

AUSTRALIAN UNIX USERS GROUP NEWSLETTER

\*\*\*\*\*  
 \* This document may contain information covered by \*  
 \* one or more licenses, copyrights and \*  
 \* non-disclosure agreements. Circulation of this \*  
 \* document is restricted to holders of a license for \*  
 \* the UNIX software system from Western Electric. \*  
 \* Such license holders may reproduce this document \*  
 \* for uses in conformity with the UNIX license. All \*  
 \* other circulation or reproduction is prohibited. \*  
 \*\*\*\*\*

---

 NEXT USERGROUP MEETING
 

---

The next Australian User Group Meeting will be held on Wednesday September 12th. Unfortunately this is in term time, but it will give those from interstate and New Zealand an opportunity to attend, given that the "Symposium on Language Design ...." is being held in Sydney that week. Also Dennis Ritchie will attend, in fact he will be giving a talk at the meeting on UNIX Version 7.

At previous meetings my experience has indicated that the informal parts of the meeting have been very useful and informative, and should be encouraged. At this meeting plenty of opportunities will be available for this very purpose, especially given the presence of DMR and other new and rarely seen faces.

Since the meeting is being held in term time getting lunch on campus will be awkward so I have arranged a smorgasbord in the University Union's private dining rooms. Term time also makes it very difficult to get a suitable room at suitable times for the meeting for free. As AGSM is currently supporting the newsletter, morning and afternoon teas are not free either. Thus a registration fee of \$10 per head is required to meet these costs. The Squarehouse where the meeting is to be held provides all the necessary facilities.

A final schedule for the meeting cannot be made until I hear from those people who wish to talk. Those of you wishing to attend please complete registration form and return by August 31st at latest. I will mail final details during first week of September.

See you all in September !!

---

 UNIX OS
 

---

I attended the last US/CANADIAN User Group Conference held at the University of Toronto on June 19-23 last. The conference enabled me to hear, meet, and speak to many other users of UNIX from America, Canada, England and Europe.

I include in this newsletter the conference report - my thanks to David Phillips from University of Toronto for preparing it. The various conference tapes alluded to in the report are available from me. See Robert Elz' letter for his initial impressions of tape contents.

---

## UNIX SOFTWARE AVAILABLE FROM OTHER SOURCES

---

Here is a list of major items of software available for use with UNIX. Unfortunately for those with Interdata machines a lot is PDP11 orientated. You can assist by helping to keep this list up to date, if any software is missing from the list or comments of an item are incorrect please advise.

MODULA - A Language for modular multiprogramming designed at Zurich by Prof. Wirth's group, and implemented for UNIX at University of York. The compiler is written in BCPL (a BCPL compiler is also provided : binary) and can produce object code capable of running under Unix, or for a stand alone PDP-11 or LSI-11 with or without EIS, for which run time systems are provided. For a description of the Modula Language see N. Wirth "MODULA: A Language for modular multi-programming", Software Practice and Experience Vol 7 No 1 (1977). For further details and information on the Unix version, contact

Mr I D Cottam  
Dept. of Computer Science  
University of York  
Heslington  
York YO1 5DD  
England

ALGOL 68S - Extended version of the Algol 68S Language, originally implemented for RSX-11, now available under Unix. Distribution details from:

Algol 68 Distribution Manager  
Dept. of Computer Science  
Winnipeg  
Manitoba R3T 2N2  
Canada

INGRES Relational Data Base System - Designed and built with Unix in mind, but puts a fairly heavy load on host system. Experience at UNSW suggests that it is barely usable on an 11/40. For details of the design philosophy see Stonebraker et al "The design and implementation of INGRES" ACM TODS Vol 1, No 3 (Sept 76) pp 189-222. For license information contact:

Bob Epstein  
Electronics Research Laboratory  
UC Berkeley  
Berkeley California 94720  
U.S.A.

FORTRAN-IV - Needs DEC Licence. Good and reasonably reliable, some bugs (none too serious), produces threaded code but still manages reasonable speed, documentation is sufficient. Overlaying linker is available at UNSW for those who like to run large programs.

Peter Bloomfield  
Department of Statistics  
Princeton University  
Princeton NJ 08540  
U.S.A.

FORTRAN 4-PLUS - Needs DEC Licence plus Licence from CULC. Cost \$3000 US. Produces good PDP11 code and runs very quickly, and has reasonable overlaying facilities, BUT it has bugs and the documentation is poor.

Commercial Union Leasing Corporation  
115 East 57th Street  
New York NY 10022  
U.S.A.

BERKELEY PASCAL - Written in C and AS - works well, very fast compilation, good diagnostics - used at UNSW for teaching undergraduates. A good compiler and interpreter with good documentation. Easy to make and install.

Bill Joy  
CS Division  
Department of EE and CS  
UC Berkeley  
Berkeley California 94720  
U.S.A.

VRIJE PASCAL - this version compiles to either EMI (interpreted) or PDP11 code. Compilation is about twice as long for the former, and about 6 times as long for the latter, with respect to Berkeley Pascal. Execution speed is about the same for EMI code, and about 3 to 4 times faster for PDP code. This version is still undergoing testing.

Dr Andrew S Tanenbaum  
Wiskundig Seminarium  
Vrije Universiteit  
Postbox 7161  
1007 MC Amsterdam  
The Netherlands

SPITBOL - an implementation for PDP11s of SNOBOL 4. A good implementation, good documentation. A few minor bugs with unevaluated expressions. Later versions are better integrated into UNIX environment.

Dewar Information Systems Corp  
221 West Lake Street  
Oak Park, Illinois 60302  
U.S.A.

BASIC-11 is a variant of DEC RT-11 BASIC, which was modified in July 74 at Harvard. Language Manual DEC-11-LIBBA-A-D. Needs DEC licence; includes matrix operations; works well. Also contains MACRO-11 assembler and linker.

Undergraduate Science Center  
Harvard University  
1 Oxford Street  
Cambridge, Mass. 02138  
U.S.A.

BASIC-PLUS - needs DEC Licence; corresponds to RSTS/E 5B. A service charge may be involved. Language manual DEC-11-ORBPB-A-D.

Dr W. H. Huggins  
Department of Electrical Engineering  
The John Hopkins University  
Baltimore MD 21218  
U.S.A.

INFORMATION SYSTEM - a general purpose information storage and retrieval system - a database package; very fast access to keyed items; concurrent updates by multiple users; easily maintained. Written in C. Documentation good; some bugs, none too serious. Cost based on the encompassing budget of the group who will use it. All things considered it is inexpensive for the facilities offered.

Bill Mayhew  
Center for Advanced Public Computing  
The Children's Museum  
The Jamaica Way  
Boston Massachusetts 02130  
U.S.A.

IDA - Interactive Data Analysis; written in Fortran and provides an excellent statistical package for student use; excellent diagnostics, and more importantly provides good assistance where necessary if the user doesn't understand what is required. Only problem is size, our binary consumes 1100 blocks.

Niesje Parker  
Graduate School of Business  
University of Chicago  
Chicago Illinois 60637  
U.S.A.

WATFOR/WATBOL - These well-known student compilers for Fortran and Cobol are now available under UNIX. As both use the same monitor, write for both together to be sure of compatibility.

Sandra Ward  
Computer Systems Group  
University of Waterloo  
Waterloo  
Ontario N2L 3GL  
Canada

BMD PDP11 Version 2 of BMDP77 - a complete implementation of the May 78 release from UCLA. Cost \$250 US for two years for degree-granting institutions. Written in FORTRAN. Unix version has just become available.

Software Development Corp  
Middlebury College  
PO Box 500  
Middlebury Vermont 05753  
U.S.A.

---

PS BUG

---

Recently a bug in 'ps' was discovered. It revealed itself when 'ps' printed times in excess of 20 days! Who said UNIX systems weren't reliable!!

---

CONTRIBUTIONS

---

I am always ready to accept same.

Ian Johnstone  
AGSM  
PO Box 1  
Kensington 2033  
AUSTRALIA

(02) 662-3752

האוניברסיטה העברית בירושלים  
THE HEBREW UNIVERSITY OF JERUSALEM



Department of Computer Science  
Institute of Mathematics  
Jerusalem, Israel.

החוג למדעי המחשב  
המכון למתמטיקה

May 17, 1979

Mr. Ian Johnstone  
Australian Graduate School of Management  
University of New South Wales  
P.O. Box 1  
Kensington, New South Wales  
Australia 2033

Dear Ian,

I wish to apologize for our extremely slow response. The reason is that Daniel Brannis is on a 6-month leave of absence right now, and our whole UNIX staff includes, at the moment, 4 M. Sc. students and some 10 users.

Not having a Magtape unit, we had some difficulties reading your tape -- we read it on our CDC and passed the information over to PDP via a UT-200 emulator (4800 baud when the CDC is not down). CDC software has a magic touch to it -- the "copy" program strictly refused to copy your tape on ours (blocks too large or something), and after a hopeless struggle we turned for help elsewhere.

We wish to thank you for the software. We implemented some of it already -- the rest will keep us busy for another decade.

Some local developments are on the enclosed tape (your tape). A list is appended. The collection is rather dull, I am afraid. I hope it will be of some use to you (at least the Hebrew-date program you surely don't have -- just what you always wanted!). Definite clues (like sourceless programs) point to more local production -- it's just that undebugged and undocumented software seems very popular around here.

Are you workbenching a Cyber, and if you do, does your workbench enable interaction with a program running on the Cyber? Our agonizing battle with the UT-200 protocol brought us barely to the point of file transfer.

Our plans to expand are in some undefined bureaucratic stage -- purchasing a VAX is contemplated, too -- no action seems probable in the near future.

Thank you again.

Yours sincerely,

*G. Steinberg*  
Gabi Steinberg

P.S. We have not yet received the billing charge for your software distribution.

5

List of included HUJI software

dos - pape-by-pape cat  
du - sync line interface driver.  
dz - 8 line mux driver.  
edd - direct access screen editor  
ed - some changes  
hdate - hebrew date.  
idle, activate - system calls for process synchronization.  
lp - print input on decwriter, in the shortest time.  
mail - without security buss (login).  
mkmesyztp - package to read & write UNIX tapes on SCOPE  
rx - a floppy disk driver.  
ut - Emulatory runs as a users process. For file transfer needs  
2 format conversion programs (included).  
ball.c, pool.c - vt games, using slib.  
st.s - stand-alone driver for GT40 (offspring of a unix GT42 driver)  
including slib.  
slib - an interactive graphic package.



THE UNIVERSITY OF NEWCASTLE  
NEW SOUTH WALES, 2308

DEPARTMENT OF MATHEMATICS

TELEPHONE 68 0401

EXT. 596

BC:AM

29th May, 1979.

Dr. I. Johnstone,  
Australian Graduate School  
of Management,  
University of New South Wales,  
P.O. Box 1,  
KENSINGTON. N.S.W. 2033

Dear Ian,

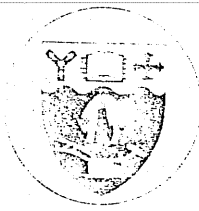
I enclose a tape on which I hope you will be able to  
send us a copy of the University of New South Wales' distribution  
of UNIX.

Thanks for your help.

Yours,

Bruce Cheek.

Enc.



25 June 1979

Mr Ian Johnstone  
Australian Graduate School of Management  
University of New South Wales  
P.O. Box 1  
Kensington NSW 2033

Dear Mr Johnstone

Thank you for including my letter in the recent edition of AUUGN. I will be interested to see what, if any, reaction it receives.

The latest Newsletter contained references to a number of items in which we are very interested. In particular, we would like to get hold of:-

C - Compiler for 8080 and 6502  
The "portable" C - Compiler  
Drivers for DZ-11 and RL01  
A copy of UNSW version of UNIX

It is clear that there is also a number of facilities (including LINT) which we did not receive with our copy of UNIX from Bell Labs (should we have?). As we were not able to attend the UNIX Conference, I am unaware of what procedure there is for copying these, say, to a mag tape and shipping them. Can you clarify please?

I guess what we are really interested in, is there a Library of Programmes available under UNIX, and, if so, costs, ordering procedures, etc.

