive floating point numbers up to 1.70e38 ARE allowed, however. What's actually happened is that for some reason the minimum value is set incorrectly in conv.c. Here is the correction (notice that the floating point numbers are given as bit patterns):

```
*** /tmp/,RCSt1004227   Fri Apr 13 01:34:35 1984
--- conv.c      Fri Apr 13 01:08:19 1984
***************
*** 21,27
  LOCAL long dminint[]  = { 0x0000d000, 0xffff00ff };

  LOCAL long dmaxreal[] = { 0xffff7fff, 0xffff7fff };
! LOCAL long dminreal[] = { 0x0000f800, 0xffffffff };

  ^L


--- 34,40 -----
  LOCAL long dminint[]  = { 0x0000d000, 0xffff00ff };

  LOCAL long dmaxreal[] = { 0xffff7fff, 0xffff7fff };
! LOCAL long dminreal[] = { 0xffffffff, 0xffff7fff };

  ^L
```

In case you're curious, the reason why the least significant bits are not all ones is that an epsilon value was subtracted (added) in order to detect overflow in atof(), which indicates overflow by returning the maximum (minimum) floating point number. This has the side effect that you can't indicate the highest and lowest DOUBLE PRECISION numbers to the precision of the representation, but this is not likely to hurt anyone (famous last words).

Bob Corbett tells me that this fix was applied at Berkeley back on Dec. 13 of last year, so if you have a distribution more recent than that you should ignore this article. Thanks to Mike Brown at the National Solar Observatory (kpno!brown) for pointing this bug out to me.

Donn Seeley UCSD Chemistry Dept. ucbvax!sdcsvax!sdchema!donn

From: donn@sdchema.UUCP

Date: Sat, 7-Apr-84 21:35:07 AEST
Newsgroups: net.bugs.4bsd
Subject: Massive f77 fixes for subroutine argument temporary bugs

Here, as promised, are Bob Corbett's changes to f77 to prevent the allocation of subroutine argument temporaries from trashing DO loop limits and other things. I have included the fixes to Bob's changes which I have made, so if you are already running the changes from Berkeley and Bob hasn't sent you the latest round of fixes, you will want to use this code instead.

(these fixes were too large to publish, mail peteri:elecvax if you need them)

From: rcj@burl.UUCP (R. Curtis Jackson)
Date: Thu, 19-Apr-84 13:15:36 AEST
Newsgroups: net.bugs.usg
Subject: yacc and lex bugs
Organization: AT&T Technologies; Burlington, NC

FIRST OFF -- AN APOLOGY: I have been informed that the Unix Hotline folks processed my MR on yacc(1) promptly, and that after sitting in Murray Hill for a year now it is considered "Under Investigation" and the status is "We'll postpone judgement until a later date". The Hotline people did their job admirably, and I am sorry I blasted them without having the MR checked first.

1) yacc

a) Problem (history):
In the 'good old days' (V6), yacc would not tell you in its debug output that it had found 'token ADDOP'; it would tell you that it had found 'token 426'; it was up to you to find out (via using the -d option and looking at y.tab.h) what token 426 really was. So it was beneficial to define your own token numbers rather than letting yacc default them; that way they were in your source file for easy access. Even today, if you have one lexical analyzer feeding two or more parsers with the same tokens, you want to make sure that the token numbers are the same in both parsers, so this feature of yacc (being able to define your own token numbers) is still quite valid and useful.

b) Problem:
yacc uses tables of ints to transition from state to state, and it uses negative numbers based on the negative of the token number and on (-(the_next_desirable_state) - 1000). In other words, if you are to transition to state 53, the number in the table will be -1053. [I am about 90% sure this is accurate -- regardless I do know the problem is related to this]. If you use token numbers > 1000, then yacc will run perfectly, generate proper y.output if you use the -v option, but when y.tab.c is compiled and executed, the results are totally unpredictable. yacc will transition to wildly inappropriate states and start generating 'Syntax error's at a phenomenal rate.

c) Cure:
Let yacc default its token numbers unless you absolutely cannot

get around it. If you really need that feature, don't use token numbers over 1000. NOTE: remember to start your token numbers above the ascii code, or yacc will think that your ADDOP, to which you have assigned a token number of 040, is a space, and vice-versa. If you have to use token numbers *AND* you have so many tokens that you are running over 1000, then wade through the yacc code and find the define for that number and increase it. (An extremely improbable situation)

2) lex

a) Problem:
lex has an input character buffer called yysbuf that is dimensioned to YYLMAX, defined to be 200. Unfortunately, the routine that reads the input file [yylook()] does not, as far as I can tell, check to make sure that it has not gathered into yysbuf (or yytext, which is also dimensioned to YYLMAX) more than YYLMAX characters. If it is matching a pattern that is more than YYLMAX characters, it writes them right past the end of yysbuf and on into 'The Memory Zone', usually producing Memory Faults or Bus Errors somewhere down the line.

b) Cure:
If you get a Memory Fault or Bus Error, and cannot seem to locate it, put the following lines into the declarations section of your lex program:

```
%{
blah;
blah;
blah;
# undef YYLMAX
# define YYLMAX 5000 /* or some other ridiculously large number */
blah;
blah;
%}
```

This will override lex's YYLMAX define (see the lex(1) documentation concerning overriding lex's input() macro and also look at the first 15 lines of any lex.yy.c for details).
If your Memory Fault/Bus Error goes away, then either:

1) Your pattern specs for lex are out of line -- you are not matching what you think you are matching -- check for rules containing things like [^x], where x is some character. Remember that rules like these match ANY character but x, including newlines.

2) Your pattern specs are OK, but you are simply trying to match more than 200 characters. Use the above method to define YYLMAX to a reasonable number for your application and go on.

Hope this helps some people, please direct any questions/comments to me at the address below,
--
The MAD Programmer -- 919-228-3313 (Cornet 291)

From: johnd@denelcor.UUCP (John Donnelly)
Date: Fri, 27-Apr-84 12:19:16 AEST
Newsgroups: net.usenix
Subject: USENIX Support For Local User Groups
Organization: Denelcor, Aurora, CO

Local User Group Support Policy

The USENIX Association will support local user groups in the United States and Canada in the following ways:

- Assisting the formation of a local user group by doing an initial mailing for the group. This mailing may consist of a list supplied by the group, or may be derived from the USENIX membership list for the geographical area involved. At least one member of the organizing group must be a current member of the USENIX Association. Membership in the local group must be open to the public.

- ;login will publish information on local user groups. Information on local groups giving the name, address (phone number and/or net address), time and location of meetings, special events, etc. is welcome.

Please contact the USENIX office if you need assistance in either of the above matters. John Donnelly is the Board member to contact if you have comments, suggestions, etc. He may be reached at (303) 337-7900, ext 268, or hao!denelcor!johnd electronically.

From lepreau@utah-cs.ARPA Fri May 25 17:51:10 1984
From: lepreau@utah-cs.ARPA
Date: Fri, 25-May-84 17:51:10 AEST
Newsgroups: net.unix-wizards
Subject: Salt Lake USENIX Conference Schedule

From: Jay Lepreau <lepreau@utah-cs.ARPA>

Sorry about the separate postings, but our news feed has been down for a bit so I thought I better get it out via Arpanet unix-wizards, too.

Track A schedule is definite (we hope!), but Track B is a bit tentative. In particular, a session or two may still be added to Track B. STUG in Track B is definite. Order of data here is: Track A Usenix, Track B Usenix, Track B STUG.