Yours sincerely

R.E.M. Cooper  
Senior Lecturer  
Dept. of Computer Science



c/o Department of Computer Science  
University of Canterbury  
Private Bag  
Christchurch  
New Zealand

June 26, 1979

Mr. I. Johnstone  
AGSM  
P.O. Box One  
Kensington 2033  
Australia

Dear Sir:

I am a post-graduate student currently researching various aspects of operating system portability, and consequently am greatly interested in UNIX. In this connection, I am planning to attend the Symposium on Language Design and Programming Methodology in September, and the Australian Unix Users' Group meeting on September 12.

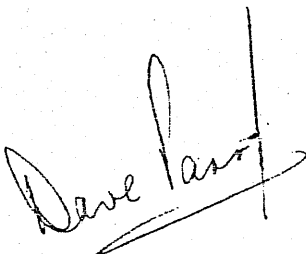
In view of the fact that, as a student, I am finding the financing of the trip to Sydney a problem, I would be grateful if you could advise if cheaper alternative accommodation is available than that suggested in the brochure for the symposium.

I would also like to take the opportunity of seeing a working UNIX installation such as your's, and would be grateful if this could be arranged while I am there. Although we are UNIX licensees, we have been supplied with the 6th Edition which does not have the essentials (to us) of RL-01 and DZ-11 drives and consequently our use of the system is severely limited.

In view of the increasing interest in UNIX within our department, I would also be grateful if you could place my name on your mailing list for the AUUG newsletter.

Thanking you in advance,

Yours sincerely,



D.M. PARRY



# University of Queensland

DEPARTMENT OF COMPUTER SCIENCE  
ST. LUCIA, BRISBANE, AUSTRALIA. 4067

3rd July, 1979

Mr. I. Johnstone,  
Australian Graduate School of  
Management,  
P.O. Box 1,  
KENSINGTON, N.S.W. 2033

Dear Mr. Johnstone,

I would be grateful if you could send me a complete release of N.S.W. unix. I believe that we have only ever received parts of unix on RK05's. We now have a tape driver and a very cannibalised form of unix and would now like to get our house in order!

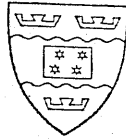
Thank you,

A handwritten signature in cursive script that reads "Hamish Bentley".

Hamish Bentley

# Victoria University of Wellington

Telephone 721-000



Private Bag  
Wellington  
New Zealand

3 July 1979

Dr I. Johnstone  
AGSM  
P.O. Box 1  
Kensington  
N.S.W. 2033  
AUSTRALIA

Dear Ian,

I hope to get to the AAEC, UNSW Symposium in September and understand that your UNIX Users' Group is meeting on the Wednesday after the Symposium. Please send me details as I would like to attend the meeting, even though we have not yet made a decision to switch to UNIX.

I look forward to your reply.

Yours sincerely,

A handwritten signature in cursive script, appearing to read 'Colin Boswell'.

Dr Colin Boswell  
DIRECTOR  
COMPUTING SERVICES CENTRE



YOUR REFERENCE:

OUR REFERENCE:

# University of Queensland

ELECTRICAL ENGINEERING  
DEPARTMENT

ST. LUCIA, BRISBANE  
AUSTRALIA, 4067

4th July, 1979.

Mr P. Ivanoff,  
Department of Computing Science,  
University of New South Wales,  
P.O. Box 1,  
KENSINGTON, N.S.W. 2033.

Dear Sir,

I am writing to you, in the absence of Ian Johstone, to see if you could send me some source files for the Unix Fortran Linker. The source files that we received here originally for link do not correspond with the binary version. The feature that is missing is the overlay switch.

I believe that Hamish Bentley of our Computing Science Department has sent down some tapes to get a copy of the latest UNSW distribution. Perhaps you could include the link source files on this. If that is not possible I could send a floppy or a cartridge.

Yours sincerely,

*C. Harridge*

(C. Harridge)  
Electrical Engineering

UNIVERSITY OF GLASGOW



TEL: 041-339 8855  
EXT. 478/7458

Computing Science Department,  
THE UNIVERSITY,  
GLASGOW, G12 8QQ.

4th July, 1979.

Dr. I. Johnstone,  
Australian Graduate School of Management,  
University of New South Wales,  
P.O. Box 1,  
Kensington,  
New South Wales,  
AUSTRALIA 2033.

Dear Ian,

I note from your latest newsletter, vol I no. IV (for which much thanks) that you have an improved version of the Princeton Fortran system, with overlay facilities added to the loader. We would be most interested in obtaining a copy of this (assuming it is not included in the stuff you gave to Peter Collinson at Toronto). I enclose a copy of the letter of authorisation from Peter Bloomfield at Princeton.

Yours sincerely,

*Alistair C. Kilgour*

Alistair C. Kilgour.

encl.

TELEPHONE  
345 1844

TELEGRAMS  
UNIMELB PARKVILLE



# University of Melbourne

DEPARTMENT OF COMPUTER SCIENCE

Parkville, Victoria 3052  
Mon, 30 Jul, 1979

Mr Ian Johnstone,  
Australian Graduate School of Management,  
University of New South Wales,  
Kensington, N.S.W. 2033.

*this is the Toronto  
conference tape!*

Dear Ian,

Thanks for the tape, it has turned out to satisfy just about every need (and all in less than a week). In case you still haven't had an opportunity to go through the incredible number of files (you didn't mention that there were 53000 blocks of disc space) I was going to include a list of what we have found so far, but it turns out to be such a miniscule proportion, as to be worthless. There have been a few days of very intensive investigation by a couple people, and there are still many "cont.a"'s lying around untouched.

A couple of things a worthy of mention, the version of APL from Purdue is a significant improvement over the version floating around Australia at the minute. There are still some notable bugs, in particular 'scan' is incorrect for non-associative operators (I have a fix for that) and function error handling gradually eats up stack space (and has a few other bugs) (fix almost ready).

There is also a 'picture editor' for drawing circuit diagrams on Tektronix 4014's, and HP flatbed plotters. We have it working on a 4012 (with some help from SU's character generation routines) and are working on mods to permit hard copy output to a Versatec.

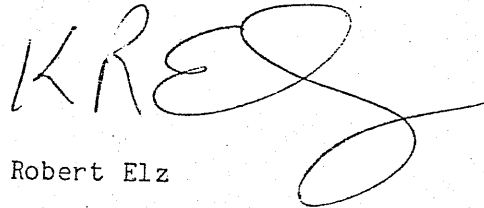
There are also a couple of editors with command file capabilities, numerous shells (typically half way between Version 6 and Version 7), several games (dungeon - binary only, patched from an LSI 11 version; adventure (naturally - the Fortran Version); a thing called 'search' which is a multi-terminal sort of space war (needs a new sys call to test for anything to read on a tty). There are also quite a number of cross assemblers for various micros.

On the system side, there are mods to handle bad disc blocks in a more orderly fashion, recover from swap errors, drivers for psuedo-ttys and any DEC device that I can think of (which isn't really very many I must admit).

There is also a whole lot of stuff written to conform to PWB specifications, which may be of interest to anyone who doesn't have PWB but wants some of the bits.

I am also including my much promised ideas on terminal handlers, I hope that they arrive in time for your deadlines for AUUGN, and apologise for the rather poor quality, our Diablo is not in the best of health just at the minute. In a way I feel that this might explain the slight tardiness of Unix users toward sending odds and ends to be included in AUUGN. The standard has been so high, that anything we may have to offer seems to vanish into obscurity. It seems strange to say it, but you may have to lower your own standards to get more submissions. I hope it doesn't come to that, so I will do my bit towards ruining AUUGN by sending this short piece. This may encourage others to reply, or adopt a similar attitude, while allowing you to continue the excellent work that you have been doing.

Thanks again for the tape, and for sending it so quickly,

A handwritten signature in cursive script, appearing to read 'KREZ'.

Robert Elz

I agree  
IANJ

## Terminal Interfaces

Robert Elz  
Computer Science  
University of Melbourne

This short opinion on the properties that a terminal interface could have, or ought have, is intended to provoke a discussion along similar lines at the forthcoming UNIX Users' Group meeting in September, with a possible outcome being the standardization of

- a) a basic terminal driver interface, to both the user and his programs.
- b) a uniform and accepted method for various installations to make their local modifications (eg: to suit a particular local terminal type.)
- c) a standard set of system calls, since the current stty, and gtty are clearly inadequate.

Before continuing, I must say that the following views are personal, and do not necessarily reflect those of other Melbourne University users. Also, I have not yet had the opportunity to examine UNIX Version 7 nor UNIX V32, so the comments contained herein should be understood to apply to Version 6 UNIX only.

Some of the 'features' discussed below are currently incorporated in the Melbourne tty driver, others are under consideration for a forthcoming version, the remainder are either not really useful, or too difficult to implement in a UNIX setting to be really serious contenders.

### 1 Motivation

Before getting down to the nitty gritty, I am inclined to spend a paragraph or two justifying the amount of time I invest in constant meddling with the terminal interface.

Apart from such pragmatic considerations as the gross inefficiency of the original Bell design for Interdata hardware, and the appalling jumble of code in 'canon()', my principal motivating force is my view that the terminal driver is the single most important component of a timesharing system to the user of that system.



Others regard the editor, or the command interpreter as being more significant, however I feel that however marvellous such utilities may be, the user will never be truly comfortable unless he is able to communicate with them in the manner with which he feels most at home. After all, while a user may spend 50% of his time in the editor, and most of the remaining 50% using the command interpreter, all of his logon time is spent using the terminal.

## 2 What the terminal interface should offer the user

- a) There must be a facility to delete the previous character on the current line, and the user ought to be able to obtain visible confirmation that the character has indeed been deleted. On a CRT this should be accomplished by actually erasing the character (with proper allowance made for erasing tabs). On hard copy terminals (or storage screen types, like the Tektronix) the user ought to have the option of seeing the character deleted if he desires (not always - it takes both time and paper), or of simply getting the erase character echoed in some visible form. An attractive idea for erasing on hard copy terminals would be to write a cross ('\' ← '/') over the character deleted, if only it weren't for the fact that most hard copy terminals run at 300 baud (or less). In all cases, if there is nothing left to erase, then nothing at all should happen on the terminal.
- b) There should be the ability to delete the whole line and start it again. On a CRT it would be ideal if the line could be made to vanish, on other terminals the user ought to have the option of whether a new line should actually be taken or not. Because of the drastic consequences of an accidental use of this command, it ought to be reversible (most usefully using the erase mechanism above), though possibly with restrictions on when this may be done. If the line delete character is erased, the terminal should be restored as nearly as is possible to the state it was in before it was entered (ie: the 'un-killed' line ought to re-appear).
- c) There should be a mechanism to have the contents of the current line redisplayed. If there is a deleted line which could still be recovered, an indication of this ought to be given (perhaps its contents too).
- d) There should be mechanisms for signalling a process from the terminal and for indicating that there is no more input to be sent.
- e) It should be possible for a user to temporarily suspend output from being sent to his terminal, and (of course) to restart it again. This is particularly important for terminals which decide for themselves when they have had enough, and send a command to the processor to stop sending.
- f) There should be a means to divert output from the terminal to another destination (especially to 'the bit bucket) without the consent of the issuing program.

- g) There should be an escape mechanism to permit any of the above commands to be entered as data.
- h) All of the command characters should be re-definable by the user at will, especially those mentioned in (e) (which vary from terminal to terminal) and (a) which is the most often used.
- i) It should be possible to have all characters typed on the terminal echoed in some visible form, that is, there should be an option to prevent there from being any character that could be treated as input to a program, and yet leave no sign on the terminal that they have been typed.
- j) The terminal should be able to operate in any of three modes.
  - (i) continuous output.
  - (ii) fixed paging.
  - (iii) variable paging.
  - (i) is the same as the way most terminals are used on UNIX now.
  - (ii) causes output to the terminal to be suspended every so-many lines, to give the user a chance to read it. Output should not recommence until the user requests it. Input from the user at this time should be treated as commands indicating what he wants done, and as a minimum he should be able to move to the next page, move some part of a page, and exit page mode (temporarily). The only time that this halt should not occur is when the line that fills the page was typed by the user. Whenever a new page is started the screen should (optionally) be cleared.
  - (iii) is similar to (ii) except that any input from the user indicates the next line is the beginning of a new page, rather than just an input line at the very end of a page.
- k) Lines being typed should be able to be given priority over output from some program, if the user wants this. That is, program output should be suspended until the user finishes typing his line.
- l) Terminals with limited line widths should be catered for by optionally folding long lines if this is not performed by the terminal hardware. In any case where folding takes place, the terminal handler should be aware of this and recognise that an extra line has been printed.
- m) Terminals with limited character sets should be supported as fully as possible, in a manner that places the minimum of strain on the user.