Preliminary Technical Program Schedules
Summer 84 USENIX Conference, Salt Lake City
and
Software Tools Users' Group

Track A - Wednesday, June 13

Wed-1A 9:00 - 10:30 a.m. UNIX Directions

Opening Remarks
Conference Organizers and USENIX board

KEYNOTE ADDRESS: An Architecture History of the UNIX System
Stuart I. Feldman, Bell Communications Research

UNIX Standards: UNIX Meets Godzilla, or How I Learned to Love the Bomb?
Michael Tilson, Human Computing Resources Corporation

10:30 - 11:00 a.m. Coffee Break

Wed-2A 11:00 - 12:30 a.m. Mail and News

ACSNET - The Australian Alternative to UUCP
Piers Dick-Lauder & R.J. Kummerfeld (University of Sydney), Robert Elz (University of Melbourne)

Broadcasting of Netnews and Network Mail via Satellite
Lauren Weinstein

The Berkeley Internet Name Domain Server
Douglas B. Terry, Mark Painter, David W. Riggle, and Songnian Zhou, U.C. Berkeley

MMDF II: A Technical Review
Douglas P. Kingston III, Ballistic Research Laboratory

DRAGONMAIL: A Prototype Conversation-Based Mail System
Douglas E. Comer and Larry L. Peterson, Purdue University

12:30 - 2:00 p.m. LUNCH

Wed-3A 2:00 - 3:30 p.m. Networks and Distributed Processing

Converting the BBN TCP/IP to 4.2BSD
Robert Walsh and Robert Gurwitz, BBN Laboratories

Network Tasking in the LOCUS Distributed Unix System
David Butterfield and Gerald Popek, Locus Computing Corporation

Project Athena
James Gettys, Digital Equipment Corporation - MIT/Project Athena

TEMPO - A Network Time Controller for a Distributed Berkeley UNIX System
Riccardo Gusella and Stefano Zatti, U.C. Berkeley

3:30 - 4:00 p.m. Coffee Break

Wed-4A 4:00 - 5:30 p.m. Distributed File Systems

The Version 8 Network File System
Peter J. Weinberger, AT&T Bell Laboratories

Towards a Distributed File System
Walter F. Tichy and Zuwang Ruan, Purdue University

The Livermore Interactive Network Communication System
Alex Phillips, Lawrence Livermore National Labs

Panel on Distributed File Systems
Session speakers and others

Track A - Thursday, June 14

Thu-1A 9:00 - 10:30 a.m. Programming Languages

An Optimizing Portable C Compiler for the New CDC CYBER 180
Kok-Weng Lee and Mario D. Ruggiero, Human Computing Resources Corporation

A Simple Simulation Toolkit in "C"
Robert P. Warnock III and Bakul Shah, Fortune Systems Corporation

An Adaptable Object Code Optimizer for UNIX Systems
Bill Appelbe and Bob Querido,
U.C. San Diego & NCR Corporation

Using Modula-2 for System Programming with Unix
Michael L. Powell, Digital Equipment Corporation

The FP-Shell
Manton Matthews and Yogeesh Kamath, University of South Carolina

10:30 - 11:00 a.m. Coffee Break

Thu-2A 11:00 - 12:30 a.m. Programming Environments and Window Systems

SYSTANT: An Integrated Programming Environment for Modular C under UNIX
Stowe Boyd, AZREX, Inc.

LIPs: Knowledge Base Development System
Kiyoki Ohkubo, PANAFACOM Limited

WINDX - Windows for the UNIX Environment
Peter E. Collins, Ithaca Intersystems, Inc.

The Maryland Window System
Chris Torek and Mark Weiser, University of Maryland

A Text-Oriented Terminal Multiplexor for Blits
Rob Pike, AT&T Bell Laboratories

12:30 - 2:00 p.m. LUNCH

Thu-3A 2:00 - 3:30 p.m. Kernel I

A Multiprocessor UNIX System
Maurice J. Bach and Steven J. Buroff, AT&T Bell Laboratories

A Demand-paging Virtual Memory Manager for System V
Richard Miller, Human Computing Resources Corporation

Resource Controls, Privileges, and other MUSH
Robert Elz, University Of Melbourne

A Dynamic Bad-Block Forwarding Algorithm
Bakul Shah and Robert P. Warnock III (Fortune Systems Corporation)

An Expandable Object-Based UNIX Kernel
Erik Reeh Nielsen, NCR Systems Engineering Copenhagen

3:30 - 4:00 p.m. Coffee Break

Thu-4A 4:00 - 5:30 p.m. Kernel II

Processes as Files
Thomas J. Killian, AT&T Bell Laboratories

Memory Management Units and the UNIX Kernel
Clara S. Lai and Chris Peer Johnson, UniSoft Systems

Techniques for Debugging XENIX Device Drivers
Paresh K. Vaish and Jean Marie McNamara, Intel Corporation

User-Mode Development of Hardware and Kernel Software
Robert P. Warnock III, Fortune Systems Corporation

Track A - Friday, June 15

Fri-1A 9:00 - 10:30 a.m. Performance Analysis and Comparisons

Relating Benchmarks to Performance Projections, or
What Do You Do With 20 Pounds of Benchmark Data?
Gene Dronek, Aim Technology

UNIX System V and 4BSD Performance
Jeffrey P. Lankford, AT&T Bell Laboratories

Measuring and Improving the Performance of 4.2BSD
Sam Leffler (Lucasfilm, Ltd), Mike Karels, and M. Kirk Mckusick
(U.C. Berkeley)

10:30 - 11:00 a.m. Coffee Break

Fri-2A 11:00 - 12:30 a.m. Applications, Text Processing, Graphics

The UNIX System HELP Facility
Thomas W. Butler and Lisa A. Kennedy, AT&T Bell Laboratories

Adventures with Typesetter (Device) Independent Troff
Mark Kahrs and Lee Moore, University of Rochester

The Readers Workbench - A System for Computer Assisted Reading
Evan L. Ivie, Brigham Young University

Circuit Design Aids - CDA: A Printed Circuit Board Manufacturing System
Terry Slattery and Willie McCool, U.S. Naval Academy

A Transition Diagram Editor
Charles C. Mills (U.C. Berkeley) and Anthony I. Wasserman
(U.C. San Francisco)

12:30 - 2:00 p.m. LUNCH

Fri-3A 2:00 - 3:30 p.m. System Management and Porting

Optical Storage Management Under the Unix Operating System
Perry S. Kivolowitz, SUNY Stony Brook

Automatic Software Distribution
Andrew Koenig, AT&T Bell Laboratories

4bsd UNIX TCP/IP and VMS DECNET:
Experience in Negotiating a Peacful Coexistence
Van Jacobson, Craig Leres, Joseph Sventek, and Wayne Graves,
Lawrence Berkeley Laboratory

Experiences with a Large Mixed-Language System Running Under the
UNIX Operating System
Richard A. Becker, AT&T Bell Laboratories

The Dynamics of a Semi-Large Software Project with Specific Reference to a
UNIX System Port
Brian Pawlowski and Alan Filipski, Motorola, Inc.

3:30 - 4:00 p.m. Coffee Break

Fri-4A 4:00 - 5:00 p.m. Open Session

Random Talks
These talks will be scheduled first-come, first-served, with signup
beginning Wednesday morning. Ten minutes/speaker, any topic allowed.

Track B: Workshops, Projects, Panels

Wed-3B 2:00 - 3:30 p.m. Workshop: How to Teach UNIX
Bubbette McLeod, chair (Informatics General Corp.), Jay Hosler (User Training Corp.), Stan Kelly-Bootle (author of "The Devil's DP Dictionary"), Bob Nystrum (Momentum Compter Systems), and Jim Joyce (International Technical Seminars)

Thu-1B 9:00 - 10:30 a.m. Workshop: TCP/IP and Networking
Michael Muuss, chair (Ballistic Research Lab), and others

Fri-1B 9:00 - 10:30 a.m. UUCP Mapping Project

What is a Domain?
Mark R. Horton, AT&T Bell Laboratories

Proposal for a UUCP/Usenet Registry Host
Mark R. Horton, Karen Summers-Horton, and Berry Kercheval, UUCP Project

Map Gathering Description and Progress Report
Scott Bradner, Harvard University

A Reliable Mail Service for the UUCP Net: Implementation Status Report
Berry Kercheval, Zehntel, Inc.

Panel and Q&A on the UUCP Mapping Project
UUCP Project members

Fri-2B 11:00 - 12:30 a.m. 4.2BSD Panel and Q & A
Kirk Mckusick, moderator (U.C. Berkeley). Panelists include Mike Karels (U.C. Berkeley), Sam Leffler (Lucasfilms), Robert Elz (University of Melbourne), Bill Joy & Bill Shannon (Sun Microsystems).

We may be adding additional sessions in Track B

Software Tools User Group - Thursday, June 14

Thu-3B 2:00 - 3:30 p.m. STUG Session I

Avionics SimuLation Package: A large System Application in Ratfor
Dave Martin, Hughes Aircraft Company

An Update on the Software Tools Standards Effort
Bill Meine, Sun Microsystems

Ada? Yet Another VOS?
Neil Groundwater, Analytic Disciplines, Inc.