### 3 Terminal / Program Interfaces

- a) A program ought to be able to determine the modes in which a terminal is currently being used, and alter any it chooses to.

- b) It should be possible to obtain a unique identification of the type of terminal in use, or inform the system of the type if the program deems itself to be a better judge.
- c) There should be a standard set of terminal control characters to perform specific terminal functions (such as clearing the screen, or moving the cursor to the left). The interfacing software ought to take care of any mapping required (possibly into a character sequence.)
- d) A process should be able to determine if there is any input from the user waiting to be read and determine if this amounts to a complete line or simply some stray characters. It should be possible to discard typeahead, but this should not be a by-product of other alterations.
- e) It should be possible to define a set of characters to be line terminators, which delimit 'lines' in the users input.
- f) The terminal's width and depth should be able to be ascertained, as should the current characters being used by the user for control purposes.
- g) Output delays for the various usual functions requiring them should be individually settable over a generous continuous range. A program ought to be able to specifically request a delay at any point in the output stream.
- h) Options should be independantly switchable - selecting a particular format should not imply that some other attribute either must or cannot be selected.

#### 4 Local Modifications

When designing a terminal interface to be used in a number on environments, it is important to recognise the fact that individual installations will have to customize it for their own particular requirements.

This may amount to no more than the deletion of unnecessary code, eg: if there are no terminals using any character other than the standard for some particular terminal function, then the mapping ought to be discarded.

In other cases, the peculiarities of a particular terminal may warrant the addition of some code, eg: to output the escape sequence to return a particular type of terminal to full duplex after an especially foolish user has sought out and maliciously depressed the 'break' key, which is carefully hidden adjacent to 'return', or to prevent transmitting the relevant control codes to those terminals which will happily reply with the complete contents of the screen, unless the program to receive it all has indicated its willingness to accept this data.

Since such modifications will take place, it is best to allow for them by providing installation dependant interface mechanisms, and specifying only that some value is the default to allow portable software. This might take the form of 'empty space' in a system call data block, where zero is to represent the normal case, and any other value will invoke some installation dependant option.

## 5 Relationship with UNIX

The terminal interface offered by UNIX is not too far removed from the ideal situation for there to be any great difficulty in making some useful improvements.

This has indeed been done at many UNIX sites, but almost without exception the changes have been more in the nature of repairs than renovation. The best indicator of this is the 'stty' system call. In a heroic attempt to remain as close as possible to the released version of UNIX, most modifiers of the terminal handler have sought to remain with the standard 'sgtty' structure, and have been content with alterations like:

"We don't really need backspace delays, so let's use that bit for ....."

I propose a total redesign of 'stty' to permit a much larger parameter area, since there is just so much that can be packed into 48 bits, and it isn't enough. Interdata UNIX allows 96 bits (3 integers, 32 bits each) and the current Melbourne terminal handler makes use of all but about 4 of them, and is still missing a lot that I regard as essential (even without the frills).

It may be appropriate to provide extra system calls, and assign each a particular role, dividing the control information among them upon a basis of likelihood of use, so that information rarely needed would not be being continually moved around. (Eg: very few programs have the slightest interest in the speed that the terminal runs at, and those that currently do usually only want to calculate how many fill characters are necessary to simulate a delay. If there was a means to request a delay of a specific length, then effectively no program at all would want to examine the terminal speed. This information would be better confined to the programs that really require it.)

At the present time, I have no fixed ideas as to what ought to replace, or at least augment the 'stty' and 'gtty' sys calls, but there should be at least 64 bits of on/off flags; 8 bit delay values for new lines, tabs, carriage returns, form feeds, vertical tabs (and probably backspaces); about a dozen user level control characters (erase, kill, interrupt, quit, eof, suspend, continue, discard, redisplay, escape); character sequences for clear the screen, simple cursor movements, and bell ringing; an array of 128 bits to specify line terminating characters; 8 (or 4?) bit fields for transmitter and receiver speeds; and lots of gaps for things we haven't thought of yet. In all a minimum of the

order of 400 bits.

## 6 The Current Melbourne Interface

As mentioned above, 'stty' and 'gtty' are still used at Melbourne, in much the same form as in the standard Version 6 UNIX, except that there is twice as much space. This excess has been used to allow 32 flag (mode) bits (which still include the delay choice), 6 settable user interaction control characters, and two speeds (though Interdata hardware utilizes only one of them, there is no 'split speed').

Following an outline similar to sections 2, 3, and 4 above, there follows a brief summary:

### 6.1 User Interface

- a) Erase is much like the UNIX standard, except that the actual erasing is performed when the erase character is read, rather than when a program requests the line. This means that on a CRT the erased character can be made to vanish (and it is, except for tabs, which are not handled correctly at all). On other terminals, the user can choose either to see the characters that are being erased, enclosed in '#' delimiters, or to have his 'erase' character echoed. If that character is 'rubout' (ASCII 0177) he may choose to have it displayed as a '#' instead. If there are no characters in the line to erase, nothing is echoed.
- b) The current line may be deleted (as in any other UNIX). It is possible to request that a 'newline' sequence follow the echo of the 'kill' character if you want that. The 'kill' may be erased, but only if the character immediately following the 'kill' is 'erase' (there are a couple of exceptions actually). If this is done, then, if a new line was taken the 'un-killed' line will be echoed, and the cursor left at the end of it, otherwise the standard erase sequence is performed, illustrating the 'kill' character being removed.
- c) It is possible to redisplay the current line. If there is a killed line that could be ressurected, then that line is displayed (it appears in the same manner as if the user had just typed it).
- d) Interrupt and quit are as in standard UNIX. It is possible to use the terminal 'break' key to signal interrupts (and it can be left enabled in 'raw' mode) as an alternate to the ordinary character. If 'break' is to be the only interrupt key, then both 'interrupt' and 'quit' can be set to the same value, the driver guarantees to signal 'quit' if the character is typed. End of file is as in standard UNIX.
- e) It is possible to suspend output, and resume again later. Both 'interrupt' and 'quit' also end a suspension. Input typed while

output is suspended is echoed when the suspension is lifted.

- f) Output cannot be discarded nor redirected.
- g) Any of the characters above can be escaped, except 'interrupt', 'quit', 'pause', and 'suspend'. Missing the latter two is a definite bug, I am unsure whether it should be possible to escape 'interrupt' and 'quit' characters.
- h) The six characters that can be altered are 'erase', 'kill', 'interrupt', 'quit', 'end of file', and 'escape'. There are two choices for each of 'suspend' and 'pause' (to correspond to the terminals that we use) - either character may be used. 'Redisplay' is control-A, and cannot be altered.
- i) The only control characters that are ever graphically displayed (other than in performing actions such as erasing a character) are ESC (ASCII 033), which can be displayed as a '\ ' if desired (and it always is) and RUBOUT (ASCII 0177), which can be displayed as a '#'. These conventions arose from the common use of ESC as the terminal 'escape' character (replacing the much overworked '\ ') and RUBOUT as 'erase' (replacing '#' which is too hard to type).
- j) There is no form of paging.
- k) There is no line folding.
- l) Output from a program cannot be restrained from interrupting the users input line.
- m) Limited character set terminals are handled much as in standard UNIX, but the user has the choice of whether he wants the pre-translated character echoed, or the post-translated character (to make life easier on terminals that can display lower case but not transmit it).

## 6.2 Program Interface

- a) 'Stty' and 'gtty' operate as in any other UNIX.
- b) Terminal identification is not available.
- c) On the one terminal type that does not use backspace (ASCII 010) for cursor left, a translation is made. No other common sequences are handled.
- d) There is no way to determine if the user has typed anything without attempting a 'read'. It is possible to discard typeahead ('stty' as is usual), but it is also possible to cause 'stty' to retain typeahead, if the terminal speed is not altered, and there is no switch between 'raw' and 'cooked' modes.
- e) Lines are terminated by 'newline' or 'end of file' as is usual.

- f) Page width and depth are not known to anyone, except the poor user who must explicitly specify them wherever a program needs the information.
- g) Delays are close to the UNIX standard, except that there are no backspace delays, but there are three different vertical motion delays. Programs cannot request delays (other than by emitting a relevant character).
- h) Flags are comparatively independent. In particular, 'raw' mode does not disable 'crmod' or 'lcase' or just about anything else.

### 6.3 Local modifications

There are even a few 'features' that are considered peculiar to our installation, with both the possibilities mentioned in section 4 being included, as well as a peculiar 'fast' mode akin to 'raw', but supposedly more efficient for reading large amounts of data at high speed, and a mechanism to put the line into a 'space' condition for about 250 milliseconds, both intended for some form of intermachine communication over RS232 lines.

### 7 Conclusion

I have attempted to keep my ideas within the bounds of reasonable possibility for a system like UNIX, and have resisted the temptation to include extras like the ability to retrieve previous lines, or the 'the way it looks on the terminal is the way I want it read' philosophy, which has much merit, but is not easy to implement.

There are 'obvious' countless other small facilities that would be useful if included, but which I haven't ever encountered and thus remain ignorant of. If any reader has any suggestions I would be grateful to learn of them, either at the Users' Group meeting, or at some other time, whether they be minor improvements, implementation methods, or straight out laughter.

The Melbourne terminal driver is available to anyone who would like a copy (send me a tape), but you should be warned that there is much in it that cannot be made to work on a PDP-11, (though nothing at the level of this discussion - mostly at the hardware interface stages) and that there is a major revision in the offing. I intend to include paging, and line folding, and probably variable line termination if I can make them all fit. This will certainly require modifications to the 'stty' format so I will not embark on this until after the Users' Group meeting, in case there is a decision made as to what things really ought to be like.

Kevin Hill,  
Department of Computer Science,  
UNSW.  
31st July, 1979.

Ian Johnstone,  
AGSM.

Dear Ian,

the 11/40 at the Department of Computer Science (UNSW) uses three rk05's for file storage and operation. As with most other similar systems, disk space is at a premium. AUUGN readers may therefore be interested in the following technique that not only leads to a significant saving of space on the system disk, but also makes linking into libraries faster.

The technique is simply to remove all local symbols from the symbol tables of each library component. This is done most simply at compile time, by executing the commands

```
`ld -rx obj.o ; mv a.out obj.o`
```

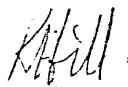
For existing libraries, a simple shell procedure can be written to extract all components, execute the above command, and replace them all back into the library.

When done on the 11/70 here, this saved 198 blocks on the system disk, and 71 blocks on the source disk (total 269 blocks). This involved stripping 16 libraries on the system disk, and the two system libraries (lib1 and lib2). Savings for the more common libraries were: libc.a (20 blocks), liba.a (3 blocks), and libS.a (1 block).

A secondary effect of this is faster linking, as the files are smaller, and there are fewer symbols to scan. Linking the system was 13% faster, and linking the editor was 12% faster.

I am currently looking into adding a flag to the assembler, to prevent it writing these local symbols in the symbol table in the first place. CC would then use this flag (by default), and it would be unnecessary to remove the local symbols at a later stage.

Yours sincerely,





Kevin Hill,  
Department of Computer Science,  
UNSW.  
31st July, 1979.

Ian Johnstone,  
AGSM.

Dear Ian,

the following notes resulted from my efforts to install the new Vrije Pascal Compiler (pc) on the 11/70 here. As this distribution presented several problems, these notes will probably be of interest to other people who are attempting to install pc.

The first thing I did (after extracting all the files and reading the READ\_ME) was to head for the 'doc' directory, to start "nroffing". It soon became apparent, however, that "life was not meant to easy". Numerous corrections (apart from obvious line and page length changes) were required to the files, as follows.

The first problem was that the file 'dist.nr' contained several macro definitions of the form '.de PA PARAGRAPH'. PARAGRAPH is obviously meant to be a comment, however, used this way, it caused our version of nroff to gobble up the entire remainder of the text, and produce nothing. The reason was that any characters after the macro name are taken by our nroff to be the macro-terminator ('..' by default). The simple fix was, for the example above, to change it to '.de PA \" PARAGRAPH'.