A LEX Tool for the VOS
Vern Paxson, Real Time Systems Group, Lawrence Berkeley Labs

3:30 - 4:00 p.m. Coffee Break

Thu-4B 4:00 - 5:30 p.m. STUG Session II

A Portable TOPS-20-like Command Parser
Nelson Beebe, Univ. of Utah, Dept. of Physics

Mine Planning Applications in Ratfor
Mike Norred, MINESoft

A Ratfor Implementation of KERMIT
Allan Cole, Univ. of Utah Computer Center

Future Directions for STUG (open discussion)
Dave Martin, moderator, Hughes Aircraft Company

From mike:food23 Tue May 29 13:30:14 1984
To: unswgurus:elecvax
Subject: Shakespear on 'C'

Found in the latest DECUS news ...

Use C, or not use C, that is the question:
Whether 'tis nobler in the mind to suffer
The flags and warnings of a rude compiler,
Or to take arms against a sea of errors,
And by debugging fix them? To code, to hack,
No more; and by a hack to say we end
The type-check and the thousand other checks
Pascal is heir to, 'tis a compilation
Devoutly to be wish'd. To code, to hack;
To hack! perchance to test: ay, there's the rub;
For in that hackers bliss what bugs may come,
When we have written out this awful code,
Must give us pause: there's the respect
That makes development of so long life.

4.2BSD Bug List from MT XINU:

GENERAL INFORMATION ON 4.2 BUGLISTS FROM MT XINU

MT XINU has completed the first round of summarizing known 4.2bsd bug reports. This is an ongoing effort; further summaries will follow.

The current summary has been derived from reports submitted to 4bsd-bugs@BERKELEY (not from reports submitted only to net.bugs.4bsd, for example). All reports on file at Berkeley as of 23 March 84 have been reviewed.

Bug lists now being distributed are essentially "raw". No judgment has been passed as to whether the submitted bug is real or not or whether it has been fixed. Only minimal editing has been done to produce a manageable list. Reports which are complaints (rather than bug reports) have been eliminated; obscenities and content-free flames have been eliminated; and duplicates have been combined. The resulting collection contains over 300 bugs.

Three versions of the list are now ready for distribution:

2-Liners:
: Two lines per bug, including a concise description, the affected module, the submittor. Approximately 42K bytes, it was distributed to net.sources on 24 April 84.

Abstracts:
: The 2-Liner + approximately a paragraph of description per bug. About 200K bytes, it will be distributed to net.sources in chunks of about 50K bytes around 1 May 84.

The Public Collection:
: All our material, except that all but the most inocuous of source material has been removed to meet AT&T license restrictions. Hundreds of K bytes, this will be distributed to net.sources in several 50K byte pieces during May 84.

Please note that local usenet size restrictions may prevent large files from being received and/or retransmitted. MT XINU will not dump this material on the net a second time; if your site has not received material of interest to you within a reasonable time, please send for a paper copy.

If you want a paper copy of the three lists, send mail to:

MT XINU
739 Allston Way
Berkeley CA 94710

attn: buglist

or electronic mail to:

ucbvax!mtxinu!buglist

--FOR SOURCE LICENSEES--

Holders of source licenses for 32V, System III or System V can obtain a tape containing all the material, including proposed source fixes where such were submitted. Once again, MT XINU has not evaluated, tested or passed judgment on proposed fixes; all we have done is organize the collection and eliminate obvious irrelevancies and duplications.
To receive a tape, send a check for $110 or a purchase order for $150 to cover MT XINU's costs to the address given above (California orders add sales tax). You MUST include with your order a copy of the cover page, the serial number sheet and the signature page of your AT&T source license agreement. You MUST specify shipment to the institution or entity named as the source license holder. Please give us a street address, not a post office box. Orders which do not meet these requirements will be returned unprocessed.

--IMPORTANT DISCLAIMERS--

Material in this announcement and the accompanying reports has been edited and organized by MT XINU as a service to the UNIX community on a non-profit, non-commercial basis. MT XINU MAKES NO WARRANTY, EXPRESSED OR IMPLIED, ABOUT THE ACCURACY, COMPLETENESS, OR FITNESS FOR USE FOR ANY PURPOSE OF ANY MATERIAL INCLUDED IN THESE REPORTS.

MT XINU welcomes comments in writing about the contents of these reports via uucp or US mail. MT XINU cannot, however, accept telephone calls or enter into telephone conversations about this material.

The following is a list of two-line descriptions of the bugs processed by Mt Xinu. The first line gives the offending program or source file and source directory (separated by --), who submitted the bug, when, and whether or not it contained a proposed fix. The second line is a VERY short description of the problem.

```
adb/runpcs.c--bin          rws@mit-bold (Robert W. Scheifler)   18 Nov 83 +FIX
        arguments to :r losing first character
ar.c--bin                          salkind@nyu (Lou Salkind)   10 Feb 84 +FIX
        usr/group ids overflowing
arff.c--etc                        salkind@nyu (Lou Salkind)   17 Nov 83 +FIX
        RT-11 files can't be read
as.c--bin                          cbosgd!mark (Mark Horton)   6 Jun 83
        large structs not handled by the assembler
atrun.c--usr.lib           hpda!hpdsd!hpdsa!mojo (Joe Moran)   14 Feb 84 +FIX
        multiple group permissions not handled
awk--bin                         sun!shannon (Bill Shannon)    9 Dec 83  +FIX
        doesn't allow replacing fields
c2/c21.c--lib                  root.Oregon-Grad@Rand-Relay     4 Nov 83
        optimizer type casting being missed for bit masking
catman--etc                 ucsfcgl!blia!eric (Eric Allman)    22 Feb 84 +FIX
        won't produce local manual with "1" argument
cc--bin                     Mike Braca <mjb%Brown@UDel-Relay>  27 Sep 83
        bit fields inconsistently treats unsigneds
cc--usr.lib                   edhall@randvax.ARPA (Ed Hall)    11 Jan 84 +FIX
        casting op= operations loses precision on floats
cc--bin       mazama!thor (Jeff Thorson) <mazama!thor@Shasta>  20 Feb 84
        relational operators on floating arrays get wrong results
cc--bin                      Preston Mullen <mullen@NRL-CSS>   12 May 83
        unsigned modulo arithmetic done wrong
changes.4-81--man                                 ogcvax!root   Jun 24 83 +FIX
        typos, etc.
compact--ucb                                      allegra!rdg   Jul 2 83  +FIX
        can't handle large files
compact--ucb                Mike Braca <mjb%Brown@UDel-Relay>  27 Sep 83 +FIX
        can't handle long filenames
compat--games                      salkind@nyu (Lou Salkind)   7 Dec 83
        not complete, environment passed wrong
config--etc                     watrose!arwhite (Alex White)   14 Dec 83 +FIX
        doesn't understand wildcarded unibus drivenames
cp--bin                     Mike Braca <mjb%Brown@UDel-Relay>  27 Sep 83 +FIX
        doesn't close files on error, fills up file table
cpp/cpp.c--lib    Spencer W. Thomas  <UTAH-GR.thomas@utah-cs>   27 Apr 83 +FIX
        misses some lines in line numbering
creat--man                    Jay Lepreau  <lepreau@utah-cs>   25 Apr 83 +FIX
        stated usefullness for locking a lie
cribbage--games                 smith@wisc-rsch (Jim Smith)    7 Dec 83
        flushes are not counted
csh--bin                     leblanc@ucbdali (Emile LeBlanc)   15 Mar 84
        history with - arguments hangs
csh--bin                          edjames@ucbcory (Ed James)   5 Oct 83
        if-then interactive doesn't do anything
csh--bin                                ralph (Ralph Campbell)   23 May 83
        piped/background sleep doesn't run in background
csh--bin                                ralph (Ralph Campbell)   25 May 83
        quietly quits on unreadable ".."
csh--bin                                csuf!dav@trw-unix.UUCP   19 May 83
        switch doesn't allow fall-throughs, requires "breaksw"
csh/sh.glob.c--bin       Christopher A Kent <cak@arthur.ARPA>   17 Oct 83 +FIX
        glob expansion fails on non-standard histchars
ctags--ucb                    steveg@ucbic (Steve Greenberg)   18 Oct 83
        meta-characters are not escaped
```

ctype.h(3)--man sjk@SRIJOYCE (Scott J. Kramer) 16 Jun 83 +FIX
isprint inconsistency
curses/cr_tty.c--usr.lib cbosgd!mark (Mark Horton) 19 Jul 83 +FIX
"tspace" too small for complex terminals
dbx--ucb ucbvax!decwrl!goldberg Jun 8 83
definitions of variables in blocks are lost
dbx--ucb ucbvax!decwrl!goldberg Jun 13 83
doesn't allow input of double precision numbers
dbx--ucb ucbvax!decwrl!goldberg Jun 8 83
external definitions not recognised
dbx--ucb ucbvax!decwrl!goldberg Jun 17 83
missed breakpoint
dbx--ucb ucbvax!decwrl!goldberg Jun 4 83
should mark binaries busy
dbx--ucb ucbvax!decwrl!goldberg Jun 27 83
thrashing apon exiting program with large data space
dbx/object.c--ucb sam@ucbmonet (Sam Leffler) 22 Oct 83 +FIX
blows up with enumerated types
dd.c--bin hpda!hpdsd!hpdsa!eric (Eric B. Wertz) 23 Mar 84 +FIX
using same filename not noticed
df.c--bin Jeff Mogul <mogul@coyote> 1 Feb 84 +FIX
offline disk kills df
dict/words--misc arnold@UCBINGRES 28 Apr 83 +FIX
misspelled words
diff/diffdir.c--bin gray@ucbarpa (Bob Gray) 27 Jan 84 +FIX
doesn't diff ".*" files
dmesg.c--etc sdcsvax!sdchema!donn 29 Jan 84 +FIX
the - flag does nothing without msgbuf file
dump--etc genji@UCBTOPAZ.CC (Genji Schmeder) 13 Oct 83
rmtopen return value inconsistent
dump--etc dlw@ucbopal.CC (David L Wasley) 2 Mar 84 +FIX
truncates the inode bitmap
dump/dump.h--etc genji@UCBTOPAZ.CC (Genji Schmeder) 9 Oct 83 +FIX
return codes inconsistent
dump/dumpitime.c--etc sun!shannon (Bill Shannon) 12 Sep 83 +FIX
message about level leaves out number
dump/dumpoptr.c--etc allegra!rdg Jul 2 83 +FIX
mishandles filesystem names prefixing filesystems
dump/dumptape.c--etc Dave Johnson <ddj%Brown@UDel-Relay> 11 Oct 83 +FIX
tape length poorly estimated
dumpmain.c--etc root.Oregon-Grad@Rand-Relay 17 Aug 83 +FIX
number of tapes poorly estimated
efl--usr.bin Vincent Broman <broman%BUGS@Nosc> 16 Feb 84 +FIX
using its own calloc, which doesn't work with stdio
error--ucb mazama!stew (Stewart Levin) 19 Jan 84
-t suffix touch list sometimes ignored
explain--bin eggert@ucsbcsl.UUCP 13 Jan 83 +FIX
creates unneeded temp file, possibly in RO directory
f77--usr.bin allegra!astrovax!gam (Gary Mamon) 19 Mar 84
exponentiation inside sum list done wrong
f77--usr.bin jerry@ucbopal.CC (Jerry Berkman) 29 Feb 84
exponentiation problem
f77--usr.bin jerry@ucbopal.CC (Jerry Berkman) 7 Dec 83
include file declarations cause unexplained compiler error
f77--usr.bin allegra!astrovax!trq (Thomas R. Quinn) 18 Mar 84
parameter values cause wrong expression evaluation

```
f77--usr.bin                    jerry@ucbopal.CC (Jerry Berkman)   28 Feb 84
        statment functions and arrays conflict
f77--usr.bin                          leres@ucbarpa (Craig Leres)   6 Nov 83
        unexplained compiler error when using optimizer
f77/src/f2--usr.bin         ucsfcgl!ucsfcgl!gregc (Greg Couch)   3 Nov 83
        unexplained optimizer infinite loop
f77/src/f77passl/exec.c--usr.bin      sdcsvax!sdchema!donn ()   23 Nov 83 +FIX
        DO loop parameters lost
f77/src/f77passl/regalloc.c--usr.bin  sdcsvax!sdchema!donn ()   23 Nov 83 +FIX
        computed GOTO causes core dump
f77/src/f77passl/regalloc.c--usr.bin     sdcsvax!sdchema!donn   23 Nov 83 +FIX
        optimizer adding redundant code
f77/src/f77passl/regalloc.c--usr.bin     sdcsvax!sdchema!donn   27 Nov 83 +FIX
        registered reals not allowed
fed--ucb                       hpda!hpdsd!edmund (Ed Trujillo)   7 Mar 84
        bad system call/core dump
find.c--usr.bin                 sjk@sri-spam (Scott J. Kramer)   5 Dec 83  +FIX
        doesn't recognize sockets
fp/fpMain.l--ucb                  baden@ucbmonet (Scott Baden)   9 Oct 83  +FIX
        incompatible system calls causing core dumps
ftp--ucb                                Mike Muuss <mike@brl-vgr>   8 Sep 83
        connections using wrong address
ftp--ucb                                                jbn@FORD-WDL1   26 May 83
        passive open doesn't follow consistent protocol
ftp--ucb   Chris Torek <chris%umcp-cs.csnet@csnet-relay.arpa>   2 Mar 84
        remote cwd doesn't work
ftp/cmds.c--ucb                       Jeff Mogul <mogul@coyote>   20 Mar 84 +FIX
        globbed files not handled
ftp/getpass.c--ucb         rws@mit-bold (Robert W. Scheifler)   31 Jan 84 +FIX
        password truncating to eight characters
ftpd.c--etc              Christopher A Kent <cak@Purdue.ARPA>   13 Jan 84 +FIX
        last logins log not updated in some cases
ftpd/ftpd.c--etc                      bloom@ucbcory (Jim Bloom)   20 Sep 83 +FIX
        null passwords security breach
getgroups.2--man            Mike Braca <mjb%Brown@UDel-Relay>   27 Sep 83 +FIX
        inconsistency
getgroups.2--man                lepreau@utah-cs (Jay Lepreau)   27 Oct 83 +FIX
        inconsistency
getitimer.2--man                   Chris Kent <kent@BERKELEY>   24 Jul 83 +FIX
        update needed
getrlimit.2--man            Mike Braca <mjb%Brown@UDel-Relay>   3 Oct 83  +FIX
        inconsistency
gettable.c--etc           jsq@ut-sally.ARPA (John Quarterman)   12 Feb 84 +FIX
        updating host table via version number
gettable.c--etc                       salkind@nyu (Lou Salkind)   17 Nov 83 +FIX
        using nicname instead of hostname
groups.2--man                                           ralph   23 Mar 83 +FIX
        reference to setgroup
h/un.h--sys                 spgggm@ucbopal.CC (Greg Minshall)   31 Jan 84 +FIX
        bind passing wrong-sized structure
ifconfig.c--etc                              sun!pugs (Tom Lyon)   1 Nov 83  +FIX
        arp flag, flags trashed after hostname
ifconfig.c--etc                 lepreau@utah-cs (Jay Lepreau)   5 Nov 83  +FIX
        arp flag, flags trashed after hostname, update
init.c--etc                                 allegra!astrovax!wls   Jun 25 83 +FIX
        race condition for HUP signal and simultaneous logout
```