The next problem was in the file 'pdp.doc', in which two lines were starting with '.globl', causing nroff to ignore the line after failing to find any macro '.gl'. This was fixed by changing them to '\&.globl'.

Three other minor fixes involved surrounding a diagram in the file 'eml.doc' with a '.nf', '.fi' pair, adding a '.ta 8,16,....' to the same file, and making a 'run' file that combined 'eml.doc' and 'opt.doc' after it was noted that the latter was using the macro definitions of the former.

The next task was to convert all the source archive libraries provided from the old to the new format. This also involved changing the file 'ass/ass00.h' to know the new magic number and new layout (the programs including this file have to be able to search various libraries).

The first C-compilation met with a sad end due to function redefinitions, i.e., declaring a function to be returning a pointer to something after using it (in which case it will have been assumed to return an integer). Several additional declarations were required in 'misc/pc.c', 'misc/eml.c' and 'ass/ass70.c'. The file 'pdp/pdpci.c' actually pre-declared a certain function correctly as returning a pointer, but, at the actual function declaration, redeclared it as returning an integer.

Two more C-problems that surfaced were as follows. The file ``ass/ass00.c'` had an error at line 354, where a variable ``n_xprocs'` should have been ``n_xproc'`. The file ``pdp/pdp40.c'` and ``pdp/pdp43.c'` both contained lines of the form ``variable=ANY'`, where ANY was defined to be -1. This was fixed by simply putting spaces around the ``='`.

The file ``misc/em1.c'` caused trouble, by executing the following command: ``ld -o fname fname'`. This resulted in ``fname'` being clobbered with an empty 16-byte header only. The program was changed so that an earlier exec to the assembler wrote its output onto `a.out`, and the load became ``ld -o fname a.out'`.

The final note in this section is to remind readers NOT to blindly execute run files provided with distributions. Some are definitely better than others, e.g., those from Bill Joy at Berkeley leave little to be desired. In this case, several compilations were done with the ``-f'` flag (for floating-point simulation) (``ass/run'`, ``pdp/run'`, and ``rundir/testC'`), and no compilation was done with the ``-s'` flag (to strip off the symbol table). Further, on occasions, a file from a higher directory was linked into the current directory under a different name, the intention being to execute this one and not the system file of the same name. However, the directory search order for root at this installation has for a long time put the current directory last, and thus the wrong file would be used.

Apart from these minor problems, the whole show was eventually put together and installed, and compared with the Berkeley Pascal Interpreter (pi) that we are also using. Compilation and execution times for four test programs supplied were:

Compilation:		t1.p	t2.p	t3.p	t4.p
pi -p (Berkeley)	real	10.00	13.00	8.00	-
	user	7.20	8.10	4.94	-
	sys	1.50	2.64	0.96	-
pc (EM1 code)	real	24.00	25.00	21.00	12.00
	user	12.16	13.08	10.16	6.88
	sys	4.44	4.84	4.42	2.70
pc -C (PDP code)	real	1:14.00	1:06.00	50.00	34.00
	user	28.20	36.44	25.00	15.14
	sys	11.70	14.48	10.96	6.98

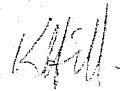
Execution:

obj	real	2.00	24.00	16.00	-
	user	0.30	18.54	9.50	-
	sys	0.42	2.66	4.04	-
e.out	real	2.00	20.00	16.00	7.00
	user	0.30	15.62	9.20	2.84
	sys	0.52	2.16	3.00	1.36
a.out	real	1.00	5.00	5.00	2.00
	user	0.04	2.36	0.60	0.38
	sys	0.24	0.50	2.16	0.60

As can be seen, pc is much slower at compiling, but execution is about 3 to 4 times faster (on average). However, this must be weighted against the size of the various libraries that it uses (about 730 blocks)!

Several immediately obvious differences were noted. PC does not force a leading blank in front of positive real numbers (when printed out), allows functions and procedures as arguments to other functions/procedures, implements the predefined operations 'mark' and 'release' (as used in test program t4.p, hence pi could not handle it), and is more intelligent about the empty set []. On the debit side, pc still requires that dummy 'getc(input)' before any input operation on the input file, complains if there are not sufficient arguments on the command line to match each file argument in the 'program' heading, and uses only 16-bit integers (pi uses 32 bit integers).

Yours sincerely,

A handwritten signature in dark ink, appearing to be 'K. Hill' or similar, written in a cursive style.

## UNIX V32 — Summary

February 12, 1979

Bell Laboratories  
Holmdel, New Jersey 07733

V32 functionally  
equivalent to  
level 7

### A. What's new: highlights of the UNIX† V32 System

**32-bit world.** UNIX V32 handles 32-bit addresses and 32-bit data. Devices are addressable to  $2^{31}$  bytes, files to  $2^{30}$  bytes.

**Portability.** Code of the operating system and most utilities has been extensively revised to minimize its dependence on particular hardware. UNIX V32 is highly compatible with UNIX version 7.

**Fortran 77.** F77 compiler for the new standard language is compatible with C at the object level. A Fortran structurer, STRUCT, converts old, ugly Fortran into RATFOR, a structured dialect usable with F77.

**Shell.** Completely new SH program supports string variables, trap handling, structured programming, user profiles, settable search path, multilevel file name generation, etc.

**Document preparation.** TROFF phototypesetter utility is standard. NROFF (for terminals) is now highly compatible with TROFF. MS macro package provides canned commands for many common formatting and layout situations. TBL provides an easy to learn language for preparing complicated tabular material. REFER fills in bibliographic citations from a data base.

**UNIX-to-UNIX file copy.** UUCP performs spooled file transfers between any two machines.

**Data processing.** SED stream editor does multiple editing functions in parallel on a data stream of indefinite length. AWK report generator does free-field pattern selection and arithmetic operations.

**Program development.** MAKE controls re-creation of complicated software, arranging for minimal recompilation.

**Debugging.** ADB does postmortem and breakpoint debugging.

**C language.** The language now supports definable data types, generalized initialization, block structure, long integers, unions, explicit type conversions. The LINT verifier does strong type checking and detection of probable errors and portability problems even across separately compiled functions.

**Lexical analyzer generator.** LEX converts specification of regular expressions and semantic actions into a recognizing subroutine. Analogous to YACC.

**Graphics.** Simple graph-drawing utility, graphic subroutines, and generalized plotting filters adapted to various devices are now standard.

**Standard input-output package.** Highly efficient buffered stream I/O is integrated with formatted input and output.

**Other.** The operating system and utilities have been enhanced and freed of restrictions in many other ways too numerous to relate.

† UNIX is a Trademark of Bell Laboratories.

## B. Hardware

The UNIX V32 operating system runs on a DEC VAX-11/780\* with at least the following equipment:

- memory: 256K bytes or more.
- disk: RP06 or equivalent.
- tape: any 9-track MASSBUS-compatible tape drive.

The following equipment is strongly recommended:

- communications controller such as DZ11 or DL11.
- full duplex 96-character ASCII terminals.
- extra disk for system backup.

The system is normally distributed on 9-track tape. The minimum memory and disk space specified is enough to run and maintain UNIX V32, and to keep all source on line. More memory will be needed to handle a large number of users, big data bases, diversified complements of devices, or large programs. The resident code occupies 40-55K bytes depending on configuration; system data also occupies 30-55K bytes.

## C. Software

Most of the programs available as UNIX V32 commands are listed. Source code and printed manuals are distributed for all of the listed software except games. Almost all of the code is written in C. Commands are self-contained and do not require extra setup information, unless specifically noted as "interactive." Interactive programs can be made to run from a prepared script simply by redirecting input. Most programs intended for interactive use (e.g., the editor) allow for an escape to command level (the Shell). Most file processing commands can also go from standard input to standard output ("filters"). The piping facility of the Shell may be used to connect such filters directly to the input or output of other programs.

### 1. Basic Software

This includes the time-sharing operating system with utilities, and a compiler for the programming language C—enough software to write and run new applications and to maintain or modify UNIX V32 itself.

#### 1.1. Operating System

- UNIX      The basic resident code on which everything else depends. Supports the system calls, and maintains the file system. A general description of UNIX design philosophy and system facilities appeared in the Communications of the ACM, July, 1974. A more extensive survey is in the Bell System Technical Journal for July-August 1978. Capabilities include:
  - Reentrant code for user processes.
  - "Group" access permissions for cooperative projects, with overlapping memberships.
  - Alarm-clock timeouts.
  - Timer-interrupt sampling and interprocess monitoring for debugging and measurement.
  - Multiplexed I/O for machine-to-machine communication.
- DEVICES    All I/O is logically synchronous. I/O devices are simply files in the file system. Normally, invisible buffering makes all physical record structure and device characteristics transparent and exploits the hardware's ability to do overlapped I/O. Unbuffered physical record I/O is available for unusual applications.

\*VAX is a Trademark of Digital Equipment Corporation.

Drivers for these devices are available:

- Asynchronous interfaces: DZ11, DL11. Support for most common ASCII terminals.
- Automatic calling unit interface: DN11.
- Printer/plotter: Versatek.
- Magnetic tape: TE16.
- Pack type disk: RP06; minimum-latency seek scheduling.
- Physical memory of VAX-11, or mapped memory in resident system.
- Null device.
- Recipes are supplied to aid the construction of drivers for:
  - Asynchronous interface: DH11.
  - Synchronous interface: DP11.
  - DECtape: TC11.
  - Fixed head disk: RS11, RS03 and RS04.
  - Cartridge-type disk: RK05.
  - Phototypesetter: Graphic Systems System/1 through DR11C.

- BOOT Procedures to get UNIX V32 started.

### 1.2. User Access Control

- LOGIN Sign on as a new user.
  - Verify password and establish user's individual and group (project) identity.
  - Adapt to characteristics of terminal.
  - Establish working directory.
  - Announce presence of mail (from MAIL).
  - Publish message of the day.
  - Execute user-specified profile.
  - Start command interpreter or other initial program.
- PASSWD Change a password.
  - User can change his own password.
  - Passwords are kept encrypted for security.
- NEWGRP Change working group (project). Protects against unauthorized changes to projects.

### 1.3. Terminal Handling

- TABS Set tab stops appropriately for specified terminal type.
- STTY Set up options for optimal control of a terminal. In so far as they are deducible from the input, these options are set automatically by LOGIN.
  - Half vs. full duplex.
  - Carriage return + line feed vs. newline.
  - Interpretation of tabs.
  - Parity.
  - Mapping of upper case to lower.
  - Raw vs. edited input.
  - Delays for tabs, newlines and carriage returns.

### 1.4. File Manipulation

- CAT Concatenate one or more files onto standard output. Particularly used for unadorned printing, for inserting data into a pipeline, and for buffering output that comes in dribs and drabs. Works on any file regardless of contents.

- CP Copy one file to another, or a set of files to a directory. Works on any file regardless of contents.
- PR Print files with title, date, and page number on every page.
  - Multicolumn output.
  - Parallel column merge of several files.
- LPR Off-line print. Spools arbitrary files to the line printer.
- CMP Compare two files and report if different.
- TAIL Print last *n* lines of input
  - May print last *n* characters, or from *n* lines or characters to end.
- SPLIT Split a large file into more manageable pieces. Occasionally necessary for editing (ED).
- DD Physical file format translator, for exchanging data with foreign systems, especially IBM 370's.
- SUM Sum the words of a file.

#### 1.5. Manipulation of Directories and File Names

- RM Remove a file. Only the name goes away if any other names are linked to the file.
  - Step through a directory deleting files interactively.
  - Delete entire directory hierarchies.
- LN "Link" another name (alias) to an existing file.
- MV Move a file or files. Used for renaming files.
- CHMOD Change permissions on one or more files. Executable by files' owner.
- CHOWN Change owner of one or more files.
- CHGRP Change group (project) to which a file belongs.
- MKDIR Make a new directory.
- RMDIR Remove a directory.
- CD Change working directory.
- FIND Prowl the directory hierarchy finding every file that meets specified criteria.
  - Criteria include:
    - name matches a given pattern,
    - creation date in given range,
    - date of last use in given range,
    - given permissions,
    - given owner,
    - given special file characteristics,
    - boolean combinations of above.
  - Any directory may be considered to be the root.
  - Perform specified command on each file found.

#### 1.6. Running of Programs

- SH The Shell, or command language interpreter.