iostat.c--usr.bin mazama!stew (Stewart Levin) 22 Mar 84 +FIX
doesn't see external/internal clock speed differences
last.c--ucb Jay Lepreau <lepreau@utah-cs> 25 Nov 83 +FIX
remote last doesn't flush message apon quit
lastcomm.c--ucb sun!Jskud 29 Nov 83 +FIX
uses whole blocks, can miss recent commands
ld--bin decvax!ubc-vision!sfucmpt!kurn (Andrew Kurn) 28 Feb 84 +FIX
load map doesn't include files that only define storage
learn--bin wjcheng@ucbernie (Wunjei J. Cheng) 18 Dec 83
can't specify user directory or subject
learn/copy.c--usr.bin ihnp4!cmcl2!rna!dan 18 Feb 84 +FIX
^D causes infinite loop / Init not executable
lex--usr.bin mazama!stew (Stewart Levin) 2 Feb 84 +FIX
manual gives wrong meaning for . inside []
lib.b--usr.lib Mike Braca <mjb%Brown@UDel-Relay> 27 Sep 83 +FIX
scripts invoking (by non-specifying) the wrong shell
libI77/err.c--usr.lib dlw@ucbopal.CC (David L. Wasley) 28 Oct 83 +FIX
fortran programs won't dump core
libI77/ioinit.f--usr.lib sdcsvax!sdchema!donn 23 Feb 84 +FIX
uses a run-time library not available to it
libI77/sfe.c--usr.lib dlw@ucbopal.CC (David L. Wasley) 30 Oct 83 +FIX
x format returns error on short input lines
lib[FU]77/signal_.c--usr.lib quarles@ucbic (Tom Quarles) 13 Oct 83 +FIX
fortran signal not compatible with 4.2
libc/gen/alarm.c--lib sun!shannon (Bill Shannon) 21 Mar 84 +FIX
alarm can truncate pending alarms
libc/gen/crypt.c--lib cooper (Eric Cooper) 9 Oct 83 +FIX
calling DES crypt routines doesn't work
libc/gen/ctime.c--lib solomon@wisc-crys (Marvin Solomon) 4 Jan 84 +FIX
ctime(0) produces garbage
libc/gen/getwd.c--lib Mike Braca <mjb%Brown@UDel-Relay> 27 Sep 83 +FIX
getwd doesn't follow symbolic links
libc/gen/popen.c--lib dlw@ucbmonet (David Wasley) 12 Aug 83 +FIX
closes good file descriptors
libc/gen/scandir.c--lib Jay Lepreau <lepreau@utah-cs> 29 Nov 83 +FIX
repeatedly (needlessly) calls realloc
libc/gen/syslog.c--lib Marshall Rose <mrose@uci-750a> 18 Jan 84 +FIX
format string with % escapes doesn't work
libc/stdio/fopen.c--lib ralph (Ralph Campbell) 25 May 83 +FIX
full file table handled wrong
libc/stdio/fputs.c--lib Mike Braca <mjb%Brown@UDel-Relay> 27 Sep 83 +FIX
given zero length string returns garbage
libc/vax/gen/bcopy.s--lib lepreau@utah-cs (Jay Lepreau) 7 Sep 83 +FIX
comment in source file wrong/misleading
libdbm/Makefile--usr.lib sjk@sri-spam (Scott J. Kramer) 11 Nov 83 +FIX
doesn't use -c option to compile libdbm.a
libplot--usr.lib sun!shannon (Bill Shannon) 5 Sep 83 +FIX
library terminal names inconsistent with filters
lint--usr.bin allegra!rdg Jul 4 83
(-h) doesn't detect constant assignments in conditional
lint--usr.bin ellis @ tektronix Jun 20 83 +FIX
command-line options mishandled
lint/llib-lc--usr.bin ucsfcgl!blia!eric (Eric Allman) 9 Feb 84 +FIX
longjmp declared wrong in library
lisp--ucb jbn@FORD-WDL1.ARPA 1 Mar 84
in maclisp there are two DEFCONSTs defined differently

```
lisp/Makefile--ucb              lepreau@utah-cs (Jay Lepreau)   14 Nov 83 +FIX
        doesn't pass on MFLAGS
lisp/franz--ucb                                jbn@FORD-WDL1.ARPA   6 Mar 84
        mis-handles atom names starting with digits
lisp/franz/lam7.c--ucb        pur-ee!malcolm (Malcolm Slaney)   18 Jan 84 +FIX
        doesn't close pipes for child processes
lisp/franz/vax--ucb                salkind@nyu (Lou Salkind)   9 Dec 83  +FIX
        rawhlist won't compile
lock.c--ucb                   hpda!hpdsd!edmund (Ed Trujillo)   7 Mar 84  +FIX
        typing ^D puts it into infinite loop
login.c--bin                                   mark@cbosgd.UUCP   6 Jan 83  +FIX
        argument to ioctl is wrong type
lpr--usr.lib                       sun!shannon (Bill Shannon)   19 Dec 83
        host names not preserved between gateways
lpr/printjob.c--usr.lib              dagobah!efo (Eben Ostby)   17 Nov 83 +FIX
        information fields too small, cause lpd coredump
ls.c--bin                     dlw@ucbopal.CC (David L. Wasley)   22 Nov 83 +FIX
        speed-up of user name searching
ls.c--bin                                          sun!Jskud   21 Nov 83 +FIX
        symbolic links and -F and -l not combined well
mail--ucb                                      mayo@UCBCALDER   7 Aug 83
        replies change address `user@system' to `system:user'
mail.c--bin                                       cbosgd!mark   Jun 4 83  +FIX
        race condition in writing mail file
mail/cmd3.c--ucb    smoot@ut-sally.ARPA (Smoot Carl-Mitchell)   27 Jan 84 +FIX
        replies figured out wrong
make--bin                          quarles@ucbic (Tom Quarles)   19 Mar 84 +FIX
        loses file descriptors for large programs
makefile--sys                                     allegra!rdg   Jul 19 83
        #ifdefs missed when "depend" is makes dependency list
makefile--sys        Chris Kent <decwrl!kent%Shasta@SU-Score>   18 Jul 83
        won't correctly build bootrl
man.c--ucb          smoot@ut-sally.ARPA (Smoot Carl-Mitchell)   15 Dec 83 +FIX
        improved searching of local man directories
man.c--ucb                                     clyde@ut-ngp.ARPA   2 Feb 84  +FIX
        mishandling of job control (leaving phantom processes)
man.c--ucb                    dlw@ucbopal.CC (David L. Wasley)   1 Feb 84  +FIX
        not using cat file for non-ttys / messy (-h) nroffing
mh--local                                        cak (Chris Kent)   12 Aug 83
        loses mail on full mailbox
mh/cmds/prompter.c--new     rws@mit-bold (Robert W. Scheifler)   21 Nov 83 +FIX
        doesn't prompt for To, Cc, and Subject
mh/cmds/replsubs.c--new         sjk@sri-spam (Scott J. Kramer)   15 Dec 83 +FIX
        puts spaces around `@'
more.c--ucb                   dlw@ucbopal.CC (David L. Wasley)   31 Jan 84 +FIX
        glitches on terminals with funny "standout mode"
msgs.c--ucb                          sam@ucbarpa (Sam Leffler)   2 Sep 83
        msgs -p can be killed by more
mt.c--bin                   genji@UCBTOPAZ.CC (Genji Schmeder)   30 Sep 83 +FIX
        given no operation, it writes a tapemark
net/if.c--sys                Mike Braca <mjb%Brown@UDel-Relay>   27 Sep 83 +FIX
        in ubareset network drivers called with wrong # of args
net/raw_usrreq.c--sys       rws@mit-bold (Robert W. Scheifler)   21 Mar 84 +FIX
        when freeing a route, doesn't zero pointer
net/route.c--sys            rws@mit-bold (Robert W. Scheifler)   22 Nov 83
        redirects are treated as the wrong type
```

netinet/if_ether.c--sys Christopher A Kent <cak@Purdue.ARPA> 8 Jan 84 +FIX
can't force local ethernet traffic to wire
netinet/if_ether.c--sys salkind@nyu (Lou Salkind) 2 Dec 83 +FIX
packet length wrong
netinet/in_pcb.c--sys watmath!arwhite (Alex White) 17 Feb 84
freeing a free socket
netinet/ip_icmp.c--sys Jeff Mogul <mogul@coyote> 27 Feb 84 +FIX
fragmented packet causes dangling pointer
netinet/ip_icmp.c--sys rws@mit-bold (Robert W. Scheifler) 1 Dec 83 +FIX
look for headers in multiple mbufs (fragmented packet, see sys/109)
netinet/ip_icmp.c--sys Michael John Muuss <mike@brl-vgr> 14 Dec 83 +FIX
ping packets lost - wrong format ID field
netinet/ip_output.c--sys Bill Croft <croft%Safe@SU-Score> 27 Oct 83 +FIX
not working on manual route
netinet/ip_output.c,net/route.c--sys Paul Kirton 5 Dec 83 +FIX
changes in route table not noticed
netinet/raw_ip.c--sys lwa@MIT-CSR 30 Nov 83 +FIX
large packets get rejected for raw internet
netinet/tcp_input.c--sys rws@mit-bold (Robert W. Scheifler) 7 Nov 83 +FIX
code change - not a bug
netinet/tcp_input.c--sys spgggm@ucbopal.CC (Greg Minshall) 9 Feb 84 +FIX
connections hang in FIN_WAIT_2 upon disconnect
netinet/tcp_output.c--sys rws@mit-bold (Robert W. Scheifler) 7 Nov 83 +FIX
code change - not a bug
netinet/tcp_subr.c--sys Christopher A Kent <cak@PURDUE.ARPA> 2 Dec 83 +FIX
maximum segment default set wrong
netinet/tcp_{input,output,subr}.c--sys Christopher A Kent 21 Mar 84 +FIX
max seg size calculations wrong
netinet/tcp_{input,output,subr}.c--sys gilligan@sri-spam 7 Dec 83 +FIX
max seg size calculations wrong
netinet/udp_usrreq.c--sys rws@mit-bold (Robert W. Scheifler) 22 Nov 83 +FIX
UDP checksums not used/don't work
netser/misc/rexecd.c--ucb root.Oregon-Grad@Rand-Relay 17 Aug 83 +FIX
TERM not set in passed environment
netser/rwho/{ruptime.c,rwho.c,rwhod.c}--ucb ogcvax!root Sep 8 83
integers not in network byte order
netser/rwho/{ruptime.c,rwho.c}--ucb ogcvax!root.tektronix Sep 19 83
handling argv, passes 0 string pointer to strcmp
nroff/n1.c,n4.c--usr.bin smoot@ut-sally.ARPA 9 Jan 84 +FIX
additions/changes for mm macro compatibility
pascal/pi--ucb emory!km (Ken Mandelberg) 3 Feb 84
real reads create "bad address" error from pix
pascal/utilities/config.c--ucb ogcvax!root.tektronix Oct 7 83
linked to non-existent file
pascal/utilities/pc.c--ucb raphael@wisc-crys 14 Feb 84
request for debugging feature
print.sh--ucb cbosgd!mark (Mark Horton) 1 Jul 83
in large case, lpr tries to create filenames with non-ascii characters
prof.c--usr.bin ralph (Ralph Campbell) 25 Apr 83 +FIX
ignores the -z option when generating plot
pstat.c,ps.c--etc watmath!arwhite (Alex White) 17 Feb 84 +FIX
-k option doesn't work with address translation
rcp--ucb mayo@UCBCALDER 10 Aug 83
`protocol screwup' message appears using globbed files
rcp.c--ucb rws@mit-bold (Robert W. Scheifler) 3 Jan 84 +FIX
3rd party copies don't follow expected equivalence

```
rcp.c--ucb                          mlgray@ucbcory (Mary L. Gray)   11 Feb 84 +FIX
        creates file in / when user doesn't have permissions
rcs/{ci.c,rcsgen.c}--new          lepreau@utah-cs (Jay Lepreau)   19 Oct 83 +FIX
        trying to ci multiple files fails on second log message
reboot.8--man                        cbosgd!mark (Mark Horton)   19 Jun 83
        no mention of front panel switch settings on 750s
refer--usr.bin                     mls@wisc-crys (Michael Scott)   17 Jan 84 +FIX
        footnotes sorted wrong on duplicate entries of a reference
refer--usr.bin                       mazama!stew (Stewart Levin)   11 Feb 84 +FIX
        interpolated signals make non-ascii suffix characters
refer--usr.bin                     mls@wisc-crys (Michael Scott)   17 Jan 84 +FIX
        sorting by author ignores %Q field
refer/addbib.c--usr.bin              salkind@nyu (Lou Salkind)   17 Nov 83 +FIX
        ^D after abstract causes infinite loop
renice--man                      ouster@ucbkim (John Ousterhout)   11 Sep 83 +FIX
        arguments list switched
restor--etc                                  bukys@Rochester.ARPA   5 Jul 83  +FIX
        if tape has inconsistend file length, restore can't read it
restor/restor.c--etc           ogcvax!root.tektronix@Rand-Relay   Oct 7 83
        using t and s option causes file extraction
restore--etc                      dlw@ucbopal.CC (David L Wasley)   1 Mar 84  +FIX
        incorrectly infers number of inodes
restore/restore.c--etc       mckusick@ucbmonet (Kirk Mckusick)   22 Mar 84 +FIX
        multi-reel sometimes loses list of files
restore/tape.c--etc            mckusick@ucbarpa (Kirk Mckusick)   7 Jan 84  +FIX
        restarting using R option causes core dump
rlogind.c--etc                    watrose!srradia (sanjay Radia)   24 Nov 83 +FIX
        wrong byte order conversionroutine used
rogue--games   mazama!paul (Paul Fowler) <mazama!paul@Shasta>   22 Mar 84
        various problems drop user into shell
ruptime.c--ucb                 genji@ucbopal.CC (Genji Schmeder)   14 Nov 83
        code for working directory is misleading
rwhod/rwhod.c--etc          jsq@ut-sally.ARPA (John Quarterman)   6 Dec 83  +FIX
        can't use dashes in host-name
rwhod/rwhod.c--etc                   salkind@nyu (Lou Salkind)   6 Mar 84  +FIX
        datagram packets not sent over point-to-point line
sa.c--etc                             munnari@kre (Robert Elz)   14 Oct 83
        -u option produces rubbish, as does AFORKed commands
sa.c--etc                             dagobah!efo (Eben Ostby)   7 Dec 83
        truncating times to seconds loses valuable information
script.c--ucb                        mazama!stew (Stewart Levin)   15 Feb 84
        (pseudo tty driver?) not handling ^S/^Q handshake
script.c--ucb                  dlw@ucbopal.CC (David L. Wasley)   22 Nov 83 +FIX
        doesn't support parity, RETURNS not stripped
see--ucb              ogcvax!root@teklabs.UUCP (Bruce Jerrick)   27 Apr 83 +FIX
        not invoking correct shell
select,getitimer,setitimer--man      cbosgd!mark (Mark Horton)   24 Aug 83 +FIX
        should reference each other
sendbug--ucb                         mazama!stew (Stewart Levin)   9 Feb 84
        no `clean' entry in makefile
sendbug/bugfiler.c--ucb         sjk@sri-spam (Scott J. Kramer)   7 Dec 83
        won't accept 3 arguments
sendbug/sendbug.sh--ucb                hpda!hpdsd!hpdsa!eric   22 Mar 84 +FIX
        doesn't interrogate editor shell variable
sendmail--usr.lib                       root@BERKELEY (Fluke )   5 Dec 83  +FIX
        concurrent alias database updating produces garbage
```

```
sendmail--usr.lib                    mazama!stew (Stewart Levin)   11 Jan 84
        duplicate copies of source make changes difficult
sendmail--usr.lib                      Yoon Kim <kaist!yoonkim>    13 Feb 84
        segmentation fault apon using option
sendmail/aux/syslog.c--usr.lib            decvax!dartlib!steve     30 Nov 83
        time mark only generated with -d option
sendmail/src/collect.c--usr.lib      salkind@nyu (Lou Salkind)    22 Nov 83 +FIX
        EOF before newline causes infinite loop
sendmail/src/collect.c--usr.lib                    Bill Nowicki   12 Mar 84 +FIX
        returns message each time TCP times-out
sendmail/src/conf.c--usr.lib  rhc@ucbopal.CC ( l'Innommable )     20 Mar 84 +FIX
        frozen config file values QueueLA and RefuseLA ignored
sendmail/src/parseaddr.c--usr.lib          pur-ee!Physics:crl     24 Feb 84
        some debug output is always produced
sendmail/src/srvrsmtp.c--usr.lib                Larry Peterson    12 Dec 83 +FIX
        alias references filing for same machine
sendmail/src/srvrsmtp.c--usr.lib           pur-ee!Physics:crl     24 Feb 84 +FIX
        variable "WizWord" #ifdef'd out
sendmail/src/{deliver,srvrsmtp}.c--usr.lib       root@BERKELEY    5 Dec 83  +FIX
        dropping mail from Berknet
setreuid--man                 Spencer Thomas  <thomas@utah-cs>    22 Jan 84 +FIX
        who can change ID of process stated wrong
sh--bin                                                jdb@sl-c   12 Mar 84
        setuid root shell scripts can give root shell to anyone
sigvec.2--man                    lepreau@utah-cs (Jay Lepreau)    7 Nov 83
        parameter description missing
socket.2--man                      Chris Kent <kent@BERKELEY>     10 Jun 83
        missing documentation
stand/boot.c?--sys                 Chris Kent <kent@BERKELEY>     15 Jun 83 +FIX
        boot doesn't follow symbolic links
stand/hp.c--sys                    mazama!thor (Jeff Thorson)     10 Mar 84 +FIX
        can't boot from eagle drive (on mba{123})
struct--usr.bin                   mazama!stew (Stewart Levin)     12 Jan 84 +FIX
        doesn't work under csh
su.1--man                        jeff@BERKELEY (Jeff Stearns)     10 Dec 83
        options not mentioned in manual
swapon.c--etc           decvax!dartlib!steve (Steve Campbell)     28 Nov 83 +FIX
        error message calls printf wrong
sys--sys                          Doug Kingston <dpk@brl-vgr>     29 Sep 83
        kernel doesn't clear exclusive open locks
sys--sys                           Jeff Mogul <mogul@navajo>      13 Dec 83
        mysterious file disappearance from iput()
sys--sys                                          mayo@ucbrenoir   2 Feb 84
        signal calls mixed with ioctl calls cause process hang
sys/init_main.c--sys        Mike Braca <mjb%Brown@UDel-Relay>     3 Oct 83  +FIX
        stack size hard limit intiallized to wrong value
sys/kern_acct.c,etc/sa.c--sys       watrose!root (Alex White)     2 Dec 83  +FIX
        acct & sa don't use same time units
sys/kern_clock.c--sys                            trw-unix!gorlick  3 Jun 83  +FIX
        "getrusage" doesn't average clock ticks
sys/kern_clock.c--sys              Chris Kent <kent@BERKELEY>     9 Jun 83
        undefined symbol "IUR" for VAX730 config
sys/kern_descrip.c--sys  decvax!uthub!thomson (Brian Thomson)     13 Feb 84 +FIX
        closing closed files references unused inodes
sys/kern_descrip.c--sys             salkind@nyu (Lou Salkind)     27 Jan 84 +FIX
        function fcntl returns wrong error number
```