- Supply arguments to and run any executable program.
  - Redirect standard input, standard output, and standard error files.
  - Pipes: simultaneous execution with output of one process connected to the input of another.
  - Compose compound commands using:
    - if ... then ... else,
    - case switches,
    - while loops,
    - for loops over lists,
    - break, continue and exit,
    - parentheses for grouping.
  - Initiate background processes.
  - Perform Shell programs, i.e., command scripts with substitutable arguments.
  - Construct argument lists from all file names satisfying specified patterns.
  - Take special action on traps and interrupts.
  - User-settable search path for finding commands.
  - Executes user-settable profile upon login.
  - Optionally announces presence of mail as it arrives.
  - Provides variables and parameters with default setting.
- TEST      Tests for use in Shell conditionals.
- String comparison.
  - File nature and accessibility.
  - Boolean combinations of the above.
- EXPR      String computations for calculating command arguments.
- Integer arithmetic
  - Pattern matching
- WAIT      Wait for termination of asynchronously running processes.
- READ      Read a line from terminal, for interactive Shell procedure.
- ECHO      Print remainder of command line. Useful for diagnostics or prompts in Shell programs, or for inserting data into a pipeline.
- SLEEP      Suspend execution for a specified time.
- NOHUP      Run a command immune to hanging up the terminal.
- NICE      Run a command in low (or high) priority.
- KILL      Terminate named processes.
- CRON      Schedule regular actions at specified times.
- Actions are arbitrary programs.
  - Times are conjunctions of month, day of month, day of week, hour and minute. Ranges are specifiable for each.
- AT      Schedule a one-shot action for an arbitrary time.
- TEE      Pass data between processes and divert a copy into one or more files.

### 1.7. Status Inquiries

- LS      List the names of one, several, or all files in one or more directories.
- Alphabetic or temporal sorting, up or down.
  - Optional information: size, owner, group, date last modified, date last accessed, permissions, i-node number.



- FILE Try to determine what kind of information is in a file by consulting the file system index and by reading the file itself.
- DATE Print today's date and time. Has considerable knowledge of calendric and horological peculiarities.
  - May set UNIX V32's idea of date and time.
- DF Report amount of free space on file system devices.
- DU Print a summary of total space occupied by all files in a hierarchy.
- QUOT Print summary of file space usage by user id.
- WHO Tell who's on the system.
  - List of presently logged in users, ports and times on.
  - Optional history of all logins and logouts.
- PS Report on active processes.
  - List your own or everybody's processes.
  - Tell what commands are being executed.
  - Optional status information: state and scheduling info, priority, attached terminal, what it's waiting for, size.
- IOSTAT Print statistics about system I/O activity.
- TTY Print name of your terminal.
- PWD Print name of your working directory.

### 1.8. Backup and Maintenance

- MOUNT Attach a device containing a file system to the tree of directories. Protects against nonsense arrangements.
- UMOUNT Remove the file system contained on a device from the tree of directories. Protects against removing a busy device.
- MKFS Make a new file system on a device.
- MKNOD Make an i-node (file system entry) for a special file. Special files are physical devices, virtual devices, physical memory, etc.
- TP
- TAR Manage file archives on magnetic tape or DECTape. TAR is newer.
  - Collect files into an archive.
  - Update DECTape archive by date.
  - Replace or delete DECTape files.
  - Print table of contents.
  - Retrieve from archive.
- DUMP Dump the file system stored on a specified device, selectively by date, or indiscriminately.
- RESTOR Restore a dumped file system, or selectively retrieve parts thereof.
- SU Temporarily become the super user with all the rights and privileges thereof. Requires a password.
- DCHECK

- ICHECK
- NCHECK      Check consistency of file system.
  - Print gross statistics: number of files, number of directories, number of special files, space used, space free.
  - Report duplicate use of space.
  - Retrieve lost space.
  - Report inaccessible files.
  - Check consistency of directories.
  - List names of all files.
- CLRI      Peremptorily expunge a file and its space from a file system. Used to repair damaged file systems.
- SYNC      Force all outstanding I/O on the system to completion. Used to shut down gracefully.

### 1.9. Accounting

The timing information on which the reports are based can be manually cleared or shut off completely.

- AC      Publish cumulative connect time report.
  - Connect time by user or by day.
  - For all users or for selected users.
- SA      Publish Shell accounting report. Gives usage information on each command executed.
  - Number of times used.
  - Total system time, user time and elapsed time.
  - Optional averages and percentages.
  - Sorting on various fields.

### 1.10. Communication

- MAIL      Mail a message to one or more users. Also used to read and dispose of incoming mail. The presence of mail is announced by LOGIN and optionally by SH.
  - Each message can be disposed of individually.
  - Messages can be saved in files or forwarded.
- CALENDAR      Automatic reminder service for events of today and tomorrow.
- WRITE      Establish direct terminal communication with another user.
- WALL      Write to all users.
- MMSG      Inhibit receipt of messages from WRITE and WALL.
- CU      Call up another time-sharing system.
  - Transparent interface to remote machine.
  - File transmission.
  - Take remote input from local file or put remote output into local file.
  - Remote system need not be UNIX V32.
- UUCP      UNIX to UNIX copy.
  - Automatic queuing until line becomes available and remote machine is up.
  - Copy between two remote machines.
  - Differences, mail, etc., between two machines.

### 1.11. Basic Program Development Tools

Some of these utilities are used as integral parts of the higher level languages described in section 2.

- AR      Maintain archives and libraries. Combines several files into one for housekeeping efficiency.
  - Create new archive.
  - Update archive by date.
  - Replace or delete files.
  - Print table of contents.
  - Retrieve from archive.
  
- AS      Assembler.
  - Creates object program consisting of code, normally read-only and sharable, initialized data or read-write code, uninitialized data.
  - Relocatable object code is directly executable without further transformation.
  - Object code normally includes a symbol table.
  - "Conditional jump" instructions become branches or branches plus jumps depending on distance.
  
- Library      The basic run-time library. These routines are used freely by all software.
  - Buffered character-by-character I/O.
  - Formatted input and output conversion (SCANF and PRINTF) for standard input and output, files, in-memory conversion.
  - Storage allocator.
  - Time conversions.
  - Number conversions.
  - Password encryption.
  - Quicksort.
  - Random number generator.
  - Mathematical function library, including trigonometric functions and inverses, exponential, logarithm, square root, bessel functions.
  
- ADB      Interactive debugger.
  - Postmortem dumping.
  - Examination of arbitrary files, with no limit on size.
  - Interactive breakpoint debugging with the debugger as a separate process.
  - Symbolic reference to local and global variables.
  - Stack trace for C programs
  - Output formats:
    - 1-, 2-, or 4-byte integers in octal, decimal, or hex
    - single and double floating point
    - character and string
    - disassembled machine instructions
  - Patching.
  - Searching for integer, character, or floating patterns.
  
- OD      Dump any file. Output options include any combination of octal or decimal or hex by words, octal by bytes, ASCII, opcodes, hexadecimal.
  - Range of dumping is controllable.
  
- LD      Link edit. Combine relocatable object files. Insert required routines from specified libraries.

- LORDER**       Resulting code is sharable by default.
- LORDER**      Places object file names in proper order for loading, so that files depending on others come after them.
- NM**            Print the namelist (symbol table) of an object program. Provides control over the style and order of names that are printed.
- SIZE**           Report the memory requirements of one or more object files.
- STRIP**          Remove the relocation and symbol table information from an object file to save space.
- TIME**           Run a command and report timing information on it.
- PROF**           Construct a profile of time spent per routine from statistics gathered by time-sampling the execution of a program.
  - Subroutine call frequency and average times for C programs.
- MAKE**           Controls creation of large programs. Uses a control file specifying source file dependencies to make new version; uses time last changed to deduce minimum amount of work necessary.
  - Knows about CC, YACC, LEX, etc.

### 1.12. UNIX V32 Programmer's Manual

- Manual**        Machine-readable version of the UNIX V32 Programmer's Manual.
  - System overview.
  - All commands.
  - All system calls.
  - All subroutines in C and assembler libraries.
  - All devices and other special files.
  - Formats of file system and kinds of files known to system software.
  - Boot and maintenance procedures.
- MAN**            Print specified manual section on your terminal.

### 1.13. Computer-Aided Instruction

- LEARN**        A program for interpreting CAI scripts, plus scripts for learning about UNIX V32 by using it.
  - Scripts for basic files and commands, editor, advanced files and commands, EQN, MS macros, C programming language.

## 2. Languages

### 2.1. The C Language

- CC**            Compile and/or link edit programs in the C language. The UNIX V32 operating system, most of the subsystems and C itself are written in C. For a full description of C, read *The C Programming Language*, Brian W. Kernighan and Dennis M. Ritchie, Prentice-Hall, 1978.
  - General purpose language designed for structured programming.
  - Data types include character, integer, float, double, pointers to all types, functions returning above types, arrays of all types, structures and unions of all types.

- Operations intended to give machine-independent control of full machine facility, including to-memory operations and pointer arithmetic.
- Macro preprocessor for parameterized code and inclusion of standard files.
- All procedures recursive, with parameters by value.
- Machine-independent pointer manipulation.
- Object code uses full addressing capability of the VAX-11.
- Runtime library gives access to all system facilities.
- Definable data types.
- Block structure

□ LINT

Verifier for C programs. Reports questionable or nonportable usage such as:  
Mismatched data declarations and procedure interfaces.  
Nonportable type conversions.  
Unused variables, unreachable code, no-effect operations.  
Mistyped pointers.  
Obsolete syntax.

○ Full cross-module checking of separately compiled programs.

□ CB

A beautifier for C programs. Does proper indentation and placement of braces.

## 2.2. Fortran

□ F77

A full compiler for ANSI Standard Fortran 77.

○ Compatible with C and supporting tools at object level.

○ Optional source compatibility with Fortran 66.

○ Free format source.

○ Optional subscript-range checking, detection of uninitialized variables.

○ All widths of arithmetic: 2- and 4-byte integer; 4- and 8-byte real; 8- and 16-byte complex.

□ RATFOR

Ratfor adds rational control structure à la C to Fortran.

○ Compound statements.

○ If-else, do, for, while, repeat-until, break, next statements.

○ Symbolic constants.

○ File insertion.

○ Free format source

○ Translation of relationals like  $>$ ,  $>=$ .

○ Produces genuine Fortran to carry away.

○ May be used with F77.

□ STRUCT

Converts ordinary ugly Fortran into structured Fortran (i.e., Ratfor), using statement grouping, if-else, while, for, repeat-until.

## 2.3. Other Algorithmic Languages

□ DC

Interactive programmable desk calculator. Has named storage locations as well as conventional stack for holding integers or programs.

○ Unlimited precision decimal arithmetic.

○ Appropriate treatment of decimal fractions.

○ Arbitrary input and output radices, in particular binary, octal, decimal and hexadecimal.

○ Reverse Polish operators:

+ - \* /

remainder, power, square root,

load, store, duplicate, clear,

print, enter program text, execute.

- BC      A C-like interactive interface to the desk calculator DC.
  - All the capabilities of DC with a high-level syntax.
  - Arrays and recursive functions.
  - Immediate evaluation of expressions and evaluation of functions upon call.
  - Arbitrary precision elementary functions: exp, sin, cos, atan.
  - Go-to-less programming.

#### 2.4. Macroprocessing

- M4      A general purpose macroprocessor.
  - Stream-oriented, recognizes macros anywhere in text.
  - Syntax fits with functional syntax of most higher-level languages.
  - Can evaluate integer arithmetic expressions.

#### 2.5. Compiler-compilers

- YACC      An LR(1)-based compiler writing system. During execution of resulting parsers, arbitrary C functions may be called to do code generation or semantic actions.
  - BNF syntax specifications.
  - Precedence relations.
  - Accepts formally ambiguous grammars with non-BNF resolution rules.
- LEX      Generator of lexical analyzers. Arbitrary C functions may be called upon isolation of each lexical token.
  - Full regular expression, plus left and right context dependence.
  - Resulting lexical analysers interface cleanly with YACC parsers.

### 3. Text Processing

#### 3.1. Document Preparation

- ED      Interactive context editor. Random access to all lines of a file.
  - Find lines by number or pattern. Patterns may include: specified characters, don't care characters, choices among characters, repetitions of these constructs, beginning of line, end of line.
  - Add, delete, change, copy, move or join lines.
  - Permute or split contents of a line.
  - Replace one or all instances of a pattern within a line.
  - Combine or split files.
  - Escape to Shell (command language) during editing.
  - Do any of above operations on every pattern-selected line in a given range.
  - Optional encryption for extra security.
- PTX      Make a permuted (key word in context) index.
- SPELL      Look for spelling errors by comparing each word in a document against a word list.
  - 25,000-word list includes proper names.
  - Handles common prefixes and suffixes.
  - Collects words to help tailor local spelling lists.
- LOOK      Search for words in dictionary that begin with specified prefix.

- CRYPT      Encrypt and decrypt files for security.

### 3.2. Document Formatting

- TROFF

- NROFF      Advanced typesetting. TROFF drives a Graphic Systems phototypesetter; NROFF drives ascii terminals of all types. This summary was typeset using TROFF. TROFF and NROFF are capable of elaborate feats of formatting, when appropriately programmed. TROFF and NROFF accept the same input language.

- Completely definable page format keyed to dynamically planted "interrupts" at specified lines.
- Maintains several separately definable typesetting environments (e.g., one for body text, one for footnotes, and one for unusually elaborate headings).
- Arbitrary number of output pools can be combined at will.
- Macros with substitutable arguments, and macros invocable in mid-line.
- Computation and printing of numerical quantities.
- Conditional execution of macros.
- Tabular layout facility.
- Positions expressible in inches, centimeters, ems, points, machine units or arithmetic combinations thereof.
- Access to character-width computation for unusually difficult layout problems.
- Overstrikes, built-up brackets, horizontal and vertical line drawing.
- Dynamic relative or absolute positioning and size selection, globally or at the character level.
- Can exploit the characteristics of the terminal being used, for approximating special characters, reverse motions, proportional spacing, etc.

The Graphic Systems typesetter has a vocabulary of several 102-character fonts (4 simultaneously) in 15 sizes. TROFF provides terminal output for rough sampling of the product.

NROFF will produce multicolumn output on terminals capable of reverse line feed, or through the postprocessor COL.

High programming skill is required to exploit the formatting capabilities of TROFF and NROFF, although unskilled personnel can easily be trained to enter documents according to canned formats such as those provided by MS, below. TROFF and EQN are essentially identical to NROFF and NEQN so it is usually possible to define interchangeable formats to produce approximate proof copy on terminals before actual typesetting. The preprocessors MS, TBL, and REFER are fully compatible with TROFF and NROFF.

- MS      A standardized manuscript layout package for use with NROFF/TROFF. This document was formatted with MS.
  - Page numbers and draft dates.
  - Automatically numbered subheads.
  - Footnotes.
  - Single or double column.
  - Paragraphing, display and indentation.
  - Numbered equations.
- EQN      A mathematical typesetting preprocessor for TROFF. Translates easily readable formulas, either in-line or displayed, into detailed typesetting instructions. Formulas are written in a style like this:

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2$$

which produces:

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2$$

- Automatic calculation of size changes for subscripts, sub-subscripts, etc.
  - Full vocabulary of Greek letters and special symbols, such as 'gamma', 'GAMMA', 'integral'.
  - Automatic calculation of large bracket sizes.
  - Vertical "piling" of formulae for matrices, conditional alternatives, etc.
  - Integrals, sums, etc., with arbitrarily complex limits.
  - Diacriticals: dots, double dots, hats, bars, etc.
  - Easily learned by nonprogrammers and mathematical typists.
- NEQN      A version of EQN for NROFF; accepts the same input language. Prepares formulas for display on any terminal that NROFF knows about, for example, those based on Diablo printing mechanism.
- TBL      A preprocessor for NROFF/TROFF that translates simple descriptions of table layouts and contents into detailed typesetting instructions.
- Computes column widths.
  - Handles left- and right-justified columns, centered columns and decimal-point alignment.
  - Places column titles.
  - Table entries can be text, which is adjusted to fit.
  - Can box all or parts of table.
- REFER      Fills in bibliographic citations in a document from a data base (not supplied).
- References may be printed in any style, as they occur or collected at the end.
  - May be numbered sequentially, by name of author, etc.
- TC      Simulate Graphic Systems typesetter on Tektronix 4014 scope. Useful for checking TROFF page layout before typesetting.
- COL      Canonicalize files with reverse line feeds for one-pass printing.
- DEROFF      Remove all TROFF commands from input.
- CHECKEQ      Check document for possible errors in EQN usage.

#### 4. Information Handling

- SORT      Sort or merge ASCII files line-by-line. No limit on input size.
- Sort up or down.
  - Sort lexicographically or on numeric key.
  - Multiple keys located by delimiters or by character position.
  - May sort upper case together with lower into dictionary order.
  - Optionally suppress duplicate data.
- TSORT      Topological sort — converts a partial order into a total order.
- UNIQ      Collapse successive duplicate lines in a file into one line.
- Publish lines that were originally unique, duplicated, or both.
  - May give redundancy count for each line.
- TR      Do one-to-one character translation according to an arbitrary code.



- May coalesce selected repeated characters.
  - May delete selected characters.
- DIFF Report line changes, additions and deletions necessary to bring two files into agreement.
  - May produce an editor script to convert one file into another.
  - A variant compares two new versions against one old one.
- COMM Identify common lines in two sorted files. Output in up to 3 columns shows lines present in first file only, present in both, and/or present in second only.
- JOIN Combine two files by joining records that have identical keys.
- GREP Print all lines in a file that satisfy a pattern as used in the editor ED.
  - May print all lines that fail to match.
  - May print count of hits.
  - May print first hit in each file.
- LOOK Binary search in sorted file for lines with specified prefix.
- WC Count the lines, "words" (blank-separated strings) and characters in a file.
- SED Stream-oriented version of ED. Can perform a sequence of editing operations on each line of an input stream of unbounded length.
  - Lines may be selected by address or range of addresses.
  - Control flow and conditional testing.
  - Multiple output streams.
  - Multi-line capability.
- AWK Pattern scanning and processing language. Searches input for patterns, and performs actions on each line of input that satisfies the pattern.
  - Patterns include regular expressions, arithmetic and lexicographic conditions, boolean combinations and ranges of these.
  - Data treated as string or numeric as appropriate.
  - Can break input into fields; fields are variables.
  - Variables and arrays (with non-numeric subscripts).
  - Full set of arithmetic operators and control flow.
  - Multiple output streams to files and pipes.
  - Output can be formatted as desired.
  - Multi-line capabilities.

## 5. Graphics

The programs in this section are predominantly intended for use with Tektronix 4014 storage scopes.

- GRAPH Prepares a graph of a set of input numbers.
  - Input scaled to fit standard plotting area.
  - Abscissae may be supplied automatically.
  - Graph may be labeled.
  - Control over grid style, line style, graph orientation, etc.
- SPLINE Provides a smooth curve through a set of points intended for GRAPH.
- PLOT A set of filters for printing graphs produced by GRAPH and other programs on various terminals. Filters provided for 4014, DASI terminals, Versatec printer/plotter.

6. Novelties, Games, and Things That Didn't Fit Anywhere Else

- BACKGAMMON      A player of modest accomplishment.
- BCD                Converts ascii to card-image form.
  
- CAL                Print a calendar of specified month and year.
- CHING             The *I Ching*. Place your own interpretation on the output.
- FORTUNE          Presents a random fortune cookie on each invocation. Limited jar of cookies included.
- UNITS             Convert amounts between different scales of measurement. Knows hundreds of units. For example, how many km/sec is a parsec/megayear?
- ARITHMETIC      Speed and accuracy test for number facts.
- QUIZ              Test your knowledge of Shakespeare, Presidents, capitals, etc.
- WUMP             Hunt the wumpus, thrilling search in a dangerous cave.
- HANGMAN         Word-guessing game. Uses a dictionary supplied with SPELL.
- FISH              Children's card-guessing game.

SOFTWARE TOOLS and UNIX Users Group  
Conference Report

by  
David M. Phillips

### Introduction

The conference ran on June 19-23 and was held at the University of Toronto. The campus was most attractive, combining grand old (1826) architecture with outstanding modern work using many interesting angular shapes of different scale. The weather was cool and clear.

The first day was a separate one-day conference on SOFTWARE TOOLS. Dennis Hall and Debbie Scherrer of Lawrence Berkeley Labs (LBL) organized the meeting. They introduced the idea at the end of the winter Unix meeting. They conducted a survey, organized a meeting, and announced it shortly before the summer conference. Even so, they attracted 96 attendees; 46 came for the tools meeting alone.

The popularity of the tools is spreading far and wide. People are using them to provide a set of powerful and portable system-interface / text processing capabilities. They also provide a common user-interface across machines and operating systems!

The UNIX conference ran three and a half days. It was organized by the Computer Science Research Group (CSRG) at U. Toronto. A wealth and vast array of software (mostly for free) was offered throughout the proceedings. Unfortunately, CSRG wasn't prepared to deal with the volume of incoming an[76E6{3f};s We'll have to spend some time and effort to get it from a few regional sites selected to get tapes.

The three areas of most significant developments seemed to me to be: languages (Pascal on Unix and C on anything else); system improvements (performance, reliability, integrity); and porting Unix to other machines. But there was a wide range of additional interests expressed. The Unix conferences have always been fun for me, because it is clear that people enjoy

using their systems. If they're not tinkering with Unix to improve in capabilities or performance, then they're implementing interesting applications on it in an economical fashion.

## SOFTWARE TOOLS

Brian Kernighan -- Bell Labs

Described STRUCT, a program to reformat Fortran programs into Ratfor. This C program is distributed with version 7 Unix.

Described EFL, extended fortran language, being worked on at labs. True compiler, but source code can be converted to standard Fortran. Extensions beyond Ratfor. Big, slow, written in C.

David Hanson -- U. Arizona

Projects: enhanced Ratfor, Ratsno, I/O systems, editors, portable directory systems, new tools, and the ICON language (working with Ralph Griswold on this; see recent SIGPLAN).

Ratfor: syntax sugar, change lookup to hashing, optimize gotos.

High performance, flexible I/O for CDC and DEC-10. File system interface conforms to what user expects on a particular machine.

Chris Fraser developed a display editor, with a small screen manager and ed as a back-end. Described how it has been applied to other processes to produce display-oriented system interaction.

The portable directory system implements the (Unix) hierarchical directory structure on top of a standard operating system file structure. Provides all directory manipulations primitives. In Ratfor. No impact on file efficiency after open or create are done.

Many tool enhancements: edit, macro (added infix expression evaluator), archive, sort, roff.

New tools: ta (tape archive); nar (faster than maklib).

Arizona made change (about 7 lines of monitor code) so instead of saying "R foo", can simply say "foo".

ICON: intended as successor to SNOBOL4 and SL5. A different string processing language. Written in Ratfor!

Doug Comer -- Purdue

Used a profiler and determined LOOKUP accounted for 60% of execution time of Ratfor. Rewrote it in Pascal to use hash codes.

Made include and define separate statements recognized by parser at upper-level.

Call it MOUSE4.

Limited to line-oriented I/O.

Using native characters, instead of mapping to internal code, saves considerably.

Allen Akin -- Georgia Tech.

Implemented complete tools system on new Prime 400 computer. Used PRIMOS operating system as base level. A complete subsystem.

PRIMOS doesn't support multiple processes per user, but their shell gets around it. Requires recursive Fortran.

Have a screen editor like Fraser's.

Put a lot of work into shell. Added concept of "funnel", multiple numbered standard I/O files that can be redirected. Added iteration for filenames with small differences. E.g. file(1 2 3) for file1 file2 file3.

[Wally has handout.]

Dennis Hall -- LBL

Implemented in Ratfor a vanilla Unix shell; plus scripts can be filters; plus program input can be included in scripts; minus if, goto, and filename wildcards (boo!).

Want to support networks with the shell; commands of the form: "A/filename" where A is hostname.

Joe Sventek -- LBL

Research on user-machine interaction in heterogenous network.

Plan is to maintain an open connection between machines. Pass environment and commands through this. For a service, establish a direct link to the server on the remote host (as with Arpanet).

Extended filenames for networks: /@host/"rest of path"/file.

Primitives extended to net.

Bringing up DECNET under Unix. Jim Hamilton at DEC wrote original package.

Robert Munn -- U. Maryland

Developing RATMAC, includes Ratfor and Macro. Responsible for distributing a system of 90,000 lines of code. It must be extremely portable and trouble-free, because it is delivered to crystallographers (naive computer users) in many countries using many different computers. The macros convey machine-dependent information.