sys/kern_exit.c--sys rws@mit-bold (Robert W. Scheifler) 17 Nov 83
wait not checking for zero rusage pointer, not sending EFAULT
sys/kern_resource.c--sys Mike Braca <mjb%Brown@UDel-Relay> 27 Sep 83 +FIX
"unlimit" returns error
sys/kern_sig.c--sys sun!shannon (Bill Shannon) 5 Sep 83 +FIX
"kill -0" returns wrong error code if uid is wrong
sys/kern_sig.c--sys Mike Muuss <mike@brl-vgr> 13 Jan 84 +FIX
signal handling changes, 4.1-4.2, changed back
sys/kern_time.c--sys salkind@nyu (Lou Salkind) 10 Mar 84 +FIX
time zone is ignored in settimeofday
sys/pty.c--sys Spencer W. Thomas <thomas@utah-cs> 26 Jul 83 +FIX
long writes to PTY controller can lose characters
sys/socket--sys Spencer W. Thomas <thomas%UTAH-GR@utah-cs> 16 Aug 83
write to pipe with bad buffer returns wrong error code
sys/sys_generic.c--sys rws@mit-bold (Robert W. Scheifler) 25 Feb 84 +FIX
stopping program waiting in select messes up select
sys/sys_xxx.c--sys cbosgd!mark (Mark Horton) 29 Jul 83 +FIX
bad accounting suspension condition calculation
sys/tty.c--sys Mike Braca <mjb%Brown@UDel-Relay> 27 Sep 83 +FIX
^S and ^Q not working in TANDEM mode
sys/tty.c--sys Michael John Muuss <mike@brl-vgr> 15 Dec 83 +FIX
sign bit extension affects bits in TIOCLSET call
sys/tty.c,tty_subr.c,vaxuba/dz.c--sys koda@hobgoblin 22 Feb 84 +FIX
3Com interface causing timing problems
sys/tty_pty.c--sys Web Dove <dove@sylvester> 21 Feb 84
not expanding timing characters for remote links
sys/tty_tb.c--sys dagobah!bill (Bill Reeves) 13 Sep 83 +FIX
tablets keep "inuse" state after being closed
sys/ufs_alloc.c--sys decvax!jmcg (Jim McGinness) 6 Feb 84 +FIX
cylinder group allocation bug causes hung system
sys/ufs_nami.c?--sys Chris Kent <kent@BERKELEY> 28 Jun 83
creat calls in uucp are failing
sys/ufs_syscalls.c--sys watmath!arwhite (Alex White) 8 Feb 84 +FIX
copen() call for FTRUNC doesn't check permissions
sys/ufs_syscalls.c--sys mazama!stew (Stewart Levin) 10 Jan 84 +FIX
lseek returns wrong value, EINVAL
sys/uipc_socket.c--sys sdcsvax!sdccsu3!madden@Nosc 7 Nov 83 +FIX
infinite loop by killing socket with pending condition
sys/uipc_socket.c--sys Mike Braca <mjb%Brown@UDel-Relay> 27 Sep 83 +FIX
large writes to pipe causes panic
sys/uipc_socket2.c--sys genji@UCBTOPAZ.CC (Genji Schmeder) 14 Oct 83
large network buffer causing "sbflush 2" panic
sys/uipc_usrreq.c--sys ralph (Ralph Campbell) 12 Sep 83 +FIX
multiple file descriptors to "message" handled wrong
sys/uipc_usrreq.c--sys watmath!arwhite (Alex White) 23 Jan 84
panic in mfree on recieving message with MSG_OOB set
sys/uipc_usrreq.c--sys watmath!arwhite (Alex White) 20 Feb 84 +FIX
running out of mbufs causes panic
sys/uipc_usrreq.c--sys spgggm@ucbopal.CC (Greg Minshall) 31 Jan 84
using connect() for data grams gives error
sys/various.c--sys kre@ucbmonet (Robert Elz) 21 Oct 83 +FIX
fix for inode counts getting < 0
sys/vm_mem.c--sys rws@mit-bold (Robert W. Scheifler) 7 Nov 83 +FIX
large partitions get block number sign-extended
sys/vmpage.c--sys ralph Jun 12 83 +FIX
raw dma write from user area can hang system

```
sys/vmsched.c--sys        Spencer W. Thomas  <thomas@utah-cs>   4 Aug 83  +FIX
        vmtotal() computes memory usage wrong
sys_inode.c--sys                      dagobah!efo (Eben Ostby)   17 Nov 83
        processes waiting for "flock" will hang if any is killed
syslog.c--lib            Christopher A Kent <cak@Purdue.ARPA>   11 Jan 84 +FIX
        syslog in C library not complete for datagrams
talk--local                           char@ucbkim (Bruce Char)   8 Sep 83
        illegal instruction if callee doesn't respond
talk.1--man                    jimbo@ucbmonet (Jim Kleckner)   17 Oct 83 +FIX
        typo
tar.c--bin                   ucbtopaz:genji (Genji Schmeder)   Sep 20 83 +FIX
        may ignore fact that directory is unreadable
tcp--sys                                      jbn@FORD-WDL1.ARPA   27 Jan 84
        retransmits too often - not conforming to its specs
telnet.c--etc              rws@mit-bold (Robert W. Scheifler)   17 Jan 84 +FIX
        "Are You There?" response to pty instead of network
telnet.c--ucb                                        lwa@MIT-CSR   23 Nov 83 +FIX
        not producing LF with CR
telnetd.c--etc               Bill Croft <croft%Safe@SU-Score>   16 Nov 83 +FIX
        dropping bytes, putting tty in RAW, writes to cloded files
termcap--etc                  Christopher A Kent <cak@purdue>   28 Mar 83 +FIX
        Adds entry not using scroll mode
termcap--etc                          cbosgd!mark (Mark Horton)   21 Sep 83 +FIX
        error in cs entry for vt100
termcap--etc                       mazama!stew (Stewart Levin)   12 Jan 84 +FIX
        gigi terminal not supported
tftpd.c--etc                        sun!shannon (Bill Shannon)   7 Mar 84
        doesn't check permissions on path to file
tip.c--usr.bin            ucsfcgl!ucsfcgl!tef (Thomas Ferrin)   3 Oct 83  +FIX
        doesn't detect loss of carrier
tip/cmds.c--usr.bin       ucsfcgl!ucsfcgl!tef (Thomas Ferrin)   3 Oct 83
        FRAMSIZE > BUFSIZ causes core dump ~taking large file
tip/uucplock.c--usr.bin               lbl-csam!astrovax!matt   9 Jan 84
        can't unlock its own files after change in UID
tip/{hunt,log}.c--usr.bin          salkind@nyu (Lou Salkind)   17 Nov 83 +FIX
        hangs if lin can't be openned, misplaced #ifdef's
tmac/tmac.me--usr.lib                              Craig Stanfill   26 Jul 83 +FIX
        interference between .(x, .(z and .sh
tmac/tmac.ms--usr.bin          sjk@SRIJOYCE (Scott J. Kramer)   7 Jun 83  +FIX
        .DS macro not working
tp--bin      mazama!stew (Stewart Levin) <mazama!stew@Shasta>   19 Feb 84 +FIX
        bus error from reallocated array
tset/tset.c--ucb                lepreau@utah-cs (Jay Lepreau)   18 Nov 83 +FIX
        not using terminal type from rlogin due to mapping
utimes.2--man                                 teklabs!ogcvax!root   Jun 14 83 +FIX
        wrong include file listed
uucp/cico.c--usr.bin      ucsfcgl!ucsfcgl!tef (Thomas Ferrin)   3 Oct 83  +FIX
        uses old signal handling procedures
uucp/conn.c--usr.bin                  dagobah!efo (Eben Ostby)   22 Nov 83 +FIX
        if write fails, a loop of failing writes is created
uucp/conn.c--usr.bin                salkind@nyu (Lou Salkind)   22 Nov 83 +FIX
        not able to transmit break for more than 1 second
uucp/uuencode.c--usr.bin     dlw@ucbopal.CC (David L. Wasley)   22 Nov 83 +FIX
        mode of file from pipe is encoded as zero
vax/autoconf.c--sys             decvax!mcvax!jim (Jim McKie)   14 Feb 84 +FIX
        uses the number of physical UBAs instead of configuration
```

vax/flp.c--sys watrose!arwhite (Alex White) 14 Dec 83 +FIX
out-of-range sector request leaves busy bit set
vax/locore.s--sys rws@mit-bold (Robert W. Scheifler) 7 Nov 83 +FIX
magic number in assembly code calculated wrong
vax/machdep.c--sys astrovax!wls (William L. Sebok) 5 Mar 84 +FIX
a tbuf parity error causes unnecessary panic
vax/machdep.c--sys astrovax!wls (William L. Sebok) 6 Mar 84 +FIX
hard memory errors not detected/corrected
vax/machdep.c--sys salkind@nyu (Lou Salkind) 1 Dec 83 +FIX
system memory size not computed correctly
vax/trap.c--sys rws@mit-bold (Robert W. Scheifler) 17 Nov 83 +FIX
interrupting wait3() with sleeping parent hangs
vax/trap.c--sys rws@mit-bold (Robert W. Scheifler) 3 Jan 84
sigstack() is using wrong stack
vax/tu.c--sys Bill Croft <croft%Safe@SU-Score> 12 Oct 83
tu_restart() is calling timeout() with bad argument
vaxif/if_ec.c--sys Jeff Mogul <mogul@coyote> 13 Feb 84 +FIX
loopback in 3Com ethernet driver doesn't work
vaxif/if_en.c--sys Jeff Mogul <mogul@coyote> 27 Feb 84 +FIX
byte-swapping done wrong on 3MB xerox ethernet driver
vaxif/if_il.c--sys Jay Lepreau <lepreau@utah-cs> 13 May 83 +FIX
after ubareset, packets get a "bad type field"
vaxif/if_uba.c--sys salkind@nyu (Lou Salkind) 13 Mar 84 +FIX
2K packets won't get through network
vaxif/if_vv.c--sys rws@mit-bold (Robert W. Scheifler) 22 Nov 83 +FIX
V2LNI driver doesn't broadcast or loopback correctly
vaxmba/hp.c--sys Doug Kingston <dpk@brl-vgr> 29 Sep 83
RM05's are improperly auto-config'd
vaxmba/hp.c--sys Mike Muuss <mike@brl-vgr> 23 Aug 83 +FIX
error in CDC 9775 partition table
vaxmba/ht.c--sys dlw@UCBTOPAZ.CC (David L. Wasley) 6 Aug 83 +FIX
"UNS"afe flag being inadvertently set
vaxmba/mt.c--sys mazama!stew (Stewart Levin) 6 Jan 84
TU78 driver can't do reverse reads / EOT happens early
vaxmba/mt.c--sys decvax!dartvax!steve (Steve Campbell) 22 Feb 84
can't use two TM78 tape formatters
vaxuba/d[hz].c--sys jbc <ut-ngp!jbc> 15 Jun 83 +FIX
IOCCDTR ioctl call not checking permissions
vaxuba/dh.c--sys salkind@nyu (Lou Salkind) 27 Jan 84 +FIX
soft carrier flag and interrupt inadvertently clears carrier
vaxuba/dh.c--sys astrovax!wls (William L. Sebok) 5 Mar 84 +FIX
soft carrier flag disables DTR
vaxuba/dmf.c--sys lbl-csam!astrovax!matt (Matt Crawford) 2 Feb 84
dmfprobe() doesn't check for actual device
vaxuba/dmf.c--sys lbl-csam!astrovax!matt (Matt Crawford) 17 Jan 84
undefined variables compiling with DMFDMA option
vaxuba/dz.c--sys koda@hobgoblin (Jim Koda) 22 Feb 84 +FIX
dztimer lowers IPL causing panic
vaxuba/uba.c--sys Mike Braca <mjb%Brown@UDel-Relay> 27 Sep 83 +FIX
UBA reset causes panic
vaxuba/uda.c--sys lbl-csam!astrovax!matt (Matt Crawford) 1 Dec 83 +FIX
"set controller char" fails if disk unit 0 not present
vi--ucb ucsfcgl!blia!eric (Eric Allman) 16 Feb 84
.exrc "version" command gives duplicate version messages
vi--ucb hamachi@ucbkim (Gordon Hamachi) 19 Dec 83
:f command doesn't update file permissions

```
vi--ucb                raphael@wisc-crys.arpa (Raphael Finkel)   22 Mar 84
        doesn't understand system error 62 (symbolic link cycle)
vi--ucb                                     leres (Craig Leres)   8 Dec 83
        interrupt message getting into text
vi--ucb                                                ttang@Uci   Nov 1 83
        numbered commands (eg 100^B) can cause core dump
vi/ex_vops2.c--ucb   sdcsvax!sdccsu3!madden@Nosc (Jim Madden)   2 Nov 83  +FIX
        put mishandles wrapmargin
vmstat.c--ucb                       mazama!stew (Stewart Levin)   22 Mar 84 +FIX
        doesn't see external/internal clock speed differences
vmstat.c--ucb                        cbosgd!mark (Mark Horton)   2 Jun 83  +FIX
        timer calculations wrong (60 vs 100 ticks/second)
vtroff--ucb                         jimbo@ucbic (Jim Kleckner)   11 Oct 83 +FIX
        handles "-" option wrong
which.csh--ucb     !garfield!andrew@BERKELEY (Andrew Draskoy)   17 Jan 84 +FIX
        doesn't see aliases
whois--man                jsq@ut-sally.ARPA (John Quarterman)   12 Jan 84 +FIX
        no manual entry
```

From: vance@mtxinu.UUCP
Date: Wed, 16-May-84 10:22:26 AEST
Newsgroups: net.general,net.unix-wizards,net.bugs.4bsd,net.unix
Subject: 4.2BUGLIST FROM MT XINU
Organization: mt Xinu, Berkeley

A legal problem has been raised with the University of California, Berkeley, regarding processing/distribution of 4.2 buglist material by mt Xinu. UC has asked us to stop any further buglist activity pending analysis of this problem and mt Xinu has agreed.

We do not know anything about the exact nature of the legal question at this time, nor do we have any idea how long our buglist effort may be on hold. We are sorry about this. We feel that the buglist distribution is a service to the 4.2 community and to UC and we believe that our arrangement with UC is fair to all concerned.

Please hold any additional requests or inquiries to mt Xinu concerning buglist material. We will make additional announcements as the issues are clarified.