Robert Gordon -- PRIME Research

Confession that they use the tools because the usual PRIMOS interface is so poor. Developing a 10Mhz network.

Fourty survey responses returned. To get the survey:

Debbie Scherrer  
Computer Science and Mathematics Department  
Lawrence Berkeley Laboratories  
Berkeley, CA 94720  
(SCHERRER@LBL-UNIX)

LBL awarded Brian a "godfather" Software Tools T-shirt. Shows a rat with tail forming the number 4, composing software at a terminal. LBL has a Ratfor mail system of the "mh" genre.

Distribution: discussion of schemes. About dozen people with automatic call units. About a dozen with Arpanet access. Develop and use a portable tape archiver.

Newsletter:

Todd Kushner  
Telenet - GTE  
8330 Old Courthouse Rd.  
Vienna, VA 22180  
703/827-9200

Neil Groundwater, Analytic Disciplines, Vienna, VA offered info on driving an ACU.

UNIX Users Group

Ron Baeker was chairman. He's a professor of C.S. and director of Dynamic Graphics Project.

Professor Andrew Tannenbaum -- U. Amsterdam -- Pascal

A portable Pascal; uses EM1, an intermediate language. Can produce code to be interpreted, or send thru an optimizer and produce executable code.

Put on distribution tape the complete Pascal system with documentation, and fixed Unix assembler and loader (to accomodate large files). (Others who've received the distribution say it's really fine. Excellent documentation.)

Fully compatible with British standard. A full implementation. get(input) is lowest level function required of operating system. Restrictions: range of cases must be < 256; max. string length is 72 chars; max set size 256. Features: separate compilation; mark/release; Unix style strings; external procedures; {} comments nest.

Supports assertions; source line numbers are kept for debug; if debug conditional; record of procedure entry/exit; warnings for variables not set or used; variables initialized to undefined.

Berkeley Pascal is twice as fast to compile (is even faster than cc), but is an interpreter. Slower than compiler.

See CACM Mar. 78 for info on EM1 machine.  
 Future editions will be available at cost from:  
 Bruce Knobe  
 Intermetrics Inc.  
 701 Concord Ave.  
 Cambridge, MA 02138

Jim Cordy -- U. Toronto -- Euclid  
 Described Euclid, a Pascal-based prog. language for secure (proveable) systems. A compiler commissioned in 1976. In Aug. 77 a joint venture was begun between I.P. Sharp and U. Toronto.  
 Jan. 78 - transliterator from "small" Euclid to C.  
 Jun. 79 - translator for full Euclid except coder, last pass.  
 Fall 79 - full Euclid. Distribution from IPS with maintenance for some cost.  
 Big - about 60,000 lines; slow because of semantic checks.

Richard Bolocca -- U. Illinois -- Path (concurrent) Pascal  
 Superset of Pascal, including concurrency and data encapsulation. Used in teaching and real-time programming (for NASA space shuttle).  
 Can create processes dynamically. Runs under Unix or stand-alone. Run-time system about 1,000 lines of code. Will run on LSI-11 with 16K. Two pass; requires separate I/O space.  
 Versions for Z80, all 11s, (and coming: Series/1, Prime 500 and above). Distributed in Fall at nominal cost.

Bill Joy -- U. C. Berkeley -- languages for VAX  
 Pascal interpreter moved; compiler being worked on.  
 SCALD -- electronics design package moved. Developed at Stanford, written in Pascal, used to design Series/1.  
 LISP written in C. Intended to reach MACLISP or LISP machine capability. Fully interpreted now - slow. Will run 2/5 of MACSYMA. Working on full compiler.  
 APL interpreter. Working on compiler.  
 MODULA from York moved.  
 RIGEL - a typed, relational data-base language (see last SIGMOD).  
 Implementing MACSYMA.  
 Developing "3", a modified FORTH.  
 Developing STAPL, a structured programming and applications language.  
 TERMCAP - data base describing how to drive terminals (have 80 now).  
 C shell - a shell command language like C.  
 Developing kernel mods to allow large text programs to run on small machines.

Keith David -- Teletype -- C cross-compiler

Z-80 C cross-compiler that runs under PWB. Developed for Teletype by Interactive Systems Corp. Available from ISC (but probably expensive; internal to Bell it's \$5,000!). Generates 42% more code than for an 11. A 3Mhz Z80 runs 1/8 the speed of an 11/70.

Brian Kernighan -- Bell Labs -- version 7 Unix

Features: big files (2<sup>30</sup> bytes); portable C compiler; LINT; standard I/O library; Fortran 77; make (build up-to-date systems); lex (lexical analyzer generator); awk (simple pattern matching language); sed (stream editor); new shell supporting C programming constructs; learn (interactive CAI system that allows student to intersperse instruction with actually using the machine); adb (symbolic, dynamic C debugger); uucp (Unix to Unix file transfer); refer (inverted indexes); troff, eqn, tbl (typesetter, equation, and table formatter processors).

New shell. No "goto". Has "for", "case", "if-then-else", "while", "until"; redirect error output; capture command output; all commands have meaningful return values; settable path; trap handling. Syntax is verbose because developed by Algol-68 fan.

Portable C. More integer types; unions; bit fields; typedefs; explicit type coercions; block structure for declarations; external static storage; initializations with declarations; structure assignment; enumeration data type; proper syntax for += (instead of =+).

lint - checks C programs for type violations, portability, problems, probable errors, suspicious code, bad coding style. Uses many heuristics.

make - constructs programs according to script of dependencies.

Fortran 77 - complete language plus a few extensions. standard Fortran I/O plus Unix I/O. Uses C internal language and back-end.

struct - converts Fortran programs to structured Ratfor.

awk - pattern scanning and processing.

learn - powerful script interpretation language.

New crypt algorithm. Implements DES plus a few goodies.

Tom London -- Bell -- UNIX/32V

Unix v.7 ported to the VAX.

Same as 11 version: file system, shell, commands, languages, system interface.

Differences: rePresentations (word size, data encoding, alignment, traps, addressing, loader info), address space, speed.

To use larger address space: scatter loading - no page of a process need be loaded contiguously; swap out only as much space as needed; a user can use all of memory except that used by system.

Ported by two people in five months (long days).



Prof. Desautels -- Wisconsin U. -- use of VAX  
[nothing noteworthy]

Howard Weiss -- DoD -- KSOS (secure kernel)

Ford Aerospace is now working on the coding. Plan to be completed by 1st of year. Contract is for it to run no more than twice as slow as v.6 Unix.

[Dennis Mumaugh reports that Ford Aerospace has done work with the ACC UMC-80 communications microcomputer.]

Al Arms -- Western Electric -- Licensing

	initial cpu	add'l cpu	binary	timesharing service
MINI	\$12,000	\$4,000		
UNIX	\$20,000	\$8,700	\$8,400	10% of net sales
PWB	\$30,000	\$10,000	\$12,000	10%
v.7	\$28,000	\$9,400	\$11,700	10%
32V	\$40,000	\$15,000	\$18,000	10%

An educational administrative license is 1/3 of the commercial fee, across the board.

PWB v.2 has just been released internal to Bell. The only thing PWB has over v.7 is scos (source code control system) and rje [which people say is a real loser].

New Unix {licenses ban teaching of Unix Internals} but documentation is in the public domain.

MERT, LSX, and Interdata Unices will not be released.

About 550 educational, 250 commercial licensees.

Lou Katz -- Columbia U. -- Usenix Association

Papers have been signed to create "Usenix Association", a not-for-profit association operating in New York.

Officers: Lou Katz - president; Lew Law - vice-president; Mel Ferentz - treasurer; Armand Gazes, Peter Weiner, and Mars Gralia board members.

Only license holders can vote. You can purchase as many memberships as you have licenses.

Classes of membership: commercial institution - \$300; educational institution - \$100; individual - \$12. These all get the newsletter [when/if ever it appears]. Institutional members get a software distribution tape. Non-voting institutional membership (for Bell components) also available.

Conf [Plan to start copying distribution tape around 1st of year. Membership forms to be sent July 1.

Usenix Association  
Box 8  
Rockefeller University  
1230 York Ave.  
New York, N.Y.

CA Canada DECUS UNIX SIG was recently formed. DECUS didn't care for the idea, but allowed it.]

Mark Krieger -- Whitesmiths -- C compilers

PDP-11 C compiler for Unix, RSX-11M, RT-11, IAS, RSTS.

Portable library: alloc, free; char, line, and formatted I/O; strings. Some routine names and calling sequences are incompatible with standard library.

8080 C cross-compiler. Full v.7 C. Code generated is 50% larger than on an 11. Uses "a natural" assembly language, really 4th pass of compiler. Interface to CPM, ISIS-II, CDOS. Includes machine library. Has 16, 32, 64-bit math.

Avail. July 79: "a natural" assembler, loader, librarian for 8080. OS takes 7k; compiler largest pass is 47k (can compile itself in that space). Runs on PDP-11 or 8080. Full C compiler under CPM.

Avail. Fall 79: IDRIS operating system on LSI-11. Functionally equivalent to Unix, including system calls, v.6 file system, shells, and pipes. But developed solely by Whitesmiths, so no Western Elec. licensing.

Early 1980: VAX native-mode C compiler; C compiler on a 16-bit micro.

David Lilienfeld -- Johns Hopkins -- YASL

Yet Another Statistical Language. Intended to be a superset of C which understands matrices. Will have regular, virtual (on disk), and dynamic matrices. Based on LINPAK and EISPAK. Expect to have first version by end of summer. Done with yacc. Richard Bartels is a co-investigator in the project.

Dennis Mumaugh -- NSA -- CORE graphics system

Implemented the SIGGRAPH CORE graphics system in C. The system is likely not well written because done by C amateurs. The distribution includes one driver for a Genisco system. [SIGGRAPH sells a tape with the CORE manual; see Dec. 78 SIGGRAPH Report.]

Martin Tuori -- DCIEM -- Research Display System

Interested in audio-visual communication, enhancing information presentation. Use a VG display system; Norpak color display, PDP-11/34.

Tom Duff -- NYIT -- color graphics

New York Institute of Technology

Make cartoons and special TV visual effects. Use up to twelve frame buffers for color work (a 512x512x8 bit memory). Bad report on reliability of Genisco frame buffers. Gave a fantastic 20 minute videotape presentation of work done at the lab. But none of their software is available -- all proprietary.

Mike Muus -- Johns Hopkins -- terminal independent graphics  
Available now. Produces a univercml position file. Have  
interpreters for incremental, vector, and raster devices.  
Includes basic CalComp stuff; i.e. axis, line, symbol, and so  
on.

Supports HP, Tektronix, HI, Versatec, Diablo, and Ramtek  
devices. Takes an afternoon to add support for a new device.

Request it from:

Dr. Henrikson  
Ballistics Research Lab  
Aberdeen Proving Grounds  
Aberdeen, MD

G. Toth -- Johns Hopkins -- VERSET

A Versatec phototypesetter emulator. Fast, small,  
efficient. Uses fonts that duplicate the GSI fonts. Wants 200  
dots/inch V. \$500 for educational (includes source).

David Macfarlane -- Bell Northern Software Research (BNSR) --  
Office automation

[Nothing noteworthy]

Bill Buxton -- U. Toronto -- music concert

Gave a music concert. Uses an LSI-11 for D/A conversion of a  
stored musical score. Then performer can "conduct" performance  
using a graphics tablet to interact with choices on a terminal  
screen.

Lynn Brock -- Computer Corp. of America -- software registry

An Arpanet contractor; they propose to produce a registry of  
Unix software and C compilers. To include free and commercial  
offerings. Intend to create a data-base and query system. Will  
have an 800 phone number, Arpanet, and possibly Telenet access.  
Will have users create/update their own entries.

John Kornatowski -- U. Toronto -- MRS data-base system

Developed a small relational data-base system. Intended to  
run on micros. Running on Unix now. For small data-bases, up  
to about 10Mb. Fast, easy, simple. Runs under Unix and Mini-U.  
on all 11s.

Runs on LSI-11 with CRT, dual-density floppy, and full  
memory.

Programs require only 25k, 300 blocks disk storage.

Available for \$200 from:

MRS Distribution Manager  
Computer Science Research Group  
U. of Toronto  
Toronto, Ontario, Canada  
M5S 1A1

Bob Hudyma -- U. Toronto -- adapting MINI-UNIX for LSI-11  
Made MINI-U run on LSI-11/1 and 11/2. Use EIS chip, 28k words memory, 2 AED floppys. Runs single user; could run 2-3 users with a fast disk for swapping. 18k words user memory.

Usual software. Pipes implemented with tmp files. No sequential contiguous files. A 200 line C program compiles in three and a half minutes.

How it was done: write device driver for disk; remove references to processor status and replace by subroutine calls (put subr. in mch.s); eliminate clock references in main.c; modify init.c for single user (eliminate switch register); and add a swap flag to prevent swapping if swap in progress.

System is distributed with MRS.

Dan Geelan -- Bell Production Systems -- performance improvement  
Working to improve performance of an 11/70-based Unix system.

Implemented: swap device driver (to allow swapping to several devices); swapmap and coremap exceptions; efficient raw mode for tty and dh (uses a wakeup char); RWP06 drivers; disk space assignments based on system use; rewrote LP11 driver.

In design: RWS04 replacement - mass memory on massbus; smart DH-11 via UMC-80.

He recommends this priority for performance improvements: add 256k bytes of memory and a floating-pt. unit; add a swap device; add another 256k memory; separate disk controllers (put TE16 tape system on Unibus if necessary to make room for disk controller); add a smart data handler; add a cpu and split up applications.

Eric Ostrum -- CMU NCSL -- real-time

Neurological Control Systems Lab. Diagnose brain disease by gait and eye movement analysis.

Has drivers for AD11-K, AA11-K, KW11-K, DR11-C. Some graphics software. Available for \$50.

Bill Croft -- Purdue -- networking

Use DMC-11 and DA-11B (1 MHz). Connected shell to communicate across systems. Simple virtual terminal protocol. Runs programs on remote host with local I/O. So, for instance, to transfer a file, simple cat it on remote host, redirected to a file on local host. Net looks like device drivers with pool of devs for each host. Uses sockets similar to Arpanet. C-callable function library of net stuff. Requires extensive kernel mods. Ad hoc development. Will be undergoing changes.

William Lindermann -- NYIT -- networking

Similar to above, but simpler. Use an 11/34 in center of star as front-end to terminals and other systems.

Mike Tilson -- Human Computing Resources -- RT-11 emulator

RT11 emulator running under Unix. A complete system, except for keyboard monitoring, which is available separately at extra cost. Runs RT-11 binaries without change. Version 3B is single-user. Files in Unix format. Educational license is \$810. [Wally has manual.]

Mark Krieger -- Whitesmiths -- more on IDRIS

IDRIS is a Unix look-alike system. Developing versions for LSI-11 and Chromemco Z-80. Multi-job LSI-11 out around Jan. 1980. A single job version for the 8080 will likely follow. Pricing is unusual -- 50 cents per line of source code for source; 5 cents per for binary license. So IDRIS will be about \$1500 for source.

Dennis Hall -- LBL -- software tools

Brief report on SOFTWARE TOOLS meeting.

Alfred Whaley -- U. Illinois -- multiple address space

Have developed a set of tools to manage memory allocation. Mentioned that problem arose because they've developed a multiplexor box that drives very many devices.

Ian Johnstone -- U. New South Wales -- 1100 users

Implemented: kernel overlays; shared data (read/write text segments); optimized disk drivers.

Using 11/70, 8 DZs, two-80MB disks; 13,000 connect hours/month. Mostly using Berkeley's Pascal.

Changes to support many users: binary password file, with much extra info; error logging and recovery; disk compactor; system printf; large file system dump/restore; new, faster \*check; profiling tools; many kernel changes.

Results: response is ok with 44 users; it takes 32 seconds to compile and execute a two second Pascal program.

Software is included on the conference tape.

George Goble -- Purdue -- 1400 users

Same problem as above; much of the same work. Also: dual swap devices; auto file system recovery; fast tty I/O.

Very large hardware configuration. Has a lot of experience dealing with bus problems. Mentioned that DEC district support has bus analyzer/testers. Says to check all power supplies for 5.05 volt logic during PM.

Mike O'Dell at U. Oklahoma is using Purdue system with good success.

Can get software by writing George.

Walter Lazear -- AFDSC -- bad blocks

Have done various software work. Most notable are changes to deal with bad blocks on disk. Made changes to kernel (small);

mkfs; \*check; df; clri; dump; and restor. Mark bad blocks when initializing a disk, but note mkfs takes up to 50 minutes on a 56k-block device. Raw devices are not protected, and there's no dynamic detection of bad blocks. Requires no changes to disk drivers. [INMAC (formerly MCA) sells packs guaranteed error free for three years.]

Also fixed up exit codes in commands, added a command synopsis feature and a consistent user interface. Their shell has an "open" edit mode for the previous command!

Software available on request to:

Walter Lazear or Charles Muir  
Air Force Data Services Center  
The Pentagon, room 1D988  
Washington, D.C.

202/695-6161 (also check for current Arpanet address)

Carl Howe -- BBN -- Microprogrammable Building Block

Described the BBN Microprogrammable Building Block, a general-purpose computer. It's a table-top computer. Intended to have the power of an 11/70 at the price of an 11/34. BBN uses it now to emulate a Honeywell 316 at 1.5 times the speed.

Specs: 16 or 20 bit words; 135 nsec cycle time for 20 bit word; 2-8k RAM and .5-8k PROM microcode memory; 32-128k main memory; 1024 high-speed microprocessor registers; one minute battery backup; 300 watts at 110 VAC; Schottky technology.

Should be available commercially, delivered with a Unix operating system, about Summer 1980. For information contact:

The BBN Computer Company  
Bolt Beranek and Newman Inc.  
50 Moulton Street  
Cambridge, MA 02138  
617/491-1850 x3642

Hale Pierson -- RLG -- Unix on a Univac

Ported Unix to the Univac V77. Complete system is up, but not all the utility software. Many difficulties because the V77 is a word (not byte) addressable machine. Also, there's no integer compare (!) or variable shift. Code generation is about 20-30% larger than for an 11. Required 5-6 months by one person to port the C compiler.

R. N. Jesse -- Johns Hopkins -- usage accounting

Use a user information file with a one-to-one line correspondence to /etc/passwd. It keeps accounting information, limits, permissions, and so on.

They only use a logged out quota. Do performance, load, and error monitoring. Only the owners of files are charged for disk space. The software is available and includes the high-performance mods. Not yet ready. Will be about \$100.

Robert N. Jesse

Westinghouse Computer Lab  
120 Barton Hall  
The Johns Hopkins U.  
Baltimore, MD. 21218  
301/338-7021

Mike Muus -- Johns Hopkins -- high-performance Unix  
Have implemented a /dev/data to hold and display performance information. Many tty mods. Security mods. Allow special files and set-UID files only on root. Memory test and reconfiguration on booting. Integrity changes (check all limits and "can't happen" conditions). Performance mods: change all process lists to queues (there are 19). Improved scheduler priority evaluation.

Steve Bellovein -- Tufts U. -- porting Unix to IBM 370  
Planning to port Unix operating system to an IBM 370 to run under MVS.

Don Sherry -- Bradley U. -- use of Unix  
5,000 students. Do data acquisition with micros, then send it to Unix. Use Johns Hopkins MINI-UNIX on an LSI-11 to drive a phototypesetter.

Mike O'Dell -- U. Oklahoma -- Goble's system  
Use G. Goble's system from Purdue. Found it easy to bring up. Made some changes in the way accounting done. Have various daemons to gather different information. 5-700 users.

Robert Pike -- Caltech -- ultimate QED  
David Tilbrook -- Bell Northern Software Research  
Put back in many of the qed features that were removed to make ed. Considerable enhancements: macros, multiple files, programmable.

George Pajari -- GTE -- tuning PWB  
Have a query program to report table usage levels, so table sizes can be set more intelligently. Can get a copy of the report from:

Paul Hart  
GTE Automatic Electric (Canada) Ltd.  
100 Stronger Blvd.  
Brockville, Ontario, Canada  
K6V 5W8

Mark Pierson -- Yourdan -- screen editor  
Has developed a simple screen editor. Front-ends the standard ed and allows one to escape to ed. The qed people said they'd include this in their distribution.

Robert Pike  
CSRG  
121 St. Joseph Street  
Toronto, Ontario, Canada  
M5S 1A1

Miscellaneous

Rand has a new distribution tape with improvements to old stuff:  
RITA, MS, MH, NED, and accounting. \$115.

Rick Kiesig  
Rand Corp.  
1700 Main St.  
Santa Monica, CA 90406

Unix T-shirts. Picture of a PDP-11 with daemons, pipes, and  
forks. Size S, M, L, XL. Trim red or white. \$6.50 plus \$0.50  
postage.

TRI-J Color Print Co.  
1616 E. Waverly Dr.  
Arlington Heights, IL 60004

The Winter Unix Users Group Meeting will be hosted by John  
Donnelly at NCAR, Boulder, Colorado on 29 Jan - 2 Feb 1980.

Working on KMC development tools:

Bill Jolitz  
U. C. Berkeley  
Berkeley, CA 94720  
415/642-7359

There is a bug in the ECC disk code. Look in pi\_get / pi\_put;  
the "mov hipri" should be "mov \$hipri". There is also a problem  
in cylinder scheduling for the RMO3. It needs to allow a lead  
of six sectors instead of four (doesn't allow enough time).  
There is a one line difference in the code generation passes of  
the C compiler between the phototypesetter version and the PWB  
version.

Digital Pathways has information on using their chronolog clock  
with Unix. Contact Bob Lumms 212/430-2579.

A MINI-UNIX is available for the LSI-11 with AED 6200 LP  
floppies. Requires a MINI-UNIX license. Send \$25 and two  
floppies to:

Dr. W. C. Gore  
Dept. of Electrical Engineering  
Barton Hall  
Johns Hopkins U.



Conference Report  
Baltimore, MD 21218

26 June 1979

Adventure T-shirts. Has picture of dragon and related stuff. Says something about computer gaming. Color yellow or blue. Size S, M, L.

Charles Andres  
Digital Equipment Corp.  
One Iron Way  
Marlboro, MA 02222

Has our Pascal and has made many improvements to it. I've requested a copy.

Mike O'Shaughnessy  
Dept. Math and C.S.  
U. New Hampshire  
Durham, NH 03824

BBN has a TECO for Unix.

Software distributions. As indicated, many people are willing to send out their own work. Most want a tape, mailing box and label, and return postage. Beyond that, there were five distributions for the conference. Unfortunately, the hosts could only make copies for a few people. The distributions are:

conference	collection of much of the stuff discussed
mini-unix	Hopkins or Toronto's (?) Unix for LSI-11
tools	collection of stuff for software tools conference
purdue	heavily modified system to support many users
toronto	graphics

The tools and toronto tapes need 1200 feet, the rest are 2400 feet. People getting tapes were:

Chris Boylan (all. Tapes returned COD)  
SICL

143 Space Science Center  
U. Minnesota  
Minneapolis, MN 55455

Michael Wendell (all except mini)  
GTE - AEL Phoenix  
11226 N. 23rd Ave.  
Phoenix, AZ 85029

Arthur Hayes (all except toronto)  
Johns Hopkins U.  
EE Dept., Barton Hall  
Baltimore, MD 21218

George Goble (all)  
Electrical Engineering  
Purdue U.  
W. Lafayette, IN 47907

John Bass @ SRI-UNIX (getting all)

There were 30 notices for places looking for one or more Unix programmers.

# A REPLY

## MEMORANDUM

TO

Jan J

FROM

Dave H

DATE

5 SW

I was most amused by your photocopy of my little note stuck on the terminal (CAVIGN #4 p4). Apparently you don't know the full story. The terminal that was in question was a Serge-Gyre terminal obtained in exchange for the G-740. It would appear that whenever the terminal was in use, the DJ-11 driver inevitably jammed up when servicing it. Removing the terminal seemed to cure the problem... And still on the subject of

## MEMORANDUM

TO

FROM

DATE

Serge terminals did you know that switching off the electric jyg near it caused it to echo the last character typed? It got so bad that people used to type in 'D' whenever someone was using the jyg. One was unable to control the jyg by use of the CONTROL/J key, however. None of the other terminals display this "feature".

Fond regards  
DH